Universitat de Girona

COVERAGE PATH PLANNING FOR AUTONOMOUS UNDERWATER VEHICLES

Enric Galceran

Dipòsit legal: Gi. 898-2014
http://hdl.handle.net/10803/133832

Universitat de Girona

Ph.D. Thesis

COVERAGE PATH PLANNING FOR AUTONOMOUS UNDERWATER VEHICLES

ENRIC GALCERAN

2014

Universitat de Girona

Ph.D. Thesis

COVERAGE PATH PLANNING FOR AUTONOMOUS UNDERWATER VEHICLES

ENRIC GALCERAN

2014

Doctoral Program in Technology

Supervised by:
Dr. Marc Carreras

Thesis submitted to the University of Girona in fulfillment of the requirements for the degree of Doctor of Philosophy

CERTIFICAT DE DIRECCIÓ DE TESI

Dr. Marc Carreras, del Departament d'Arquitectura i Tecnologia de Computadors de la Universitat de Girona,

DECLARO:

Que el treball titulat Coverage Path Planning for Autonomous Underwater Vehicles, que presenta Enric Galceran per a l'obtenció del títol de doctor, ha estat realitzat sota la meva direcció i que compleix els requisits per poder optar a Menció Internacional.

I, perquè així consti i tingui els efectes oportuns, signo aquest document.

Girona, Març de 2014

Dr. Marc Carreras

To Jordi, Mari and Darija.

ACKGRAÏHVALIMENTI

Han estat moltes les persones que m'han infuenciat, animat i ajudat durant els anys al CIRS que m'han dut a completar aquesta tesi. A totes elles els dec una gran gratitud. La meva llarga llista de gràcies comença amb el meu director de tesi, en Marc Carreras, que em va donar l'oportunitat d'immergir-me (i no va amb segones) en el meravellós món de la robòtica i que sempre m'ha aconsellat amb saviesa, circumspecció i enteniment de la recerca.

Així com en Marc, en Pere Ridao i en David Ribas sempre han estat oberts a dedicar-me temps en interminables discussions sobre la meva recerca. Puc dir vertaderament que ells han estat els meus altres directors de tesi.

I am indebted with Howie Choset, who hosted me in his lab at CMU и с Владимиром Ђапићем, ко је био мој домаћин у NURC-у. When time allowed, Howie provided rigorous and valuable feedback with his deep knowledge of robotics and math. Moreover, he introduced me to the mouth-watering goodness of bagels and cream cheese (I will never forget that first taste). Задовољство је радити с Владимиром, он ме је упознао с италијанским језиком и тестирао је моју издржљивост на мору, у разним временским условима на броду Леонардо, током три недеље проведене на Елби.

Els meus companys de laboratori Narcís, Tali, Sharad και Άγγελος han estat sempre amablement disposats a ajudar-me quan feia falta una solució efectiva de programació, un pedaç ràpid, agilitat matemàtica ή εμπειρία (no necessàriament en ordre respectiu). Nuno também foi com seu extenso conhecimento de visão computacional. De l'Andrés i l'Emili n'he apreciat les cruels i necessàries dosis de realitat que donaren a aquest pobre estudiant al principi dels seus dies de doctorat, sempre assegurant-se que tocava de peus a terra. Un moltes gràcies ben gros va per en Lluís per tenir sempre els robots a punt per fer experiments al món real i per atendre pacientment les meves demandes de configuració. I agraeixo als meus altres companys de laboratori, Arnau, Albert, Guillem, Jep, Juan David, Carles i Eduard que sempre estiguin a punt per fer-la petar sobre el que sigui. És gràcies a aquests petits interludis que la inspiració apareix a la llarga. I no podria oblidar-me dels altres membres dels equips amb els que vam prendre part a la competició SAUC-E, Miki, Carlos, Simone, 李志星, Sebas, Xevi, Simó i Pere, quedant campions el 2010 i deixant el llistó ben alt el 2011.

UWSim, el software de simulación submarina, ha sido una herramienta de crucial importancia en el desarrollo y prueba de los algoritmos de esta tesis. Estoy sinceramente agradecido a sus desarrolladores, especialmente a Mario Prats, por haberlo desarrollado y por proporcionar un valioso soporte a la comunidad de usuarios.

Estic agraït a la Joseta, la Mireia i l'Anna per organitzar els viatges a conferències, solucionar tot tipus de maldecaps burocràtics i muntar fantàstiques activitats en grup que ens han fet a tots feliços.

Dec un gran agraïment al meu col·lega Ricard que, amb el seu poderós kung fu de reconstrucció 3D, ha contribuït elegantment a il·lustrar els resultats de cobertura 3D d'aquesta tesi. I agraït estic als meus amics que estan també atrapats (o ho estaven, no fa tant) en un doctorat, Masi, Albert Pla, Gerard, Mariano, Pablo, Albert Gubern, Eloy, Pepe, Marc y Sergio per estar nominalment a punt per passar-ho bé, es tracti de córrer, pàdel, futbol o (sempre inicialment sagaces i cada vegada més divergents) discussions entre cerveses.

Na red dolazi Hrvatska mafija, Nikola Mišković, Đula Nađ, Matko Barišić i Damjan Miklić, koji su me srčano ugostili na FER-u i vodili me uokolo za vrijeme mojih posjeta Zagrebu. Također Lela, Martina, Maša, Sanja, Ivana, Hrvoje, Neno i Zdravko zbog kojih sam se tamo osjećao kao da sam kod kuće. Ako postoji nešto kao Hrvatska mafija, tada je Marin "Stipe" Stipanov, neupitno, capo. Marin je bio moj dobar prijatelj od kad smo se upoznali u NURC-u, te je kontinuirano dokazivao svoju jedinstvenost dijeleći samnom svoju neprikosnovenu mudrost, pomažući mi na mome putu ka prosvjetljenju, te vodeći me oko, a i preko, granice hrvatskog vokabulara.

Nel NURC, Alberto Grati e Stefano Fioravanti mi hanno fatto conoscere Gemellina, l'allegro ASV e Goldrake, il grintoso UUV, e preso cura di me durante le mie vacillante esperienze a bordo del CRV Leonardo. Così ha fatto il resto dell'equipaggio del Leonardo, e Giacomo, Brendan, Brett, Rubén and so many others, from the shore, contributed to making my time there such a delightful experience. Also at NURC, Warren Fox, David Williams y Alberto Alvarez made time to review my work and provided fruitful discussions. At CMU, mein Mitbewohner und Laborkollege Florian and lab mates Connie, Dave, Ross, Steve, Khoudor, Nico, Cornell, Евгении, 超慧 and Matt Tesch always made me feel a true member of the group by showing me around Pittsburgh, bowling at Forward Lanes, helping out with math or going to Hemingway's, and Glenn Wagner and Matt Travers gave me a boost by repeatedly ruining my research (although I must admit it did not feel that edifying in the first place). And I also thank the people from pickup ultimate and football at CMU (I continue to refuse "soccer", such an abominable term) for all those wonderful games. In addition, I am grateful to Franz Hover, Brendan Englot, Josh Leighton, Heather Beem, Amy Gao and Eduard Valera for warmly welcoming me at MIT.

I appreciate the valuable feedback that Ιωάννης Ρεκλείτης and the anonymous reviewers of our papers have provided throughout these years.

Voldria agraïr als contribuents de Catalunya and the EU que hagin donat suport econòmic a aquesta tesi amb el seu deute amb la banca. Esperem que aviat tots plegats poguem finançar la recerca amb diners d'una economia real al servei de la gent i no pas al d'avariciosos jugadors compulsius.

Finalment, dec el més especial dels agraïments a les tres persones a qui va dedicada aquesta tesi. Als meus pares, Jordi i Mari, per portar-me a la vida i preparar-me amb una forta ètica de treball sense la qual no hauria mai aconseguit res. I, na kraju, Dariji, jer kad rad postane samo mali dio života doista letiš slobodno.

ACKNOWLEDGMENTS

Many persons have influenced, encouraged, and helped me during my years at CIRS that have led to the completion of this thesis. To them I owe a great deal of gratitude. My long list of thank you's starts with my advisor, Marc Carreras, who gave me the opportunity to dive (no pun intended) into the wonderful world of robotics and who has always counseled me with his wisdom, circumspection and insight into research.

Just like Marc, Pere Ridao and David Ribas have always been willing to make time for me and engage in never-ending discussions about my research. As far as I am concerned, they have been my back-up advisors.

I am indebted with Howie Choset, who hosted me in his lab at CMU and with Vladimir Djapic, who was my host advisor during my stint at NURC. When time allowed, Howie provided rigorous and valuable feedback with his deep knowledge of robotics and math. Moreover, he introduced me to the mouth-watering goodness of bagels and cream cheese (I will never forget that first taste). Vladimir is a pleasure to work with, he got me started with the Italian language and tested my resilience to a variety of sea states aboard CRV Leonardo during three weeks of sea trials in Isola d'Elba.

My lab mates Narcís, Tali, Sharad and Angelos have been always nice and helpful whenever an effective robot programming solution, a quick workaround, math agility or experience were required (not necessarily respectively). So has been Nuno Gracias with his extensive computer vision knowledge. From Andrés and Emili I appreciate the mean and necessary doses of reality they gave to this wishful grad student of myself in the early days, always making sure I had sufficient situational awareness in grad school. A big thank you goes to Lluís for always having the robots up to speed for real-world experiments and for patiently attending my configuration demands. I thank my other lab mates, Arnau, Albert, Guillem, Jep, Juan David, Carles and Eduard for being always down for a cheerful chat. It is thanks to these short breaks that inspiration eventually pops up. And I couldn't omit the other team members that took part with me in the SAUC-E underwater robotics competition, Miki, Carlos, Simone, Chee Sing, Sebas, Xevi, Simó and Pere, championing in 2010 and doing awesome in 2011.

UWSim, the underwater simulation package, has been a tool of utmost importance in developing and testing the algorithms of this thesis. I am sincerely grateful to its developers, especially Mario Prats, for developing it in the first place and providing valuable support to the user community.

I am grateful to Joseta, Mireia and Anna for sorting out conference trips, going through the hassle of all sorts of paperwork and organizing awesome team building events that kept all us happy.

I owe a great deal of gratitude to my friend Ricard who, with his powerful 3D reconstruction kung fu, has contributed beautifully to illustrate the 3D coverage results of this thesis. And grateful I am to my friends who are also trapped (or were so, not so long ago) in grad school, Masi, Albert Pla, Gerard, Mariano, Pablo, Albert Gubern, Eloy, Pepe, Marc and Sergio for being nominally ready for great fun, whether it was running, padel, football or (always initially witty and increasingly deviating) arguments among beers.

And here comes the Croatian mafia, Nikola Mišković, Đula Nađ, Matko Barišić, and Damjan Miklić, who kindly hosted me at FER and showed me around in my visits to Zagreb. Likewise, Lela, Martina, Maša, Sanja, Ivana, Hrvoje, Neno and Zdravko made me feel there like I was at home. And if there is such a thing as a Croatian mafia, then Marin "Stipe" Stipanov is undoubtedly the capo. Marin has been my good friend since we met at NURC and continuously proved to be one of a kind by sharing with me his unparalleled wisdom, helping me get enlightened, showing me around and beyond the limits of Croatian vocabulary and teaching me how to embrace the stochastic nature of the cosmos.

Many high quality individuals have made my stints abroad just awesome experiences. At NURC, Alberto Grati and Stefano Fioravanti introduced me to Gemellina, the cheerful ASV and Goldrake, the gutsy UUV and took care of me during my shaky experiences aboard CRV Leonardo. So did the rest of Leonardo's crew, and Giacomo, Brendan, Brett, Rubén and so many others, from the shore, contributed to making my time there such a delightful experience. Also at NURC, Warren Fox, David Williams and Alberto Alvarez made time to review my work and provided fruitful discussions. At CMU, my room mate and lab mate Florian and lab mates Connie, Dave, Ross, Steve, Khoudor, Nico, Cornell, Evgeni, Chaohui and Matt Tesch always made me feel a true member of the group by showing me around Pittsburgh, bowling at Forward Lanes, helping out with math or going to Hemingway's, and Glenn Wagner and Matt Travers gave me a boost by repeatedly ruining my research (although I must admit it did not feel that edifying in the first place). And I also thank the people from pickup ultimate and football at CMU (I continue to refuse "soccer", such an abominable term) for all those wonderful games. And I am grateful to Franz Hover, Brendan Englot, Josh Leighton, Heather Beem, Amy Gao and Eduard Valera for warmly welcoming me at MIT.

I appreciate the valuable feedback that Ioannis Rekleitis and the anonymous reviewers of our papers have provided throughout these years.

I would like to thank the taxpayers from Catalonia and the EU for sponsoring this research with their debt with the banks. Hopefully soon enough we will be able to sponsor research with real-world money at the service of people and not of some miser compulsive gamblers.

Finally, I owe the most special thank you of them all to the three persons to whom this thesis is dedicated. To my parents, Jordi and Mari, for bringing me to life and endowing me with a strong work ethic, without which I would never have accomplished anything. And, lastly, to Darija, because it is when work becomes only a small part of life when you truly fly free.

PUBLICATIONS

The work developed in this thesis led to the following publications:

JOURNAL ARTICLES

- [JBR'13] Enric Galceran, Narcís Palomeras and M. Carreras. *Profile Following for Inspection of Underwater Structures.* Paladyn Journal of Behavioral Robotics. Volume 4, Issue 4, December 2013, Pages 209-220.
- [RAS'13] Enric Galceran and M. Carreras. *A Survey on Coverage Path Planning for Robotics.* Robotics and Autonomous Systems. Volume 61, Issue 12, December 2013, Pages 1258-1276.

PEER-REVIEWED CONFERENCES AND WORKSHOPS

- [ICRA'14] Enric Galceran, Ricard Campos, Narcís Palomeras, Marc Carreras and Pere Ridao. *Coverage Path Planning with Realtime Replanning for Inspection of 3D Underwater Structures.* International Conference on Robotics and Automation (ICRA). Hong Kong, China. June 2014 (to appear).
- [IROS'13] Enric Galceran, Sharad Nagappa, Marc Carreras, Pere Ridao and Albert Palomer. *Uncertainty-driven Survey Path Planning for Bathymetric Mapping.* Intelligent Robots and Systems (IROS). Tokyo, Japan. November 2013.
- [ICRA'13A] Enric Galceran and Marc Carreras. *Planning Coverage Paths on Bathymetric Maps for In-Detail Inspection of the Ocean Floor.* International Conference on Robotics and Automation (ICRA). Karlsruhe, Germany. May 2013.
- [ICRA'13B] Ross Hatton, Ross A Knepper, Howie Choset, David Rollinson, Chaohui Gong and Enric Galceran. *Snakes on a Plan: Toward Combining Planning and Control.* International Conference on Robotics and Automation (ICRA). Karlsruhe, Germany. May 2013.
- [IROS'12] Enric Galceran and Marc Carreras. *Efficient Seabed Coverage Path Planning for ASVs and AUVs.* Intelligent Robots and Systems (IROS). Vilamoura, Portugal. October 2012.
- [NGCUV'12] Enric Galceran, Vladimir Djapic, Marc Carreras and David P. Williams. *A Real-time Underwater Object Detection Algorithm for Multi-beam Forward Looking Sonar.* IFAC's workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV). Porto, Portugal. April 2012.

- [MARTECH'13] Enric Galceran, Ricard Campos and Marc Carreras. *Automating Seafloor Inspection using Autonomous Underwater Vehicles.* Fifth International Workshop in Marine Technology (MARTECH). Girona, Catalonia (Spain). October 2013.

- [OCEANS'13] Enric Galceran, Marc Carreras, Narcís Palomeras and Pere Ridao. *Complex Structure Profile Estimation and Following with the GIRONA500 AUV.* OCEANS'13. Bergen, Norway. June 2013.

- [AUTOMAR'12] Enric Galceran. *Coverage Path Planning for In-Detail Seafloor Inspection.* V Jornadas Automar. Girona, Catalonia (Spain). December 2012.

- [OCEANS'12] Enric Galceran and Marc Carreras *Coverage Path Planning for Marine Habitat Mapping.* OCEANS'12. Hampton Roads (VA), USA. October 2012.

- [OCEANS'11] Emili Hernández, Marc Carreras, Enric Galceran and Pere Ridao. *Path Planning with Homotopy Class Constraints on Bathymetric Maps.* OCEANS'11. Santander, Spain. June 2011.

ACRONYMS

AHRS Attitude and Heading Reference System

ASV Autonomous Surface Vehicle

AUV Autonomous Underwater Vehicle

BSA Backtracking Spiral Algorithm

CMRE Centre for Maritime Research and Experimentation

CIRS Underwater Robotics Research Center, *Centre d'Investigació en Robòtica Submarina*

CMU Carnegie Mellon University

COLA2 Component-Oriented Layer-based Architecture for Autonomy

DOF Degree Of Freedom

DVL Doppler Velocity Log

FER Faculty of Electrical and Computer Engineering, *Fakultet Elektrotehnike i Računarstva*

FLS Forward-Looking Sonar

FOV Field Of View

GVD Generalized Voronoi Diagram

LBL Long BaseLine

MCM Mine CounterMeasures

MDP Markov Decision Process

MIT Massachusetts Institute of Technology

MOOS Mission Oriented Operating Suite

MSTC Multi-robot Spanning Tree Coverage

NURC NATO Undersea Research Centre

POMDP Partially Observable Markov Decision Process

PRM Probabilistic Roadmap

ROI Region Of Interest

ROS Robot Operating System

ROV Remotely Operated Vehicle

RRT Rapidly-exploring Random Tree

SAS Synthetic Aperture Sonar

SAUC-E Student Autonomous Underwater Challenge - Europe

SLAM Simultaneous Localization And Mapping

SSS Side-Scan Sonar

STC Spanning Tree Coverage

STOMP Stochastic Trajectory Optimization for Motion Planning

TRN Terrain-Relative Navigation

TSP Traveling Salesman Problem

UAV Unmanned Aerial Vehicle

USBL Ultra-Short BaseLine

UUV Unmanned Underwater Vehicle

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

CONTENTS

ABSTRACT

At present, a mission to survey the ocean floor with an Autonomous Underwater Vehicle (AUV) is typically planned by selecting a list of waypoints that then the vehicle will automatically navigate through while keeping a safe distance from the bottom. Nonetheless, this approach has major drawbacks: (1) it does not allow the vehicle to safely operate amidst protrusions on the sea floor; (2) when traversing rugged terrain, the vehicle is forced to keep a conservative altitude, limiting applications requiring close proximity such as photomosaicing; (3) when inspecting 3D, protruding structures from the bottom the vehicle can only perceive them from an overhead viewpoint, resulting in poor data collected from these sites; (4) it does not account for the effect of the terrain on the vehicle's position estimates.

To address these limitations, this thesis proposes new methods to find collision-free paths allowing an AUV to cover an area of the ocean floor with its sensors, which is known as coverage path planning. First, we propose a coverage path planning method to plan 2D, safe-altitude surveys which provides a principled way to account for obstacles in AUV survey planning. Its main advantage is that it minimizes redundant coverage when the vehicle navigates at constant depth, leading to shorter paths. Second, we provide a method to account for the uncertainty in the vehicle position estimates when planning 2D surveys. The method minimizes the uncertainty induced by the path and leads to better maps of the ocean floor as a result. Third, we provide a coverage path planning method suitable for inspecting areas of the ocean floor including 3D structures. The resulting coverage paths enable applications requiring close proximity and allow viewpoints for full 3D sensing of the structure. Moreover, by contrast to most existing methods, we provide two techniques to adapt the planned path in realtime using sensor information acquired on-line during the mission, rather than only planning the path off-line and relying on the unrealistic assumption of an idealized path execution by the AUV. The proposed methods are validated in simulation and in experiments with a real-world AUV.

RESUM

Actualment, una missió per a cartografiar el fons marí amb un vehicle submarí autònom és planificada típicament seleccionant una llista de punts a través dels quals el vehicle navegarà automàticament mentre es manté a una distància segura del fons. Tanmateix, aquesta metodologia presenta considerables inconvenients: (1) no permet al vehicle operar amb seguretat entre obstacles que sobresurten del fons; (2) en recórrer terreny escarpat, el vehicle és forçat a mantenir una altitud conservadora, impedint tasques que requereixen proximitat al fons, com ara la creació de fotomosaics; (3) en inspeccionar estructures 3D que sobresurten del fons, el vehicle pot només percebre-les des d'un punt de vista zenital, obtenint dades de baixa qualitat com a resultat; (4) no té en compte l'efecte del terreny sobre les estimacions de posició del vehicle.

Amb l'objectiu d'esmenar aquests inconvenients, aquesta tesi proposa nous mètodes per generar camins lliures de col·lisions per a vehicles submarins autònoms que permeten cobrir una àrea del fons de l'oceà usant els sensors del vehicle, tasca coneguda com a planificació de camins de cobertura. Primer, proposem un mètode de planificació de camins de cobertura per a planificar missions en un espai 2D a una altitud segura, proporcionant una manera fonamentada de tenir en compte obstacles en la planificació de missions per a vehicles submarins autònoms. L'avantatge principal del mètode proposat és que minimitza la cobertura redundant sorgida quan el vehicle navega a profunditat constant, obtenint camins més curts com a resultat. Segon, presentem un mètode per tenir en compte la incertesa de les estimacions de posició del vehicle durant la planificació de missions 2D. Aquest mètode minimitza la incertesa induïda pel camí i genera mapes més acurats del fons oceànic com a resultat. Tercer, presentem un mètode de planificació de camins de cobertura adequat per inspeccionar àrees del fons oceànic amb estructures 3D. Els camins de cobertura resultants permeten tasques que requereixen proximitat al fons i permeten una completa percepció 3D de les estructures d'interès. A més, a diferència de la majoria dels mètodes existents, proporcionem dues tècniques per adaptar els camins planificats en temps real utilitzant informació sensorial adquirida durant la missió, per contra de planificar només abans de l'execució de la missió i confiar en la poc realista assumpció d'una execució idealitzada del camí per part del vehicle. Els mètodes proposats han estat validats en simulació i en experiments amb un vehicle submarí autònom real.

RESUMEN

Actualmente, una misión para cartografiar el fondo marino con un vehículo submarino autónomo es planificada típicamente seleccionando una lista de puntos a través de los cuales el vehículo navegará de forma automática mientras se mantiene a una distancia segura del fondo. No obstante, esa metodología presenta considerables inconvenientes: (1) no permite operar con seguridad entre obstáculos que sobresalen del fondo; (2) al recorrer terreno escarpado, el vehículo es forzado a mantener una altitud conservadora, impidiendo tareas que requieren proximidad al fondo, como por ejemplo la creación de fotomosaicos; (3) al inspeccionar estructuras 3D que sobresalen del fondo, el vehículo puede sólo percibirlas desde un punto de vista cenital, obteniendo datos de baja calidad como resultado; (4) no tiene en cuenta el efecto del terreno sobre las estimaciones de posición del vehículo.

Con el objetivo de enmendar esos inconvenientes, en esta tesis se proponen nuevos métodos para generar caminos libres de colisiones para vehículos submarinos autónomos que permiten cubrir un área del fondo oceánico utilizando los sensores del vehículo, tarea conocida como planificación de caminos de cobertura. Primero, proponemos un método de planificación de caminos de cobertura para planificar misiones en un espacio 2D a una altitud segura, proporcionando una forma fundamentada de tener en cuenta obstáculos en la planificación de misiones para los susodichos vehículos submarinos. La ventaja principal del método propuesto es que minimiza la cobertura redundante surgida cuando el vehículo navega a profundidad constante, obteniendo caminos más cortos como resultado. Segundo, presentamos un método para tener en cuenta la incertidumbre de las estimaciones de posición del vehículo durante la planificación de misiones 2D. Este método minimiza la incertidumbre inducida por el camino y genera mapas más precisos del fondo oceánico como resultado. Tercero, presentamos un método de planificación de caminos de cobertura adecuado para la inspección de áreas del fondo oceánico con estructuras 3D. Los caminos de cobertura resultantes permiten tareas que requieren proximidad al fondo y permiten una completa percepción 3D de las estructuras de interés. Además, a diferencia de la mayoría de métodos existentes, proporcionamos dos técnicas para adaptar los caminos planificados en tiempo real utilizando información sensorial adquirida durante la misión, en vez de confiar en la poco realista asunción de una ejecución idealizada del camino por parte del vehículo. Los métodos propuestos han sido validados en simulación y en experimentos con un vehículo submarino autónomo real.

1

INTRODUCTION

"Don't blame you," said Marvin and counted five hundred and ninety-seven thousand million sheep before falling asleep again a second later.

— Douglas Adams, "The Hitchhiker's Guide to the Galaxy" (1979)

In this chapter, we introduce the real-world problem that has motivated this thesis: inspection of the ocean floor by an Autonomous Underwater Vehicle (AUV). This challenge has motivated our development of new path planning techniques that achieve sensor coverage of both 2-dimensional, effectively planar regions and 3-dimensional, complex structures of the ocean floor while accounting for the particulars of the underwater domain. We first detail the motivation of this thesis and state the general problem it addresses in Section 1.1. Next, we present the goal and objectives of this thesis in Section 1.2 and briefly introduce the context where this work has been carried out in Section 1.3. Lastly, we overview the contents of this thesis and state its contributions in Section 1.4.

1.1 MOTIVATION AND PROBLEM STATEMENT

"Space... the final frontier." This legendary phrase not only has dazzled the millions of fans of the Star Trek franchise, but it arguably captures what has been the focus of discovery and exploration in the last fifty years. The Space Race led to putting the first man in orbit and the first man on the Moon in the 1960s and following efforts brought on the launch of space probes sent throughout the entire Solar System and even beyond, after NASA's Voyager 1 probe surpassed its limits in August 2012.

Paradoxically, we know less about the oceans on Earth than we do about the dark side of the Moon. Despite covering more than seventy percent of the Earth's surface, ninety percent of the ocean floor has not even been charted and a similar portion of it remains unseen by human eyes. Despite remarkable achievements, such as the manned descends into the Mariana Trench by bathyscaphe "Trieste" in 1960 and by submersible vessel "Deepsea Challenger" in 2012, the technology to explore the ocean's floors is still being developed.

Oceans support a great diversity of life and ecosystems, play a major role in controlling our climate, provide a large source of food, and its organisms are said to hold the promise of cures for an array of diseases. Therefore, exploring and understanding the oceans is a matter of utmost importance. Just to make some compelling cases: a drug isolated from the dogfish shark led to advances in combating blindness (Brunel et al., 2005), the horseshoe crab was crucial in developing a test for bacterial contamination (Bang, 1956) and sea urchins helped in the development of test-tube fertilization (Loeb, 1914).

Initially developed in the 1960s to perform deep-sea rescue operations, Remotely Operated Vehicles (ROVs) are valuable tools for exploration of the ocean floor. ROVs are unmanned, tethered submersible vehicles operated by a crew aboard a vessel, involving a ship. This way, they enable exploration of deep regions far beyond reach of human divers while avoiding risk to human lives. Figure 1 shows one of the many ROVs in operation today. ROVs are heavily used in the offshore oil industry, where they have become an essential tool without which the offshore hydrocarbon extraction would not have been possible. Nonetheless, ROVs are also used in a variety of science applications (Hudson et al., 2005), allowing exploration and sampling of zones of geological or biological interest, and have been used to locate many historic shipwrecks, including that of the RMS Titanic (Ward, 2012).

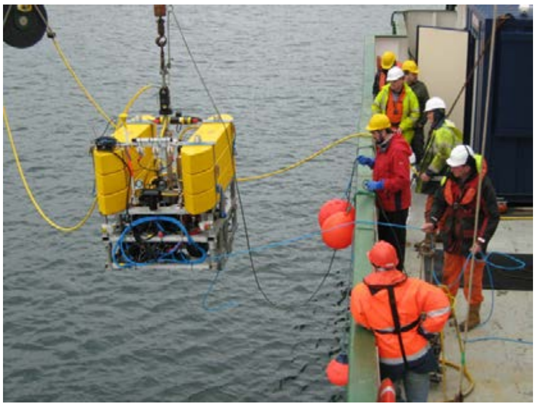


Figure 1: The LATIS ROV being launched at sea. Image credit: Mobile & Marine Robotics Research Center, University of Limerick.

Despite their valuable capabilities, ROV operations require a complex infrastructure involving a ship with its corresponding crew and specialized ROV operators. The ship must be equipped with a crane for deployment and recovery, a winch with sufficient tether for the desired operation depth and, in some scenarios, even a tether management system to minimize drag in the presence of currents. Moreover, operations may take place far from the coast and last for several days or even weeks, leading to a high cost of operation. On the other hand, the autonomy of ROVs is severely limited by the tether, restricting the extent of the area of operation in a single mission.

To address these limitations, efforts have been made to provide ROVs with a higher level of autonomy. The focus has been to remove the tether to expand the vehicle's capabilities and, at the same time, reduce the cost of operation. This is achieved by equipping the submersible with its own power source, and the capability to determine its actions based on inputs from its onboard sensors and a pre-defined mission plan. The result of those efforts are AUVs. Figure 2 shows a torpedo-shaped AUV; as illustrated in this figure, AUVs are often smaller and easier to deploy than their ROV counterparts.

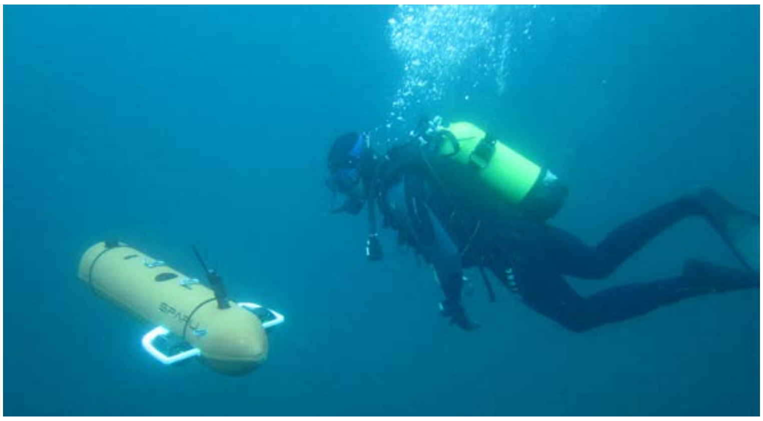


Figure 2: The Sparus AUV (left) next to a human diver.

Thanks to technology breakthroughs in the last two decades, AUVs have become a standard tool for mapping the ocean floor using optical (Eustice et al., 2006) and acoustic (Paduan et al., 2009) sensors, with applications to dam inspection (Ridao et al., 2010), marine geology (Escartin et al., 2008; Yoerger et al., 2000), underwater archaeology (Bingham et al., 2010) and Mine CounterMeasures (MCM) (Williams, 2010) to mention but a few. Particularly remarkable is the role AUVs have played in bathymetric mapping[*] for charting the ocean floor. Bathymetric mapping supports safe navigation, helps protect and monitor marine areas of biological interest and is key to geology, archaeology and military applications. Thanks to near-bottom surveys, AUVs provide high resolution maps and require little human supervision compared to their ship- or ROV-assisted counterparts, and hence at a lower cost.

Recently, the number of AUVs in operation throughout the world has increased remarkably (Nicholson and Healey, 2008). This is largely because of their excellent ability to explore and assess underwater environments. As the technology development progresses at a steady pace, the cost of AUVs is decreasing and becoming available to an equally increasing number of people. Whether it is mapping the ocean floor, assessing a naval mine threat or collecting oceanographic data, these vehicles provide users a great resource for better understanding of the ocean in general.

Surveying an area of interest on the ocean floor by completely covering its extent with the vehicle's sensors is an integral task in most AUV applications. Examples include microbathymetric mapping, image photomosaicing and searching for a target object. At present, these missions are typically pre-

[*] Bathymetry is the study of underwater depth of lake, river or ocean floors. In other words, bathymetry is the underwater equivalent to hypsometry (the measurement of land elevation relative to sea level).

programmed by a human operator selecting a list of waypoints on an *a priori* chart of the target area. A typical survey mission path is composed of a sequence of straight lines that connect at waypoints in a "mowing the lawn" manner, as shown in Figure 3. Once downloaded to the AUV, the vehicle executes the mission while keeping a safe altitude from the sea bottom.

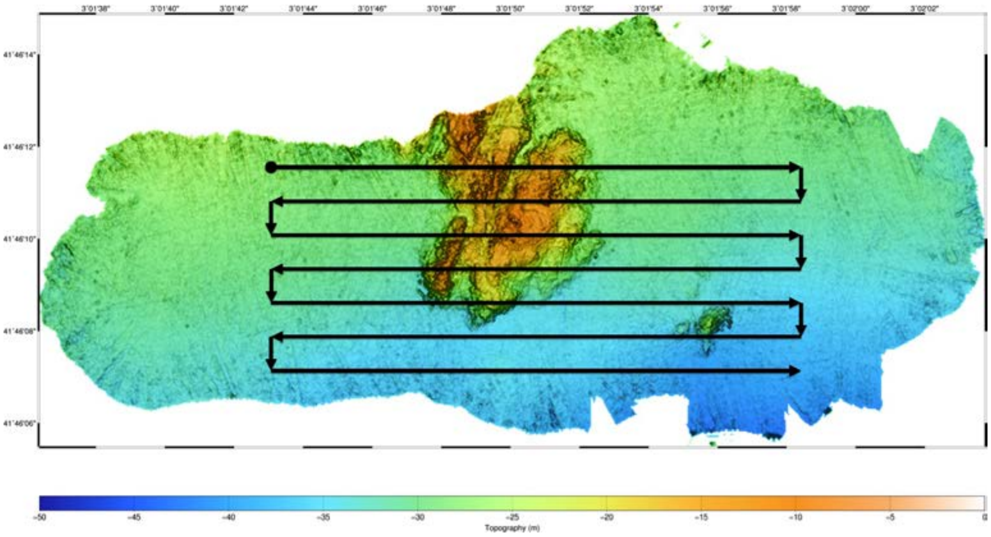


Figure 3: Typical "mowing the lawn" AUV survey path planned on a diving site bathymetric chart off the Costa Brava in Catalonia (Spain).

This is an approach that suits most commercially available AUVs, which are typically torpedo-shaped and have very limited obstacle avoidance and maneuvering capabilities. Nevertheless, this approach has several important drawbacks, and, especially, it imposes serious limitations to the applicability of an increasingly developing new class of AUVs which are able to hover and maneuver with more agility: the so-called hovering AUVs. In particular, the aforementioned AUV survey planning approach presents the following drawbacks. First, it has no principled way of accounting for obstacles in the survey path planning phase, precluding the vehicle from safely operating amidst rugged protrusions on the sea floor. Second, due to the limited obstacle avoidance capabilities of most AUVs, they are forced to fly at a conservative altitude to avoid collisions when traversing rugged terrain. This hinders coverage operations such as optical imaging that require to survey the bottom in close proximity for acceptable visibility. Third, when surveying 3-dimensional structures on the ocean floor such as coral reefs or shipwrecks, the overhead point of view provided by these survey paths results in poor imaging of the target structure. Lastly, while state-of-the-art AUV navigation techniques seek to match the vehicle's sensor measurements to landmarks in the terrain, standard lawnmower-type survey paths do not look for discriminative actions that lead to more accurate position estimates. Clearly, an automated survey path planning method that addresses these drawbacks is highly desirable.

This thesis addresses the aforementioned limitations of AUV survey mission planning in the context of coverage path planning, which is the task of passing a sensor over all points in a target surface or volume of interest while avoiding obstacles. This task is integral to many robotic applications, such as vacuum cleaning robots, painter robots, lawn mowers and automated harvesters and, indeed, AUV surveys. Although, as we will see in Chapter 2, a considerable body of research has addressed coverage path planning, little attention has been paid to the coverage path planning problem in underwater environments. While many underwater robotics applications such as microbathymetric mapping, habitat monitoring, image photomosaicing or target search and recovery can benefit greatly from the complete coverage guarantees and robustness of coverage path planning methods, prior research in coverage path planning has not addressed the aforementioned limitations of standard AUV survey planning. In fact, while coverage path planning has been applied to a variety of problems in robotics and automation, it has been mostly demonstrated in controlled environments including absolute positioning. Under the surface, by contrast, there is no ubiquitous absolute positioning (such as GPS), sensors are often limited by the narrow bandwidth and low accuracy of underwater acoustics and AUVs are exposed to currents and all sorts of unexpected disturbances, just to name but a few of the numerous challenges of underwater robotics. Therefore, studying the coverage path planning problem in the context of the underwater domain opens the door to pushing the envelope of AUV capabilities in particular and of ocean exploration and understanding in general.

1.2 GOAL OF THE THESIS

Once the motivation and the problem addressed in this thesis have been set forth, the goal of this thesis is stated as follows:

To develop coverage path planning techniques for autonomous underwater vehicles that endow them with a principled way to account for obstacles, allow to survey the ocean floor in close proximity, provide suitable viewpoints for imaging applications and take into account the effect of the terrain being traversed on the position estimates.

1.2.1 *Objectives of the Thesis*

The general goal can be categorized more specifically in the following objectives:

REVIEW OF THE COVERAGE PATH PLANNING LITERATURE. To carry out a thorough review of prior work in the coverage path planning literature and to identify requirements of the underwater domain that have not been properly addressed in said work.

2D COVERAGE PATH PLANNING FOR AUVS. To propose a coverage path planning method that provides a principled and automated manner to account for obstacles in survey mission planning for AUVs.

UNCERTAINTY-DRIVEN 2D COVERAGE PATH PLANNING FOR AUVS. To propose a coverage path planning method that takes into account the effect of the terrain on the uncertainty of AUV pose estimates to plan a survey path that leads to more accurate estimates and therefore to higher quality data products of the ocean floor.

3D COVERAGE PATH PLANNING FOR AUVS. To propose a coverage path planning method that goes beyond traditional 2D surveys at a safe altitude from the sea bottom and allows inspection of intrinsically 3-dimensional regions such as shipwrecks and coral reefs in close proximity.

VALIDATION IN A REAL-WORLD AUV. To validate the proposed techniques in a real AUV with experiments in real-world underwater environments that go beyond simulation and controlled experimentation in the laboratory.

1.3 CONTEXT

The research presented in this thesis has been carried out at the Underwater Robotics Research Center, *Centre d'Investigació en Robòtica Submarina* (CIRS) of the Computer Vision and Robotics Institute of the University of Girona. Research in underwater robotics has been ongoing there since 1992, supported by several Spanish and European research programs. The group has developed several AUV prototypes: GARBI (Amat et al., 1999), a vehicle originally conceived as a ROV that was restyled as an AUV; URIS (Batlle et al., 2004), a lightweight AUV; Ictineu (Ribas et al., 2007), which won the first Student Autonomous Underwater Challenge - Europe (SAUC-E) competition in 2006; Sparus, which championed SAUC-E in 2010 and has recently been restyled and presented as a commercial platform (Carreras et al., 2013); and GIRONA 500 (Ribas et al., 2012), a reconfigurable AUV that has been the main platform for experimentation in this thesis (see Appendix A for more details on this later vehicle).

Research at CIRS revolves around AUV applications, and has focused on control architectures (Ridao et al., 2002; Palomeras et al., 2012), model identification (El-Fakdi et al., 2003), machine learning (Carreras et al., 2001; El-Fakdi et al., 2010), mission control (Palomeras et al., 2006), mapping (Nicosevici et al., 2009) and, more recently, AUV intervention (Prats et al., 2012b; Ribas et al., 2012), Simultaneous Localization And Mapping (SLAM) (Ribas et al., 2008; Zandara et al., 2013) and path planning (Hernandez et al., 2011). Noteworthily, this thesis represents the first endeavors of the group in coverage path planning, opening a new line of research in the subject.

In addition, the work presented in this thesis has contributed to the following European research projects where CIRS took part:

- 2011-2015. FP7 EU project "MORPH: Marine robotic system of self-organizing, logically linked physical nodes".

- 2011-2014. FP7 EU project "PANDORA: Persistent autonomy through learning, adaptation, observation and re-planning".
- 2010-2013. FP7 EU project "TRIDENT: Marine robots and dexterous manipulation for enabling autonomous underwater multipurpose manipulation".

1.4 OVERVIEW AND CONTRIBUTIONS OF THE THESIS

The remainder of this thesis is organized as follows. In Chapter 2 we review the algorithms and applications of coverage path planning presented in the literature, providing a classification of the most popular and successful and pointing out directions for further research in the problem. This review comprises foundational building blocks of the algorithms developed in this thesis. In addition, this review represents the first coverage path planning literature survey in more than a decade. Therefore, it is an important contribution to the field capturing the state of the art and focusing on the breakthroughs occurred in the last ten years.

In Chapter 3 we introduce a new 2D coverage path planning method providing a principled way to account for obstacles in AUV survey path planning. The key innovation of this method is that it minimizes redundant coverage incurred when sweeping the ocean floor from a constant depth. In this situation, the vehicle's sensor footprint varies according to the distance to the bottom, leading to redundant coverage among the back-and-forth laps of the coverage path. The proposed method uses information on an *a priori* bathymetric map of the target region to minimize this redundant coverage. As a result, a shorter, more efficient coverage path is obtained. The proposed method is validated in simulation experiments conducted with a real-world bathymetric dataset. Results show a significant increase on path efficiency in comparison with a standard coverage path.

In Chapter 4 we present a novel 2D survey path planning algorithm for area coverage which minimizes the robot's position uncertainty along the planned path. Rather than avoiding obstacles, which can be achieved with the method presented in Chapter 3, the focus of this algorithm is to plan a coverage path that leads to better position estimates and hence to more accurate data products resulting from AUV surveys. The proposed algorithm especially targets bathymetric mapping applications and respects application constraints such as the desire to survey in parallel tracks. Our proposed algorithm uses the saliency on an *a priori* map to predict how the terrain will affect the robot's belief at every point on the target area. Based on this magnitude, we compute the order in which to trace parallel tracks to cover the target area in order to minimize the overall uncertainty along the path. A particle filter keeps track of the robot's position uncertainty during the planning process and, in order to find useful loop-closures for mapping, crossing tracks that visit salient locations are added when the uncertainty surpasses a user-provided threshold. We test our method in simulation using real-world datasets collected off the coasts of Catalonia (Spain), Greece and Australia and in sea trials with a real AUV in the Catalan coast. We evaluate the expected robot's position uncertainty along the planned paths with a particle filter and their associated mapping performance using a bathymetric mapping algorithm. Results show that our method offers benefits over standard lawnmower-type paths both in terms of position uncertainty and map quality.

In Chapter 5 we introduce new 3D coverage path planning algorithms for inspection of structures on the ocean floor that can not be successfully surveyed in detail without resorting to a 3D workspace. Examples of such structures include coral reefs, shipwrecks, breakwater structures in harbors or the continental slope. We first present an off-line coverage path planning algorithm that generates a complete coverage path suitable for covering sea floor areas with 3D structures. The algorithm identifies high-slope regions on an *a priori* bathymetric map and provides different path patterns suiting both effectively planar and high-slope regions, completely covering the target area. While most coverage algorithms operate off-line and assume the path will be perfectly executed, the vehicle suffers from localization error and environmental disturbances during the path execution, rendering such an assumption unrealistic. This is especially patent when navigating amidst complex structures, where the threat of collision increases. Rather than relying on an idealized execution of the planned path, we present two approaches to deal with the discrepancy between a nominal 3D coverage plan and the actual situation the AUV faces *in situ* during the coverage task. The first approach is reactive and uses range sensor measurements to cover cross-section profiles of the target surface in realtime. The second approach uses realtime replanning to reshape the nominal coverage path according to the actual target structure perceived during the mission via the vehicle's onboard sensors. We experimentally validate the proposed algorithms both in simulation and in real-world experiments with a physical AUV in the CIRS pool and at sea.

We conclude in Chapter 6 by summarizing the work completed, reviewing the contributions of this thesis and the list of publications that have resulted from it and, finally, identifying compelling areas for future work.

2

STATE OF THE ART

"What's up?"
"I don't know," said Marvin,
"I've never been there."

— Douglas Adams, "The Hitchhiker's Guide to the Galaxy" (1979)

This chapter presents a survey of prior work in coverage path planning, which is the central problem this thesis addresses. As mentioned earlier, coverage path planning is the task of determining a path that passes over all points of an area or volume of interest while avoiding obstacles. This task is integral to many robotic applications, such as (just to name a few) vacuum cleaning robots (Yasutomi et al., 1988), painter robots (Atkar et al., 2005b), demining robots (Gage, 1994; Najjaran and Kircanski, 2000; Acar et al., 2003), lawn mowers (Cao et al., 1988; Bosse et al., 2007), automated harvesters (Ollis and Stentz, 1997), window cleaners (Farsi et al., 1994) and, indeed, AUVs (Hert et al., 1996; Englot and Hover, 2012). While the survey we present covers coverage path planning theory and applications in the general field of robotics, it highlights as well coverage path planning research focused in the underwater domain. After introducing the coverage path planning problem in Section 2.1, we first present a thorough review of coverage path planning methods for 2-dimensional (Sections 2.2-2.7) and 3-dimensional workspaces (Section 2.8), which comprise foundational building blocks of the algorithms developed in this thesis. We follow with a review of coverage path planning methods focused on achieving optimal coverage (Section 2.9) and coverage path planning methods that take uncertainty into account and intend to reduce the accumulation of localization error while executing the planned coverage path (Section 2.10). We then briefly review several multi-robot coverage path planning methods (Section 2.11). We close this chapter with a discussion of the reviewed methods placing an emphasis on the underwater domain and providing a summary table and directions for further research in coverage path planning (Section 2.12).

2.1 THE COVERAGE PATH PLANNING PROBLEM

As mentioned earlier, the coverage path planning problem calls for finding a collision-free path that allows a robot to pass over all points in a target area or volume of interest. In one of the earliest works on coverage path planning found in the literature, Cao et al. (1988) defined the requirements a robot must meet to perform a coverage operation. Albeit the target application in the aforementioned paper is a mobile robot moving in a flat 2-dimensional environment, the same criteria are applicable to other coverage scenarios. The requirements are as follows:

1. Robot must move through all the points in the target area covering it completely.
2. Robot must fill the region without overlapping paths.
3. Continuous and sequential operation without any repetition of paths is required.
4. Robot must avoid all obstacles.
5. Simple motion trajectories (e.g., straight lines or circles) should be used (for simplicity in control).
6. An "optimal" path is desired under available conditions.

However, it is not always possible to satisfy all these criteria in complex environments. Therefore, sometimes a priority consideration is required.

The coverage path planning problem is related to the covering salesman problem, a variant of the Traveling Salesman Problem (TSP) where, instead of visiting each city, an agent must visit a neighborhood of each city (Arkin and Hassin, 1994). Recall that, given a list of cities and the distances between each pair of cities, the TSP calls for the shortest route that visits each city exactly once and returns to the origin city. However, in coverage path planning, the agent must pass over all points in the target area in contrast to visiting all the neighborhoods. Since the TSP is NP-hard, the computational time required to solve the problem increases drastically when the dimension of the problem increases. Actually, finding a path to cut all the grass of a given region covered by grass, which is known as the "lawnmower problem", is proven to be NP-hard (Arkin et al., 2000). Notice that the lawnmower problem does not account for obstacles. In fact, even the basic path planning problem, known as the "piano mover's problem", of finding a collision-free path from a start configuration to a goal configuration is shown to be PSPACE-hard, which implies NP-hard (LaValle, 2006; Reif and Sun, 2001). Two additional, similar problems related to coverage path planning are the art gallery problem and the watchman route problem. The art gallery problem calls for the minimum number of guards needed to station in a polygonal gallery so that each point in the gallery is visible to at least one guard (Shermer, 1992). The watchman route problem calls for the shortest route from a given point back to itself so that each point in a given space is visible from at least one point along the route (Li and Klette, 2008). Simple cases of the watchman route problem such as covering the interior of simple polygons can be achieved in polynomial time (Chin and Ntafos, 1991). But, in general, both the art gallery problem and the watchman route problem are NP-hard (Shermer, 1992; Li and Klette, 2008). Some coverage algorithms we discuss below approach coverage path planning as the art gallery problem and the watchman route problem.

Coverage algorithms can be classified as heuristic or complete depending on whether or not they mathematically guarantee complete coverage of the free space. Independently, they can be classified as either off-line or on-line. This classification was originally proposed by Choset (2001). Off-line algorithms rely only on stationary information, and the environment is assumed to be known in advance. However, assuming full prior knowledge of the environment might be unrealistic in many scenarios. On the other hand, online algorithms do not assume full prior knowledge of the environment to be covered and utilize real-time sensor measurements to sweep the target space. Thus, these later algorithms are also called sensor-based coverage algorithms.

In certain scenarios, a valid approach to solve the problem is to randomize. This is an approach that some floor-cleaning robots rely on: if the floor is swept randomly for long enough, it should become cleaned. Examples of commercial floor-cleaning robots based fully or partially on this strategy are the RC3000 by Karcher, Trilobite by Electrolux and Roomba by iRobot (Palacin et al., 2005). These randomized strategies often combine randomization with path templates such as wall following. Figure 4 shows a randomized coverage path traced by a team of Roomba cleaning robots. There are advantages to this approach, the main one being that no complex sensors for localization nor expensive computational resources are needed. However, for covering vast areas, and especially for underwater or aerial robotics operations which deal intrinsically with a 3-dimensional space, it is difficult to think that a randomized "algorithm" could be usable, as the cost of operating the vehicle (energy and time) would be unaffordable.

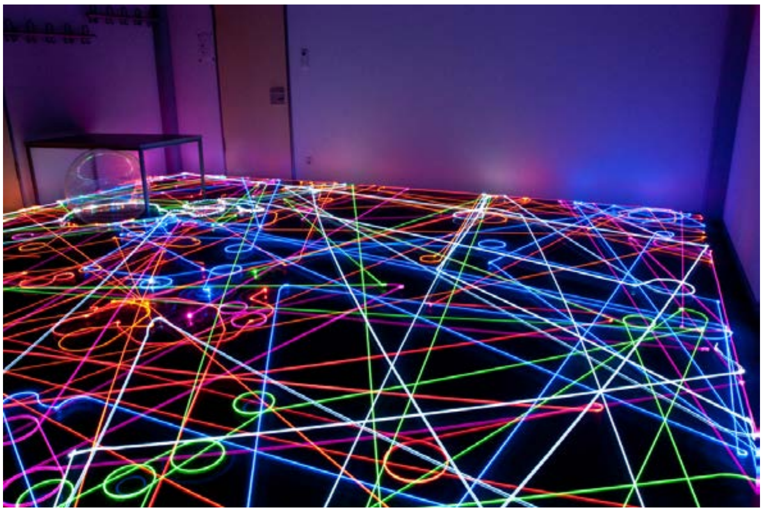


Figure 4: Long-exposure photo of the randomized coverage path traced by a team of Roomba cleaning robots using LED light sources in a dark room. Image credit: Tobias Baumgartner, Marcus Brandenburger, Tom Kamphans, Alexander Kroeller and Christiane Schmidt of the IBR Algorithm Group at Braunschweig University of Technology.

A considerable body of research addressing the coverage path planning problem can be found in the literature. Choset (2001) presented a survey on the field. However, no updated surveys on coverage path planning reflecting its recent advances have been presented in the past ten years. This survey bridges this gap by presenting a thorough review of the most successful coverage path planning methods, focusing in the achievements made in the past decade. Furthermore, we discuss reported field applications of the coverage path planning methods described.

Since most coverage path planning algorithms decompose the target space in sub-regions (called cells) to achieve coverage, Choset (2001) classified coverage algorithms according to the type of decomposition used. Hence, his taxonomy comprises heuristic and randomized approaces (which typically do not use a representation of the environment and therefore neither use a decomposition), and approximate, semi-approximate and exact cellular decompositions. However, we argue that qualitatively different approaches can be distinguished among these categories. Thus, the outline of this survey does not bear a one-to-one correspondence with Choset's taxonomy, but rather reflects the common underlying ideas used in the discussed approaches. Nonetheless, Choset's taxonomy is commonly used throughout the literature, and hence we also provide the corresponding Choset's classification for the methods reviewed.

2.2 CLASSICAL EXACT CELLULAR DECOMPOSITION METHODS

Exact cellular decomposition methods break the free space (i.e., the space free of obstacles) down into simple, non-overlapping regions called cells. The union of all the cells exactly fills the free space. These regions, which contain no obstacles, are "easy" to cover and can be swept by the robot using simple motions. For instance, each cell could be covered using a zigzag, "mowing the lawn" pattern like the one shown in Figure 5. Generation of these zigzag motions, also called seed-spreader motions, is well documented in the literature (Lumelsky et al., 1990; Choset and Pignon, 1997; Acar et al., 2002).

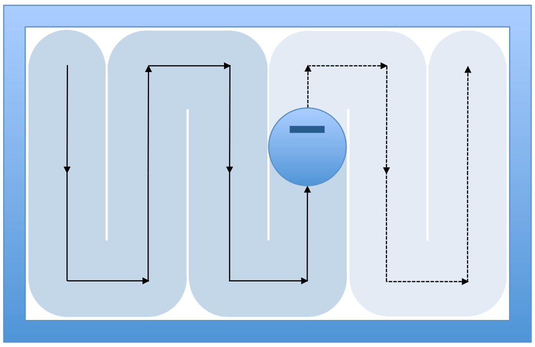


Figure 5: Typical "mowing the lawn" path. The shaded area indicates the area already covered (darker) and the area that will be covered (lighter) when the robot finishes following the path.

Two cells are said to be adjacent if they share a common boundary. An adjacency graph can be used to represent the cellular decomposition, where a node represents a cell and an edge represents an adjacency relationship between two cells (see Figure 6). Exact cell decompositions can be generated by sweeping a line through the space (e.g. from left to right). The cell boundaries are then formed when some event is encountered by the sweep line. For instance, a change on the number of times the sweep line intersects with obstacle boundaries can be used as an event.

Typically, a planner based on exact cellular decomposition generates a coverage path in two steps. First, it decomposes the free space into cells and stores the decomposition as an adjacency graph. Next, it computes an exhaustive walk through the adjacency graph (i.e., a sequence that visits each node in the graph exactly once). It is worth noting that the exhaustive walk obtained is a sequence of nodes (i.e., a sequence of cells), and not an actual coverage path. Therefore, an explicit path for covering each cell must be derived using simple motions as discussed above.

Next, two popular off-line cellular decomposition approaches that laid down the foundations for more advanced methods are discussed.

2.2.1 *Trapezoidal Decomposition*

One of the simplest exact cellular decomposition techniques which can yield a complete coverage path is the trapezoidal decomposition (Latombe, 1991; Choset et al., 2005), which handles only planar, polygonal spaces. Given that it uses no sensor information, this method can be classified as off-line. In the trapezoidal decomposition, each cell is a trapezoid, as shown in Figure 6. Thereby, simple back-and-forth motions can be used to cover each cell. Complete coverage is guaranteed by finding an exhaustive walk through the adjacency graph associated to the decomposition (overlaid on the decomposition in Figure 6). Finally, a specific zigzag path to cover each cell is generated. The exhaustive walk determines the order in which the cells are visited to achieve complete coverage. Finally, specific paths to cover each cell are generated, typically using back-and-forth motions in a "mowing the lawn" manner.

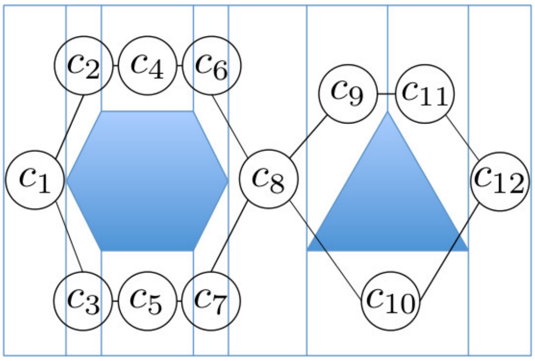


Figure 6: Trapezoidal decomposition of an example workspace with its corresponding adjacency graph.

As an application example, Oksanen and Visala (2009) proposed an offline algorithm based on the trapezoidal decomposition for coverage path planning in the case of agricultural fields. Their algorithm applies a trapezoidal decomposition of the field followed by a cell merging procedure. The resulting cells are similar to those produced by the boustrophedon decomposition, introduced in the next section (2.2.2). To optimize the path, they use a path-based cost function to assess the largest cell arising in six different trapezoidal decompositions obtained by using a sweep line inclined at 30° intervals. Then, the three most favorable directions are selected and the process is repeated, with additional decompositions at 15° either side of the selected headings. The process continues iteratively until the improvement per step falls below a threshold, which for their application was achieved after 5 steps (about 1° accuracy), requiring 36 separate decompositions. Then, the largest cell in the minimum-cost decomposition is removed from the target area, and the process is repeated for the remainder of the field until all the area is covered by the path. This scheme produces effectively optimal coverage paths for a convex field and high-quality paths for a field whose boundaries consist of long, straight segments as well.

2.2.2 *Boustrophedon Decomposition*

A drawback of the trapezoidal decomposition is that it generates many cells that, intuitively, can be merged together to form bigger cells. This is clearly inconvenient, as the more cells are present, the longer the final coverage path becomes, as shown in Figure 7. This happens because the trapezoidal decomposition creates only convex cells. However, some non-convex cells can also be completely covered by simple motions. To overcome this limitation, Choset & Pignon proposed the boustrophedon cellular decomposition (Choset and Pignon, 1997; Choset et al., 2000a). The word "boustrophedon" comes from ancient Greek, and literally means "the way of the ox", signifying the pattern in which an ox drags a plow back and forth. The boustrophedon decomposition is similar to the trapezoidal decomposition introduced above, but it only considers vertices in the environment where a vertical segment can be extended both above and below the vertex. The vertices where this occurs are called critical points.

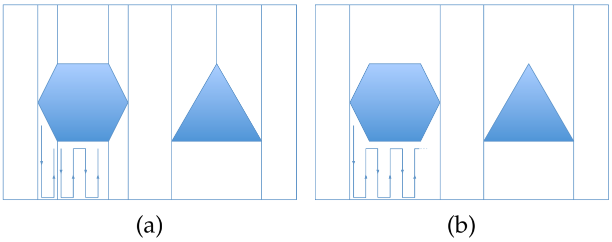

(a) (b)

Figure 7: A decomposition with less cells allows for shorter coverage paths. Note how an extra strip is needed in the trapezoidal decomposition in (a) with respect to the boustrophedon decomposition in (b).

By adhering to this strategy, the boustrophedon decomposition effectively reduces the number of cells in trapezoidal decomposition. Hence, shorter coverage paths are obtained. Notice that, as the trapezoidal decomposition, this method assumes polygonal obstacles and the terrain to be known *a priori*, and thus classifies as an off-line method.

2.3 MORSE-BASED CELLULAR DECOMPOSITION

Later, Acar et al. (2002) generalized the boustrophedon decomposition by proposing a novel cellular decomposition approach based on critical points of Morse functions (Milnor, 1963). In fact, they show that the boustrophedon decomposition is a particular case of Morse decomposition. With respect to the original boustrophedon decomposition, the Morse-based decomposition has the advantage of handling also non-polygonal obstacles. By choosing different Morse functions, different cell shapes are obtained, e.g. circular or spiked cells. Theoretically, Morse decompositions can be applied to any n-dimensional space. Moreover, they presented a method to perform coverage of planar spaces by detecting the critical points using sensory range information, and a motion-template-based algorithm that ensures to encounter all the critical points in the target area. Therefore, this method allows complete coverage on-line (Acar and Choset, 2001, 2002a).

The Morse decomposition is based on a roadmap method for start-to-goal path planning proposed by Canny (Canny, 1988, 1993). Critical points of a Morse function restricted to the obstacle boundaries are used to determine the cell decomposition. Recall that, given a real-valued function $h\colon \mathbb{R}^m \to \mathbb{R}$, its differential at $p \in \mathbb{R}^m$ is $dh(p) = [\frac{\partial h}{\partial x_1}(p) \ldots \frac{\partial h}{\partial x_m}(p)]$. A critical point is a value $p \in \mathbb{R}^m$ where either the function is not differentiable or all its partial derivatives are 0, i.e., $\frac{\partial h}{\partial x_1}(p) = \ldots = \frac{\partial h}{\partial x_m}(p) = 0$, and its Hessian $(\frac{\partial^2 h}{\partial x_i \partial x_j}(p))$ is non-singular. For instance, in the case of a single variable function, a critical point corresponds either to a local maximum, a local minimum or an inflection. A Morse function is one whose critical points are nondegenerate (Milnor, 1963). Practically speaking, this means that critical points are isolated from one another.

To determine the cell decomposition, a slice is swept through the target space. Formally, the slice is a codimension one manifold defined in terms of the preimage of a real-valued Morse function, $h\colon \mathcal{W} \to \mathbb{R}$, where $\mathcal{W}$ is the robot's workspace, i.e., the space to be covered. For instance, in the plane ($\mathcal{W} = \mathbb{R}^2$), choosing $h(x, y) = x$ will make the slice be effectively a vertical line. Changes on the connectivity of the slice occur at critical points of the Morse function restricted to the obstacle boundaries. To put it more simply, at a critical point the sweep line encounters an obstacle whose surface normal is perpendicular to the sweep line, as shown in Figure 8. Morse theory guarantees that, between critical points, the connectivity of the slice remains unchanged. Thus, as no obstacles are located between critical points, the space between them can be covered easily by simple motions, and critical points can be used to determine the cell boundaries.

Choosing different Morse functions produces different slice shapes and hence different cell decomposition patterns. For simplicity, we will describe the Morse-based boustrophedon decomposition (Choset, 2000), which happens in the plane. Later, we will give examples of different decomposition patterns obtained using different Morse functions.

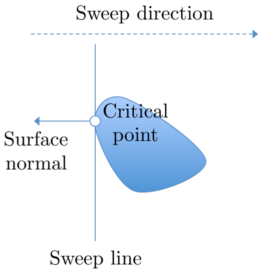
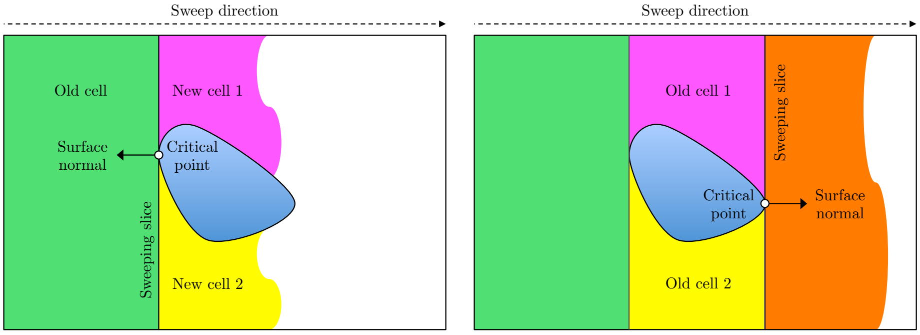


Figure 8: Cell boundaries in Morse decomposition are placed at critical points, where the surface normal of the obstacle is perpendicular to the sweep slice, and parallel to the sweep direction.

In the boustrophedon decomposition, a vertical slice, defined in terms of the Morse function $h(x, y) = x$, is swept from left to right in the workspace, i.e., along the abscissae axis. Thus, the vertical slice is determined by the preimage of this Morse function, $\mathcal{W}_\lambda = h^{-1}(\lambda)$. The slice is parametrized by $\lambda \in \mathbb{R}$, which fixes its location in the target space. Increasing the value of the slice parameter, λ, sweeps the slice from left to right through the workspace.

As the slice sweeps the space, it intersects (or stops intersecting) obstacles, which divide it into smaller pieces as the slice first encounters an obstacle, that is, the connectivity of the slice in the free space increases. Also, immediately after the slice leaves an obstacle, smaller slice pieces are merged into larger pieces (the connectivity of the slice in the free space decreases). The points where these connectivity changes occur are the critical points. (Recall that critical points are always located on the obstacle boundaries.) Thus, at critical points, the slice is used to determine the cells in the decomposition. Notice that within a cell, the slice connectivity remains constant. Figure 9a shows how, at the critical point, the connectivity of the slice changes from one to two, and hence the old cell is closed and two new cells are created. In Figure 9b, at the critical point, the connectivity of the slice changes from two to one, and hence two old cells are closed and a new cell is created.

Once the cell decomposition is constructed, an exhaustive walk through its associated adjacency graph is determined by the planner. Then, it generates the explicit coverage path in each cell. The coverage pattern within each cell has three parts: motion along a slice, motion orthogonal to the slice, and motion along the cell boundary, as shown in Figure 10. First, the robot laps along the current slice, $\mathcal{W}_\lambda$. Then the robot steps outward of the slice by going orthogonally to it by an inter-lap distance, typically by a distance of one robot sensor range; λ is also increased by this distance to form a new slice. If the robot encounters an obstacle (i.e., the cell boundary) while moving along the slice, the planner directs the robot to follow the obstacle



(a)

(b)

Figure 9: Cell determination with the Morse-based boustrophedon cell decomposition method.

boundary until it has moved an inter-lap distance and then a new lap along a new slice is started. The process repeats until the cell is completely covered.

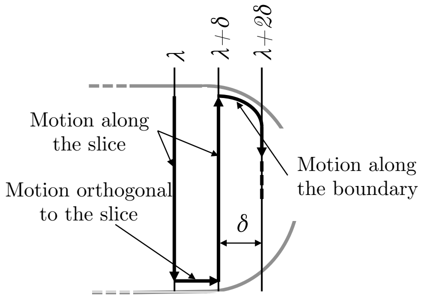


Figure 10: Boustrophedon path construction process, where δ is the inter-lap spacing and λ is the slice parameter

Figure 11 shows the Morse-based boustrophedon cell decomposition of an example workspace with its associated adjacency graph. A key point of Morse decompositions is that, by choosing different Morse functions to define the slice that is swept through the space, different decomposition and coverage path patterns can be generated, such as the spiral pattern (Acar et al., 2002). Figure 12 shows a spiral pattern obtained using the morse function $h(x, y) = \sqrt{x^2 + y^2}$. Allowing different coverage patterns is useful for vehicles with kinematic constraints. For instance, a spiral path can be easily followed by an underactuated car-like vehicle unable to make sharp turns (Bosse et al., 2007).

A limitation of the Morse decomposition method is that it cannot handle rectilinear environments. This is because it is not possible to determine critical points in those environments which correspond to a change in the topology of the space (the critical points are said to be degenerate in this case (Milnor, 1963)).

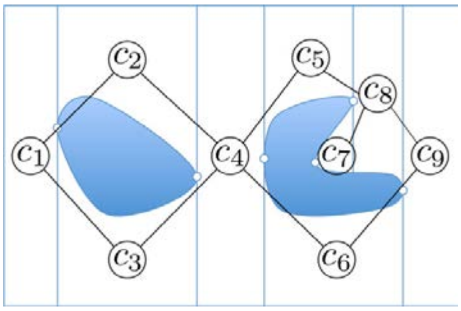


Figure 11: Morse decomposition of an example workspace with its associated adjacency graph.

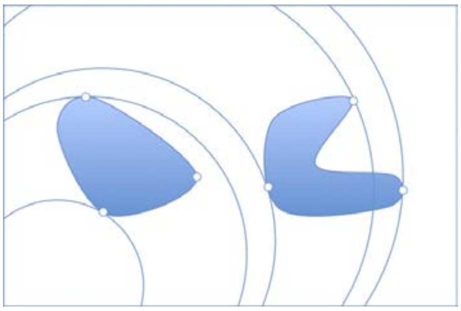


Figure 12: Spiral Morse decomposition obtained using the Morse function $h(x, y) = \sqrt{x^2 + y^2}$.

2.3.1 *On-line Morse-based Boustrophedon Decomposition*

To face the sensor-based coverage problem, Acar and Choset (2002a, 2000) gave a method to detect the critical points of a Morse-based boustrophedon decomposition on-line using range sensor information. Furthermore, they presented an algorithm that ensures to encounter all the critical points while performing coverage. To detect the critical points, they use an omnidirectional range sensor to look for points where the surface normals $\nabla m(x)$ of obstacles and the sweep direction are parallel. Given a robot located at point x, let c_0 be the closest point to x on the surface of obstacle C_i:

$$c_0 = \underset{c \in C_i}{\operatorname{argmin}} \|x - c\|, \tag{1}$$

and let $d_i(x)$ be the distance between point x and the obstacle C_i. Now, the gradient of $d_i(x)$, $\nabla d_i(x)$ can be calculated as

$$\nabla d_i(x) = \frac{x - c_0}{\|x - c_0\|}. \tag{2}$$

Recall that, by definition, a gradient is a unit vector normal to a surface at a given point. In Equation 2, as c_0 is a point on the surface of the obstacle C_i, $x - c_0$ is a vector pointing outward from c_0 towards x. Given that c_0 is the closest point to x on the obstacle surface, the vector $x - c_0$ is hence normal to the surface of the obstacle. By dividing by its norm, $\|x - c_0\|$, the result is turned into a unit vector.

A detection of a critical point occurs when $\nabla d_i(x)$ is parallel to the sweep direction. Or, in other words, when the sweep direction and the obstacle surface normal are parallel. Figure 13 illustrates this situation.

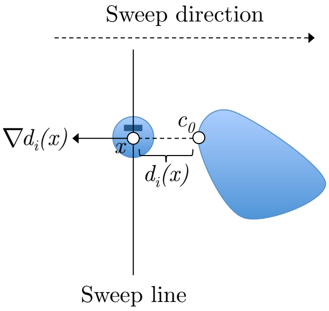


Figure 13: Critical point detection occurs on the side of the range sensing robot, whose heading is indicated by the rectangular mark on the circle representing the robot.

Notice the fact that critical points can only be detected when they are the closest point on the obstacle surface from the robot compared to all other points on the obstacle surface. This implies that they can only be detected when the robot is performing wall following. Therefore, using a simple zigzag can result in some critical points being missed, as those shown in Figure 14.

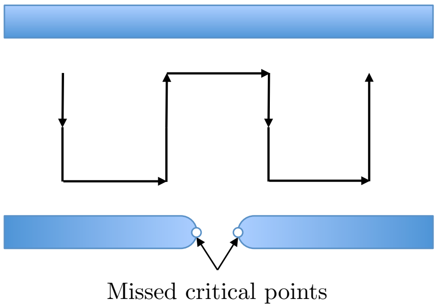


Figure 14: With Morse decomposition, a range sensing robot following a simple zigzag path will miss the critical points in the figure unless it performs wall following both on the top and the bottom of a cell.

To address this issue, (Acar et al., 2002) presented an algorithm that uses repeated rectangular motion cycles with wall following on both ends of a lap, as shown in Figure 15. The algorithm is termed "cycle algorithm". Notice that this cyclic path includes retracing, and hence is longer than the simple zigzag path.

The cycle path generation process of their proposed algorithm is shown in Figure 16. Initially (Figure 16a), the robot starts a forward phase at point S_i and moves downward. When it encounters an obstacle, it performs wall following until it has reached the next strip or until a critical point is detected. If a critical point is detected, the robot then enters in a reverse phase where it

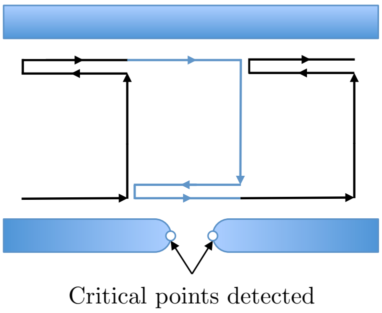
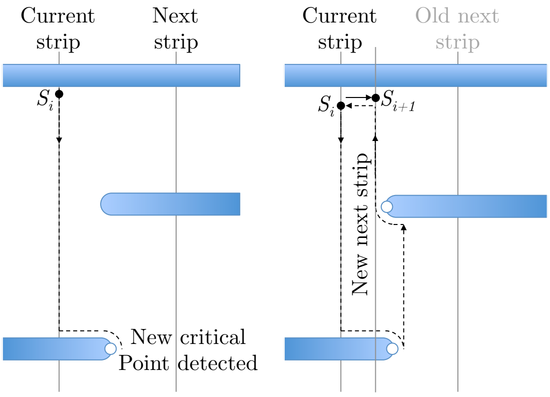


Figure 15: A path composed of rectangular cycles allows detection of all the critical points. This pattern is used in on-line Morse decomposition and the CC_R algorithm as well.

moves upward (see Figure 16b). In this phase, if an obstacle is detected, the robot wall-follows it. If a critical point is detected on the obstacle boundary, the next strip is moved to where the critical point is. The motion continues until S_i is reached again on the current strip. Then, the robot starts a new cycle at point S_{i+1}. The algorithm is proven to detect all the critical points.



(a) Forward phase (b) Reverse phase

Figure 16: Critical point detection using the cycle algorithm.

In order to store and incrementally construct the Morse decomposition online, it is stored as a Reeb graph (Reeb, 1946). The Reeb graph is dual to the adjacency graph in that the nodes of the Reeb graph are the critical points and the edges connect neighboring critical points, i.e., correspond to cells. An example Morse decomposition with its associated Reeb graph is shown in Figure 17.

Whenever a critical point is encountered, the robot updates the Reeb graph. When a cycle path where critical points were found is finished, the robot looks for uncovered cells at the last encountered critical point. If the critical point is associated with two uncovered cells, the robot picks one of the cells associated as the next cell to cover. If there are no uncovered cells associated with the last encountered critical point, a depth-first search is performed on the Reeb graph. To travel to the selected uncovered cell, the robot follows the Reeb graph and plans a path that passes through cells and critical points.

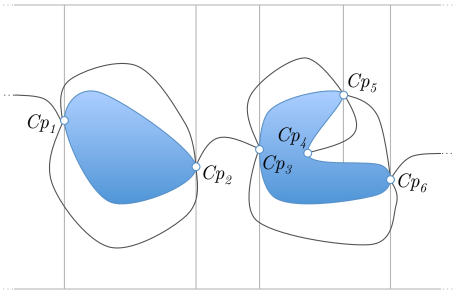


Figure 17: Morse decomposition of an example workspace with its associated Reeb graph. $Cp_1 \ldots Cp_6$ are the critical points.

When no uncovered cells (edges) remain in the Reeb graph, the environment is completely covered.

The cycle algorithm just described, however, may fail to detect critical points on certain non-convex obstacles. In particular, concave critical points such as Cp_2 in Figure 18 cannot be detected by range data when the boundary's curvature is smaller than the robot's periphery (Garcia and de Santos, 2004). However, the closest convex critical point (Cp_3 in the example shown in Figure 18) to a critical point like Cp_2 in Figure 18 will indeed be detected. This leads to adding a spurious edge to the Reeb graph that does not correspond to any existing cell. As a result, the algorithm will fail to detect the closing critical point for the newly added edge. Garcia and de Santos (2004) proposed a solution to this issue which involves associating each critical point with its obstacle and defining unique entry and exit critical points for each obstacle. Additionally, their paper discusses several implementation details of the cycle algorithm.

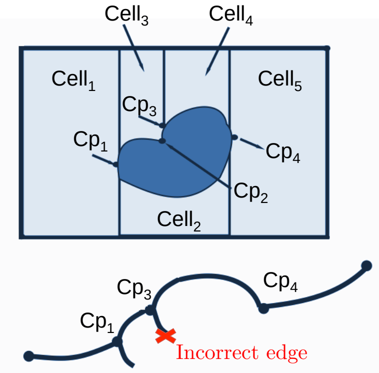


Figure 18: A concave critical point will not be detected if the boundary's curvature is smaller than the robot's periphery and will lead to an incoherent Reeb graph. This is the case of Cp_2 in this example environment, which does not get detected and produces an incorrect edge emanating from Cp_3.

Acar et al. (2003) discussed coverage path planning in relation to a demining application. In their article, they show that Morse decomposition overcomes the randomized coverage approach in this task, which used to be considered the state of the art on demining operations.

2.3.2 *Morse-based Cellular Decomposition Combined with the Generalized Voronoi Diagram*

Acar et al. (2001, 2006) presented a sensor-based coverage approach with extended range sensors. As they point out, "prior work in coverage tends to fall into one of two extremes: coverage with an effector the same size of the robot, and coverage with an effector that has infinite range." In this work, they consider coverage in the middle of this spectrum: coverage with a detector range that goes beyond the robot, and yet is still finite in range. They term these sensors extended range sensors. In this work, coverage is achieved in two steps: the first step considers vast, open spaces, where the robot can use the full range of its detector; the robot covers these spaces as if it were as big as its detector range (see Figure 19, on the right). Here previous work in using Morse cell decompositions (Acar et al., 2002) is employed to cover unknown spaces. As explained above, cells in this decomposition can be covered via simple back-and-forth motions, and coverage of the vast space is then reduced to ensuring that the robot visits each cell in the vast space. The second step considers the narrow or cluttered spaces where obstacles lie within detector range, and thus the detector "fills" the surrounding area. In this case, the robot can cover the narrow space by simply following the Generalized Voronoi Diagram (GVD) of that space, which are sets of points equidistant to two obstacles (Choset and Burdick, 2000; de Berg et al., 2008) (see Figure 19, on the left). The GVD can be constructed on-line using range sensor information and has been previously used for sensor-based exploration (Choset et al., 2000b) and inspection of 3D structures (Choset and Kortenkamp, 1999). A hierarchical decomposition that combines the Morse decompositions and the GVDs is introduced to ensure that the robot indeed visits all vast, open, as well as narrow, cluttered, spaces. In their article, it is shown how to construct this decomposition on-line using sensor data accumulated while the robot covers the environment.

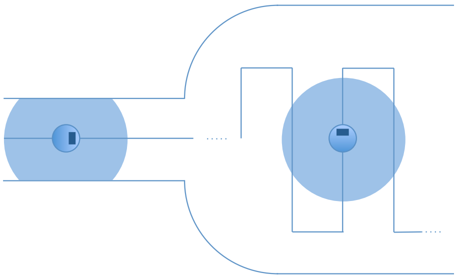


Figure 19: Combination of Morse decomposition and GVD for extended range sensor coverage. In cluttered spaces (left) the robot just follows the GVD of that space. In vast areas (right), the robot follows the pattern generated using a Morse decomposition scheme.

2.4 LANDMARK-BASED TOPOLOGICAL COVERAGE

Wong and MacDonald (2003) presented an on-line topological coverage algorithm for mobile robots based on detection of natural landmarks. This work is intended for simple planar environments. As in Morse decomposition, their method also uses concepts introduced by the boustrophedon decomposition. However, the topological algorithm proposed here uses different events to determine cell boundaries. Morse decomposition places cell boundaries on the critical points on the obstacles surface. However, as commented before, rectilinear environments cannot be handled by Morse decomposition, as the critical points in such environments are degenerate. On the other hand, as critical points can only be discovered on the side of the robot while performing wall following, a rectangular coverage pattern which includes retracing is needed. By contrast, the topological approach presented here uses simpler landmarks to determine an exact cellular decomposition termed "slice decomposition". Due to the use of simpler landmarks, slice decomposition can handle a larger variety of environments, including those with polygonal, elliptical and rectilinear obstacles. Moreover, obstacles can be detected from all sides of the robot, allowing a simpler zigzag pattern without retracing to be used. As a result, the generated coverage path is shorter with this method.

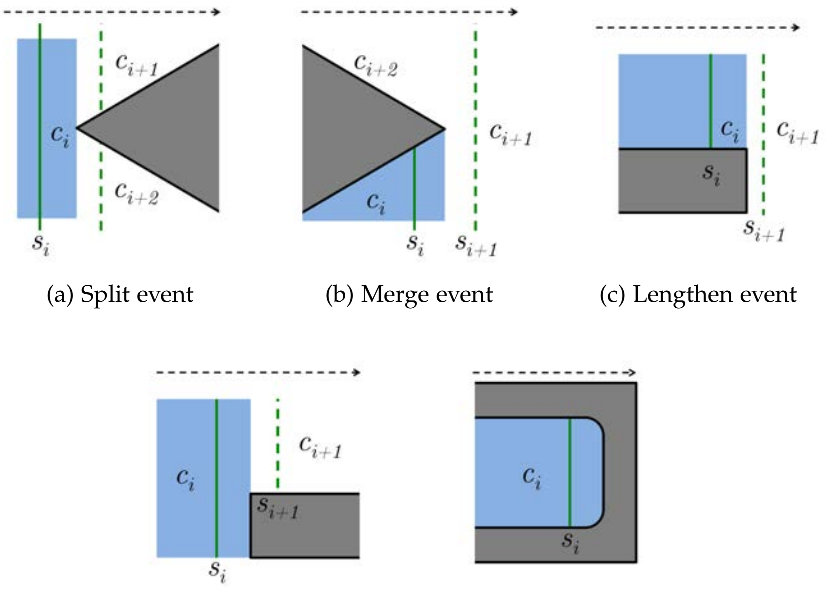
2.4.1 *Slice Decomposition*

The slice decomposition is constructed by sweeping a line through the space. It uses five different events to determine the cell boundaries:

1. *Split event*: A free space segment in the previous strip is split into two by the emergence of an obstacle, as in Figure 20a.
2. *Merge*: Two free space segments in the previous strip are merged into one by the disappearance of an obstacle, as in Figure 20b.
3. *Lengthen*: The current strip is much longer than the previous strip, as in Figure 20c.
4. *Shorten*: The current strip is much shorter than the previous strip, as in Figure 20d.
5. *End*: The previous free space segment is the final one in the current cell, as in Figure 20e.

All these events or landmarks can be detected using a combination of range measurements thresholding and temporal sequence comparisons (comparing current sensor reading with previous ones) and odometry (comparing length of consecutive strips). An alternative solution that uses a neural network to detect the events is also presented in this work (Wong, 2006).

Whenever one of the stated events occur, a cell boundary is placed along the sweep line where the event takes place. The slice decomposition can be



(a) Split event

(b) Merge event

(c) Lengthen event

(d) Shorten event

(e) End event

Figure 20: Events (landmarks) in the slice decomposition. In all the events shown c_i is the current cell (shaded) and s_i is the current strip. The dashed arrow indicates the sweep direction.

encoded as a topological map. A topological map is represented as a planar graph, where the nodes represent landmarks (i.e., split, merge, end, lengthen or shorten events) and edges indicate the types of motion required to travel between nodes they are incident upon. For example, whether the edge is next to a wall and which side the wall is on. They also store estimated distances separating the two nodes they connect.

2.4.2 *On-line Topological Coverage Algorithm*

An algorithm that constructs the slice decomposition on-line while performing coverage is presented in Wong et al.'s work. The algorithm guarantees complete coverage. It iteratively constructs the topological map associated to the slice decomposition of the environment using a finite state machine with three states: boundary, normal, and travel. Figure 21 shows its state transition diagram. The algorithm starts in the boundary state, as it is assumed that the robot is initially located in a corner of the environment. This assumption is not a shortcoming as it is easy to program a robot to look for a corner by following simple forward and wall following motions. In the boundary state, the robot explores the current cell boundary. The aim of the boundary exploration is to expose all cells neighboring the current border. Whenever the robot arrives at a landmark or at an end of the cell boundary, the topological map is updated. When the boundary exploration has finished, the algorithm switches to the travel state. In the travel state, the robot searches the topological map for an uncovered cell and it is directed to that cell. Then, the robot enters the normal state, where it follows a zigzag pattern to cover the current cell. Again, whenever a landmark is found the topological map is updated and the algorithm switches to the boundary state. The algorithm finishes when there are no more uncovered cells in the topological map.

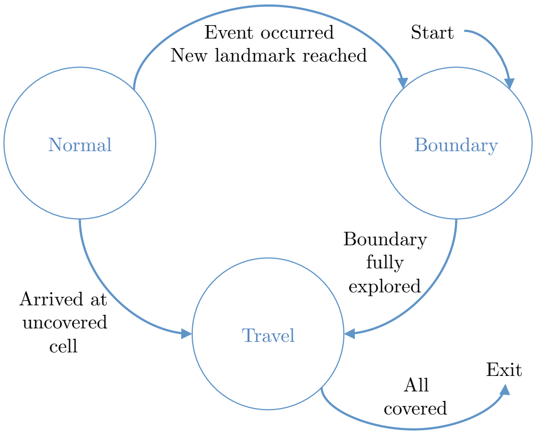


Figure 21: State transition diagram of the topological coverage algorithm.

2.5 contact sensor-based coverage of rectilinear environments

Butler et al. (1999) proposed CC_R, an exact cell decomposition algorithm for contact sensing robots (i.e., robots without range sensing capabilities) covering unknown rectilinear environments on-line. Their motivating application for coverage of rectilinear environments is calibration of an automated assembly system in which planar linear motors operate on table-like surfaces to transfer products through a factory.

In CC_R (for contact-based coverage of rectilinear environments), the robot follows a cyclic path with retracing as shown in Figure 15. At the same time, it iteratively constructs a cellular decomposition of the environment. An example rectilinear decomposition produced by CC_R is shown in Figure 22. In fact, the decomposition constructed by CC_R can be seen as the case of Morse decomposition where all the critical points are degenerate, as this is the case in rectilinear environments.

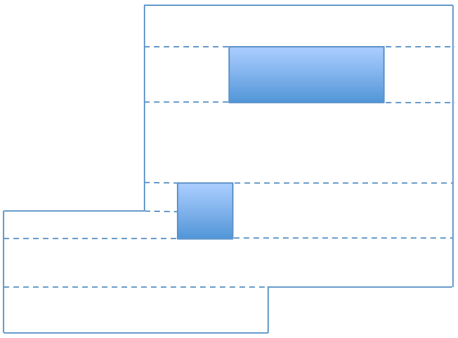


Figure 22: CC_R uses an exact cell decomposition for rectilinear environments.

Normally, CC_R follows the cyclic path. An event (and hence a cell boundary) occurs whenever the robot is prevented from successfully executing a full path cycle. When such an event occurs, CC_R chooses a new trajectory based only on the robot's environment and its current position. The next trajectory is determined by a list of rules that are designed to continue coverage in all possible cases.

A proof of completeness for CC_R is given by creating a finite state machine that describes all possible events encountered by the robot, and demonstrating that the finite state machine has no infinite loops and that it stops only when coverage is complete.

2.6 GRID-BASED METHODS

Grid-based methods use a representation of the environment decomposed into a collection of uniform grid cells. This grid representation was first proposed by Moravec and Elfes (1985) to map an indoor environment using a sonar ring mounted on a mobile robot. In this representation, each grid cell has an associated value stating whether an obstacle is present or if it is rather free space. The value can be either binary or a probability (Elfes, 1987). Typically, each grid cell is a square, but also different grid cell shapes can be used, such as triangles. As grid representations only approximate the shape of the target region and its obstacles, Choset classified grid-based methods as approximate cellular decompositions (Choset, 2001). As a result of this approximate representation, most grid-based methods are "resolution-complete", that is, their completeness depends on the resolution of the grid map. Figure 23 shows an example grid map.

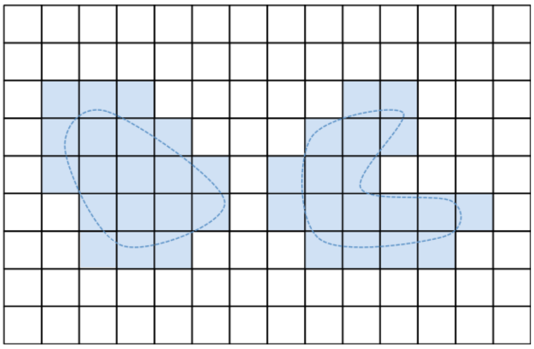


Figure 23: An example grid map. Grid cells with obstacles present are shaded.

It is easy to create a grid map, as it can be represented as an array where each element contains occupancy information associated with a cell. On the other hand, it is simple to mark covered areas in a grid map. As a result, grid-based representations are the most widely used for coverage algorithms. Nonetheless, grid maps suffer from exponential growth of memory usage because the resolution remains constant regardless of the complexity of the environment (Thrun, 1998). Also, they require accurate localization to maintain the map's coherency (Castellanos et al., 1997; Thrun, 2003). For these reasons, grid-based coverage methods are suited for indoor mobile robot operations, as the size of the area to be covered is typically relatively small.

Usually, cells in a grid map are square in shape and robot-sized. Oh et al. (2004) proposed a coverage algorithm that uses a grid map in which the cells are triangles instead. The rationale behind the choice of triangular cells is that they offer a higher resolution in comparison to rectangular cells of similar size. However, the resolution of the grid can also be augmented by using finer-grained squared cells. In mobile robotics, field for which the mentioned algorithm is intended, most mobile robots are not capable of making very fine movement adjustments, and hence there is no need for ultra high resolution in coverage path planning. Therefore, the extra effort devoted to implementing a triangular grid seems not to be worthwhile.

2.6.1 *Grid-based Coverage using the Wavefront Algorithm*

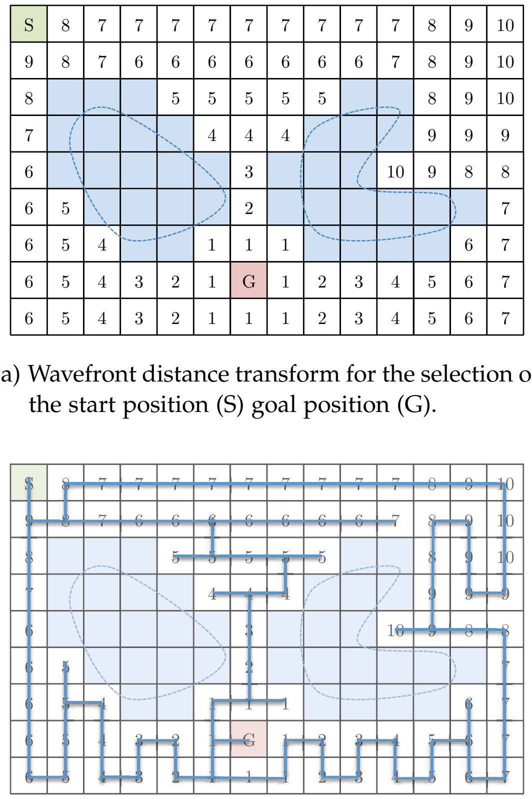
Zelinsky et al. (1993) presented the first grid-based method for coverage path planning. In their off-line method, they use a grid representation and apply a complete coverage path planning algorithm to the grid. The method requires a start cell and a goal cell. A distance transform that propagates a wave front from goal to start is used to assign a specific number to each grid element. That is, the algorithm first assigns a 0 to the goal and then a 1 to all its surrounding cells. Then, all the unmarked cells neighboring the marked with a 1 are labeled with a 2. The process repeats incrementally until the start cell is reached by the wavefront. Figure 24a illustrates this procedure on an example environment.

Once the distance transform is calculated, a coverage path can be found by starting on the start cell and selecting the neighboring cell with the highest label that is unvisited. If two or more unvisited neighbors share the same label, one of them is selected randomly. This process to find a coverage path is equivalent to using pseudo-gradient descent from the start point on the numeric potential function constituted by the labeling, that is, following the equipotential curves from top to bottom. Figure 24b shows the generated coverage path for the example environment on Figure 24a. A unique feature of this coverage algorithm is that a start and a goal point can be specified.

Shivashankar et al. (2011) introduced a generalization of the wavefront algorithm to unknown environments to achieve on-line coverage with a mobile robot.

2.6.2 *Grid-based Coverage using Spanning Trees*

Gabriely and Rimon (2002) proposed the Spiral-Spanning Tree Coverage (STC) algorithm, an on-line approach for mobile robots which consists in subdividing the workspace into a grid map and following a systematic spiral path. This systematic spiral path is generated by following a spanning tree of the partial grid map that the robot incrementally constructs using its onboard sensors. The robot is able to cover every grid cell precisely once,



S	8	7	7	7	7	7	7	7	7	7	8	9	10
9	8	7	6	6	6	6	6	6	6	7	8	9	10
8				5	5	5	5	5			8	9	10
7					4	4	4				9	9	9
6						3				10	9	8	8
6	5					2							7
6	5	4			1	1	1					6	7
6	5	4	3	2	1	G	1	2	3	4	5	6	7
6	5	4	3	2	1	1	1	2	3	4	5	6	7

(a) Wavefront distance transform for the selection of the start position (S) goal position (G).

(b) Coverage path generated using the wavefront distance transform.

Figure 24: Coverage path planning using the wavefront algorithm for an example environment.

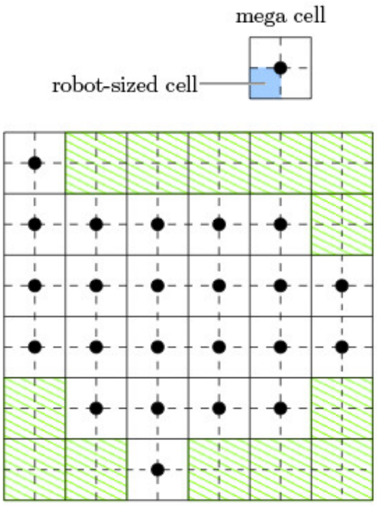
and travel a complete coverage path. They validate the proposal in simulation. The Spiral-STC algorithm works as follows. Two different grid cell sizes are used. Bigger cells (so called mega cells) are divided in four smaller cells, which are the same size as the robot. This is shown in Figure 25a. To perform coverage, the robot executes the following procedure. Starting at the current cell, the robot chooses a new travel direction by selecting the first new mega cell in the free space in anti-clockwise direction. Then, a new spanning-tree edge is grown from the current mega cell to the new one. The algorithm is called recursively. The recursion stops only when the current cell has no new neighbors (a mega cell is considered old if at least one of its four smaller cells is covered, it is considered new otherwise). As a result of this recursion, the robot moves along one side of the spanning tree until it reaches the end of the tree. At that point, the robot turns around to traverse the other side of the tree. It is worth noticing that, when coverage is completed, the robot returns to the start cell, facilitating its collection and storage. On the other hand, STC never visits any small cell twice and thus minimizes the coverage time. Figure 25b shows an example of a coverage path generated by the Spiral-STC algorithm.

An extension to the Spiral-STC is the Backtracking Spiral Algorithm (BSA) by Gonzalez et al. (2005), which is also an on-line approach intended for mobile robots. As an advantage in regard to the Spiral-STC algorithm, they proposed an extension to cover not only unoccupied cells, but also the partially occupied ones. This extension is based on the idea that the partially-occupied cells are part of the external ring of the spiral path. These cells are covered by a wall-following procedure. The proposed extension can be applied to most grid-based coverage algorithms. Simulation experiments validate the proposed algorithm.

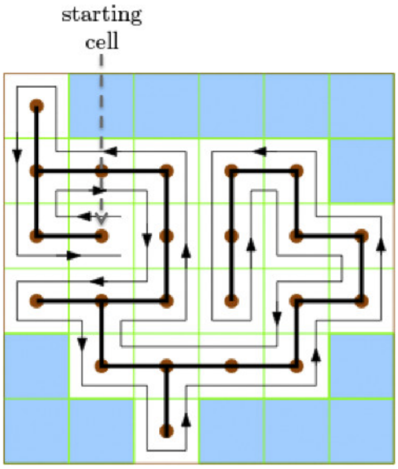
Choi et al. (2009) proposed an on-line complete coverage path planning solution based on the ideas introduced by the Spiral-STC algorithm and the BSA algorithm. They also use systematic spiral paths to achieve coverage, based on active wall finding. Nonetheless, they introduce a map coordinate assignment scheme based on the history of sensor readings to improve the time-to-completion by reducing the number of turns on the generated path. The generated spiral paths are then linked by an inverse distance transform they introduce. This proposal is validated in simulation and also with real-world experiments conducted inside a room with a mobile robot.

2.6.3 *Neural Network-based Coverage on Grid Maps*

Luo et al. (2002) and, in a latter work, Yang and Luo (2004) propose to use a neural network to achieve coverage path planning on-line targeted to a floor cleaning application. They discretize a 2D space in a grid map where the diagonal length of each grid cell is equal to the robot sweeping radius, and then a neuron is associated to each and every grid cell. Each neuron has connections to its immediate 8 neighbors. These concepts are shown in Figure 26.



(a) Approximate cell decomposition in mega cells and robot-sized cells.



(b) Coverage path generated with the Spiral-STC algorithm.

Figure 25: Coverage path planning using the Spiral-STC algorithm. Image credit: Wong (2006).

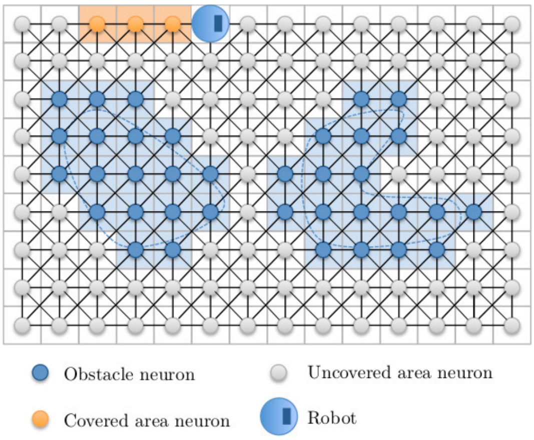


Figure 26: Schematic of the neural network used by Luo, Yang and others to achieve coverage.

A shunting equation based on the membrane equation by Hodgkin and Huxley (1952) determines the dynamics of each neuron in the network. The activity landscape (i.e., the output value of all neurons at a given instant) of the shunting model used attracts the robot to unclean areas, while the robot is repulsed by already cleaned areas and obstacles.

The next position of the robot is determined by the current position of the robot and the activity of the neuron associated to its current position, without any prior knowledge about the environment. It is assumed that the current state of the robot (if it is in a clean or dirty area, or in front of an obstacle, and its location) can be determined via sensory information. The state of the robot is an input to the neural network. The model used has six parameters that can be tuned in a wide range of values at the neural network design phase, and hence coverage is achieved without any learning procedures. An advantage of this method is that it can handle non-stationary environments (i.e., dynamically changing obstacles). The proposed neural network approach is validated in simulation. In (Luo and Yang, 2008) further simulation results are presented as well as a method to perform mapping on-line simultaneously with coverage navigation. In this later work, they consider a typical grid-based map and also a triangular mesh representation of the space, such as the one used by Oh et al. (2004).

An application of this neural network-based method to an AUV covering a 2D workspace in the seabed is proposed in (Yan and Zhu, 2011). The proposal is validated in a simplistic simulation environment. Nonetheless, this approach suffers from scalability issues since discretization in a grid map of a vast environment such as the seabed presents a tough challenge in terms of computational burden.

Qiu et al. (2006) added a local path planning technique on top of the biologically inspired neural network approach discussed above. In their approach, the next robot position is not determined immediately but rather a local path planning occurs in a window comprising a determined vicinity of the robot. By using this technique, they reduce the computational burden in comparison with the neural-network-only approach.

In a similar approach, Guo and Balakrishnan (2006) present a method based on a neural network to generate continuous steering control for a robot to completely cover a bounded region over a finite time. First, they discretize the space in a regularly spaced, disk-shaped grid. Then, a neural network based on the same biologically inspired shunting equation as in the works discussed above is used to provide continuous steering to the robot. The algorithm works for car-like robots which have non-holonomic motion constraints. The approach is validated in simulation.

2.6.4 *Hexagonal Grid Decomposition for Robots Equipped with Side-looking Sensors*

Paull et al. (2010, 2012) presented an on-line coverage method for robots equipped with side-looking sensors. Their target application is MCM operations using an AUV equipped with a side-scan sonar. This coverage method continuously directs the vehicle's heading using multi-objective optimization to maximize the information gain produced by the sensor measurements. The optimization procedure uses a grid decomposition composed of uniform hexagonal cells. The advantage of using a hexagonal grid is twofold. On one hand, distance does not need to be taken into account in the objective functions, because the distance from a given cell to its neighboring cells is the same. On the other hand, assuming the hexagons are small enough, visiting one cell guarantees coverage of two neighboring cells by the side-looking sensor, minimizing the amount of partially covered cells. Although the proposed method is able to cover target areas with non-convex shapes, obstacles present amidst the workspace are not considered. The efficacy of this method is demonstrated in simulation and experimentally on an AUV conducting MCM operations.

2.7 GRAPH-BASED COVERAGE

Xu (2011) presented coverage algorithms for environments that can be represented as a graph, such as a street or road network. In particular, this work addresses the following issues in the coverage problem. First, it takes into account that the prior map information provided as a graph might be incomplete. Second, it accounts for environmental constraints, such as restrictions in certain directions in the graph (corresponding to a one-way street, for example). Third, it provides strategies for on-line re-planning when changes in the graph are detected by the robot's sensors when performing coverage. Finally, strategies for coverage using multiple robots are provided.

Graph search algorithms are proposed to solve the coverage problems considered. Optimality is addressed by assigning a cost to each edge in the graph and either looking for the optimal solution when deliberation time allows or rather quickly finding an approximated solution when time constraints apply.

2.8 3D COVERAGE

Most coverage path planning methods, and in particular the methods reviewed so far in this survey, assume that the environment can be modeled as a simple planar surface. This assumption is valid for floor cleaning, land mine detection, lawn mowing, etc. However, some surfaces in nature are 3-dimensional, and 3-dimensional coverage path planning is required instead to cover these surfaces. This is the case of an AUV covering the seabed (Hert et al., 1996) or a robot spray-painting vehicle parts (Atkar et al., 2005b), for instance. Next, we review several 3-dimensional coverage methods. It is worth noticing that, except for the algorithm discussed in Section 2.8.1, the methods discussed below actually focus on coverage of a surface of lower dimension than the robot's workspace. Indeed, in 3-dimensional coverage, covering 2-dimensional surfaces embedded in 3-dimensional space such as the boundaries of automotive parts, the boundaries of buildings, the ocean floor, rugged agricultural fields or the boundaries of the in-water part of a ship hull are the main focus. This contrasts with the standard coverage path planning problem, in which all the free space must be covered.

2.8.1 *3D Coverage using a Planar Coverage Algorithm in Successive Horizontal Planes*

Hert et al. (1996) presented a 3D coverage algorithm that is based on a planar 2-dimensional terrain-covering algorithm (Lumelsky et al., 1990). Their target application is an AUV imaging the sea bottom. Their solution applies to a 3D projectively planar environment by applying the planar terrain-covering algorithm in the successive horizontal planes at different depths. The restriction to projectively planar environments means that elements such as caves are not handled by this method. Their 2D terrain-covering algorithm uses a partial discretization of the space in where the space is divided in vertical slices of the same width, but where the top and bottom of each slice can have any shape. This discretization is classified as a semi-approximate cellular decomposition according to Choset's taxonomy (Choset, 2001). A robot following this algorithm may start at an arbitrary point in the environment and will zigzag along parallel straight lines (grid lines) to cover the given area. Portions of the area that either would not be covered or would be covered twice using the zigzag procedure are detected by the robot and covered using the same procedure; that is, the procedure is applied recursively. These smaller areas, called inlets, are covered as soon as they are detected and inlets within inlets are treated in the same way. Hence, the inlets are covered in a depth-first order. By requiring the robot to remember the points at which it enters and exits every inlet it covers (which define the inlet doorways), the algorithm assures that each inlet is covered only once.

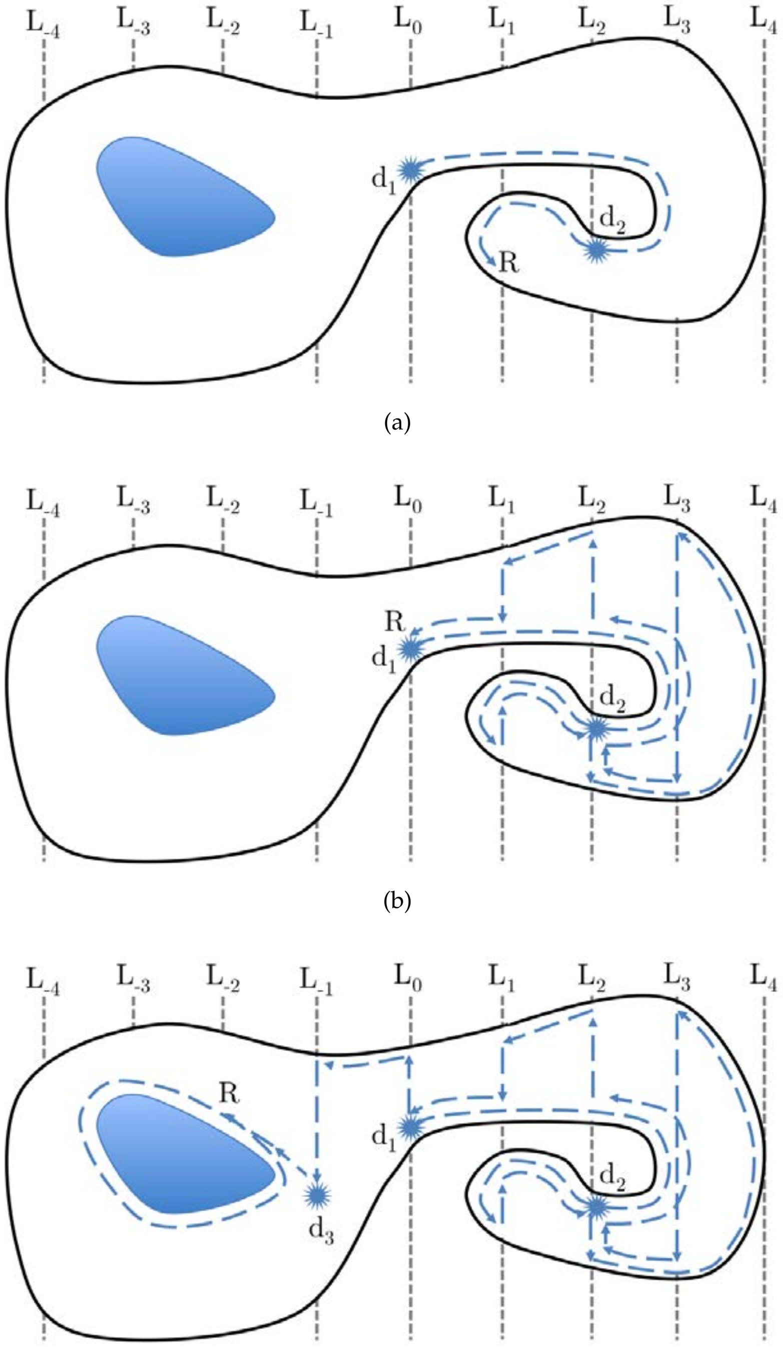
When entering or exiting a certain type of inlet, the robot may cover the same area more than once, or miss some area at the inlet. Those inlets are called diversion inlets, and special procedures are necessary for covering them effectively. The robot enters a diversion inlet by moving along its boundary. After covering a given diversion inlet, the robot exits it by resuming its path as if the diversion inlet did not exist. When the area to be covered is not simply connected and contains islands as well as inlets, the same basic procedures are used, but with minor modifications to ensure that the area surrounding every island is covered. The robot is able to convert the part of the area around each island that would normally not be covered into an artificial inlet by remembering certain points along its path. Artificial inlets are covered in the same way that real diversion inlets are. Figure 27 illustrates this procedure. It is worth noticing, however, that details on how to detect the inlets used by the algorithm using sensor information are not provided.

The work of Hert et al. (1996) was extended later by Lee et al. (2009) to cover only areas that are close to the sea bottom surface. In this latter work, it is assumed that the regions of interest in underwater environments are the ones close to the sea bottom. Therefore, aiming to make the robot navigate only in areas close to the surface, artificial obstacles (artificial islands) are introduced in the robot's map of the environment. This way, the volume of water at a certain distance from the seafloor surface is discarded and a more efficient exploration of the sea bottom is achieved.

A theoretical proof of correctness of the algorithm is given by Hert et al. (1996) and the extension proposed by Lee et al. (2009) is validated in simulation.

2.8.2 *3D Cellular Decomposition*

Atkar et al. (2001) considered the problem of trajectory generation for spray-painting robots. In their initial work, they proposed an on-line, 3-dimensional coverage path planning method for closed, orientable surfaces embedded in $\mathbb{R}^3$. The method extended the ideas of Morse decomposition to non-planar spaces. However, obstacles on the target surface are not considered in this work. Addressing their spray-painting target application, the method does not plan a coverage path on the target surface, but the coverage path is rather planned in an offset surface from which the end effector will spray the target surface. That is, the path is planned on a "virtual" surface that wraps the target object at a fixed offset distance. The coverage path is generated by intersecting a slice plane with the offset surface at equally spaced intervals. At each interval, the intersection of the slice plane with the offset surface forms a closed one-dimensional loop around the object. The robot traces this loop and moves to the next slice plane, and iteratively repeats the process. If the target surface is convex, the described process will achieve complete coverage. However, if the surface is non-convex and includes elements such as bifurcations, the planner will use the critical points occurring in such shape changes to divide the surface in cells that will be covered individually. As in the on-line Morse decomposition for planar spaces, a Reeb graph is used to encode the topology of the target surface. When all edges in the graph are covered, the coverage task is successfully completed. The method was validated in simulation using target surfaces constituted by polyhedra. It is



(a)

(b)

(c)

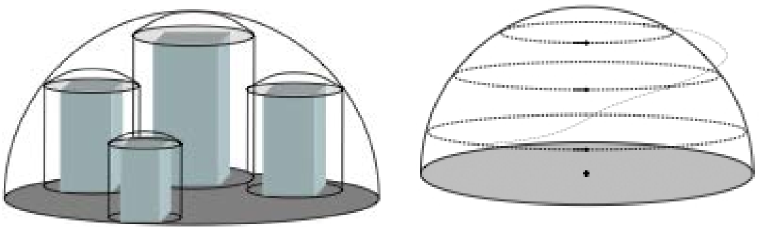
Figure 27: The path a robot R follows in a non-simply connected environment when applying the algorithm proposed by Hert et al. (1996). First (a), the robot detects an inlet at d_1 and starts to cover it following a depth-first order. A second inlet is detected at d_2, and the robot starts covering it likewise. The robot continues to cover the rest of the inlets until it goes back to d_1 (b). Here, the robot continues to cover the main region until it detects an inlet at d_3. This inlet corresponds to an island, and hence the robot continues to circumnavigate it completely (c). Then, the robot will eventually pass through d_3 again and there it will resume the covering of the main area.

worth noting that this method requires a robot equipped with a 2D omnidirectional range sensor in order to detect the critical points.

In later work (Atkar et al., 2003, 2005a,b), they presented an off-line coverage path planning method specifically targeted for spray-painting of automotive parts. They term such surfaces pseudo-extruded surfaces. By contrast with their initial work, the problem tackled here is the uniform coverage problem, where the target surface not only needs to be completely covered but also the resulting paint deposition must meet certain uniformity requirements. To achieve uniform coverage, their proposed method takes a CAD model of the target automotive parts as input and segments their surface into topologically simple cells of similar curvature. Then, individual, optimal paint-deposition coverage paths in each cell are determined. Simulations as well as experiments with real robots validate their proposal.

2.8.3 *3D Urban Structure Coverage*

Cheng et al. (2008) presented an off-line approach for planning time-optimal trajectories for Unmanned Aerial Vehicles (UAVs) performing 3D urban structure coverage. First, they simplify the structures to be covered, namely buildings, into hemispheres and cylinders. Then, trajectories are planned to cover these simpler surfaces. Their proposal is validated in hardware-in-the-loop simulations using a fixed-wing aircraft. Figure 28 illustrates this method.



(a) Simplified model of an urban environment.

(b) Illustration of a coverage path.

Figure 28: Approach for 3D urban structure coverage. Image credit: Cheng et al. (2008).

2.8.4 *3D Coverage for Arable Farming*

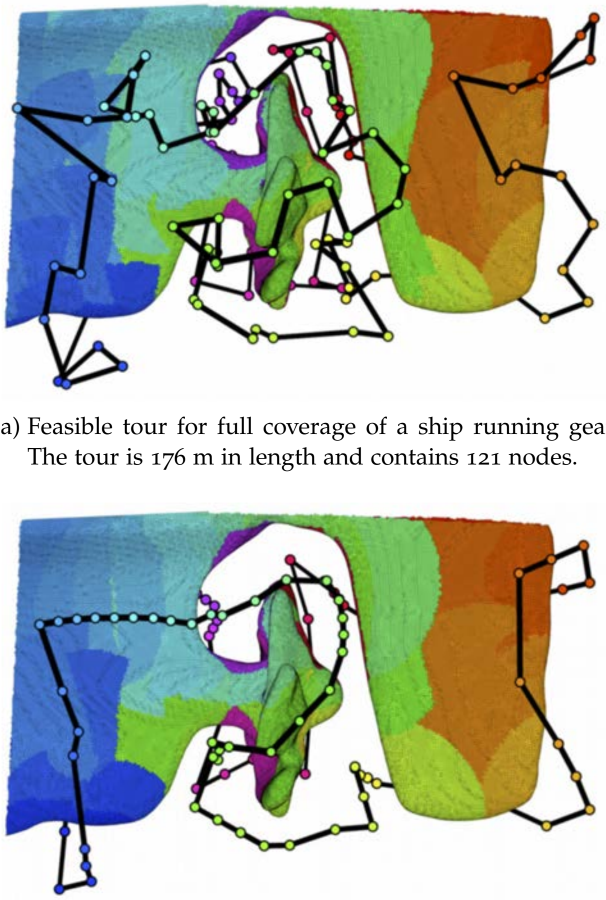
Jin and Tang (2011) presented coverage algorithms for arable fields represented as elevation maps. Previous work in coverage for agricultural fields dealt with planar terrain, but many fields present 3-dimensional features that have an impact on coverage performance. Addressing this issue, this work provides coverage algorithms based on a seed curve that is incrementally offset on both sides to generate a coverage path. The method optimizes the seed curve selection by taking into account its associated number of turns, the soil erosion cost and the skipped area. The method is validated on real-world elevation maps of agricultural fields.

2.8.5 *Random Sampling-based Coverage of Complex 3D Structures*

In confined 3-dimensional areas where a robot cannot go through the spaces between component structures, or where occluded areas are only visible from a reduced set of viewpoints, modular approaches such as those described above are unsuitable. To handle this family of problems, global path planning strategies utilizing sampling-based planning (Danner and Kavraki, 2000) have been applied to find feasible, collision-free paths through confined areas and obtain full coverage of a 2-dimensional target structure. Their approach is based on the art gallery problem. Building upon a similar idea, Englot and Hover (2012) introduce an off-line, sampling-based coverage algorithm to achieve complete sensor coverage of complex, 3-dimensional structures. Their target application is autonomous ship hull inspection, in which the robot must cover the in-water part of the hull surface using a sensor such as a sonar. The sensory data collected *in situ* is later used to construct an accurate 3-dimensional model where anomalies in the hull surface can be searched for. They consider the planning problem with a fully-actuated, 6-Degree Of Freedom (DOF) hovering AUV that uses a bathymetry sonar to inspect the structures in the ship hull. The method requires a discrete model of the structure to be inspected provided in the form of a closed triangular mesh. The planning is performed in two steps. First, a graph of feasible paths for the robot is constructed using random sampling until the set of nodes of the graph allows complete coverage of the structure. This is equivalent to solving a variant of the art gallery problem. Then, a minimum-cost closed walk along the graph which fully covers the structure is searched for in the graph. This second step involves solving a variant of the TSP. By favoring a random sampling method, they reduce the computational burden necessary to face the high-dimensionality of the problem. It should be noted that the generated paths cover cluttered spaces where complex structures such as shafts and rudders are present. The approach is validated using sensor imagery of real vessels and with experiments conducted at sea. Furthermore, a method for smoothing and shortening the paths initially generated is provided. This procedure can be incrementally applied while computation time allows. Figure 29 shows examples of planned inspection paths for a ship hull with and without smoothing.

As discussed above, the approach by Englot and Hover (2012) first generates a set of view configurations that completely cover the target surface (by solving an instance of the art gallery problem) and then finds a path that connects them (by solving an instance of the TSP). This might pose a problem for robots with differential constraints, given that a path connecting to a given view configuration might be infeasible. To tackle this problem, Papadopoulos et al. (2013) presented a random sampling-based algorithm that incrementally explores the robot's configuration space while constructing an inspection path until all points on the target surface are guaranteed to be covered. In contrast to the aforementioned approaches, which first plan a set of view configurations that cover the target environment, their algorithm generates view configurations and at the same time validates the feasibility of the path connecting them. Only view configurations reached by feasible paths are incorporated in the final coverage path. Additionally, this method is probabilistically optimal with respect to a given cost function. The method is validated in simulation.

Given the wide variety of structures that are able to handle, these approaches constitute the state of the art in coverage of complex 3-dimensional structures.



(a) Feasible tour for full coverage of a ship running gear. The tour is 176 m in length and contains 121 nodes.

(b) Tour of (a) after applying the refinement procedure. The shortened tour is 102 m in length and contains 97 configurations.

Figure 29: Full-coverage inspection paths obtained with the method of Englot and Hover (2012). Image credit: Brendan Englot.

2.9 OPTIMAL COVERAGE

Work addressing the optimality of the planned coverage paths, in terms such as path length and time to completion, appears in the coverage path planning literature. Notice that it is only possible to find an optimal solution for an *a priori* known environment, or partially known at least, since an antagonistic example can always be found for a sensor-based approach. Hence optimal coverage methods are classified as off-line methods.

Huang (2001) presented an optimal line-sweep based method for cellular decomposition algorithms in planar spaces. This approach produces an optimal length coverage path by allowing different sweep directions in the lawnmower paths used to cover each cell. The main idea is to minimize the number of turns in the path, as each turn typically implies the added cost of the robot decelerating and accelerating again after the turn. This is achieved by allowing a different sweep direction in each cell. The number of turns is minimized by sweeping each cell in parallel to its maximal altitude axis. That is, the method intends to maximize the length of the laps in the zigzag pattern in order to minimize the number of turns. However, this approach does not take into account the cost of traveling from cell to cell. The method is validated in simulation.

Jimenez et al. (2007) proposed to use a genetic algorithm to achieve optimal coverage. In this proposal, workspace and obstacles are assumed to be polygonal and known beforehand. Then, the free space is divided in subregions using the trapezoidal cellular decomposition method (Latombe, 1991; Choset et al., 2005). Finally, a genetic algorithm is used to plan an optimal path that covers all the subregions. This proposal is tested in simulation.

Mannadiar and Rekleitis (2010) proposed an algorithm based on the boustrophedon cellular decomposition that achieves complete coverage of known spaces while minimizing the path of the robot. The algorithm encodes the cells to be covered as edges of the Reeb graph. Then, the optimal solution to the Chinese Postman Problem is used to calculate an Euler tour, which guarantees complete coverage of the available free space while minimizing the coverage path length.

Xu et al. (2011) presented an application of the optimal Morse-based boustrophedon decomposition method (Mannadiar and Rekleitis, 2010) for UAVs. First, they generate an optimal exhaustive walk through the adjacency graph of the cell decomposition of the terrain. Then, they cover each cell with zigzag motions taking into account the kinematic constraints of the vehicle, as the fixed-wing UAV they use has non-holonomic constraints. Extensive experimental results in simulation validate the presented system, along with data from over 100 kilometers of coverage flights using a real fixed-wing aircraft.

2.10 COVERAGE UNDER UNCERTAINTY

In many scenarios, the lack of a global localization system such as GPS makes the robot accumulate drift, and hence a growing uncertainty about its pose. Although the topological representations such as the adjacency graph are tolerant to localization error, the performance of coverage algorithms, even if using such representations, is still affected (Mazo and et al., 2004; Choset, 2001). This is because the amount of coverage within a cell depends on the direction of the zigzag pattern.

Recent advances in SLAM have greatly improved robot localization. SLAM uses statistical techniques to correct the robot's pose (position and orientation) estimation. However, the problem of correcting the robot's pose while performing coverage has been only addressed in little research.

Acar and Choset (2002b) propose to plan the paths of their sensor-based Morse decomposition approach by relying on the boundaries of each cell, hence minimizing the dead-reckoning error.

Tully et al. (2010) used a fleet of three robots, each one of them equipped with a red ball (easily detectable using standard computer vision techniques) to follow a strategic path in formation to minimize the localization error. The mentioned robot fleet is shown in Figure 30. The path consists of a series of steps, or leaps. In each leap, two robots are static and serve the third one as beacons, and this later one advances. The robots successively interchange their roles. Real experiments show a minimization of the localization error, reporting one of the most successful 2-dimensional coverage experiments to date. However, obstacles are not considered in this work.

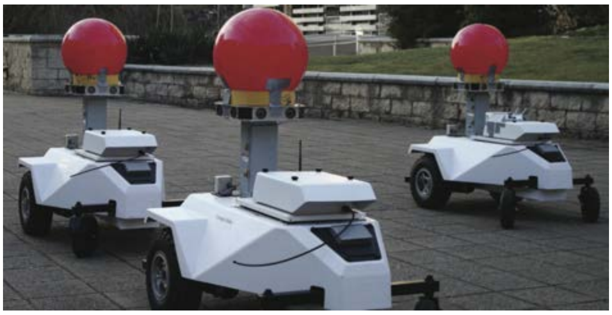


Figure 30: Team of three robots used for experimental evaluation of the leap-frog localization and coverage strategy. Image credit: Stephen Tully.

Kim (2012) propose an active SLAM approach to coverage path planning for ship hull inspection in a 3D scenario. The proposed algorithm drives the robot along a pre-planned coverage trajectory on the ship hull, and during trajectory execution the robot selects candidate locations that, once revisited, can help reduce the robot's pose uncertainty. The algorithm chooses to revisit a candidate location once the pose uncertainty surpasses a user-provided threshold, and otherwise follows the pre-planned path.

Bretl and Hutchinson (2013) suggest a way to plan modified coverage paths for a mobile robot whose position and velocity are subject to bounded error. Assuming a worst-case model of uncertainty they are able to guarantee complete coverage. This guarantee comes at the cost of a longer path, since paths generated by their algorithm include retracing. Nonetheless, this work provides the first guaranteed coverage results for the case of bounded position and velocity error.

2.11 MULTI-ROBOT METHODS

There are advantages in using multiple robots in a coverage path planning task. Using multiple robots clearly decreases the time to complete the task due to workload division. But a team of robots can go further, for example using each other as beacons to minimize localization error. Additionally, using multiple robots improves robustness, as failure of some members of the robot team can be compensated by others. There exist a number of multi-robot coverage path planning proposals in the literature. Most approaches extend single-robot ideas presented above to multiple robots by using a strategy to divide the workload. In this section, we discuss multi-robot coverage methods based on the single-robot boustrophedon decomposition, on spanning trees, on the biologically inspired neural-network approach and on the graph-based approach. Nonetheless, there are genuine multi-robot approaches which are not based in any particular single-robot algorithm, which we also discuss below.

2.11.1 *Multi-robot Boustrophedon Decomposition*

Rekleitis et al. (2009) presented a collection of algorithms for the complete coverage path planning problem using a team of mobile robots on an unknown environment (on-line). Their algorithms aim to minimize repeated coverage. The algorithms use the same planar cellular decomposition as the Boustrophedon single robot coverage algorithm, but provide extensions to handle how robots cover a single cell, and how robots are allocated among cells. Their solution takes into account communication restrictions among the members of the team. To achieve coverage in line-of-sight-only communications, the robots take two roles: some members, called explorers, cover the boundaries of the current target cell, while the other members, called coverers, perform simple back-and-forth motions to cover the remainder of the cell, as shown in Figure 31. For task/cell allocation among the robots, a greedy auction mechanism is used. Experimental results from different simulated and real environments are provided to illustrate their proposed approach.

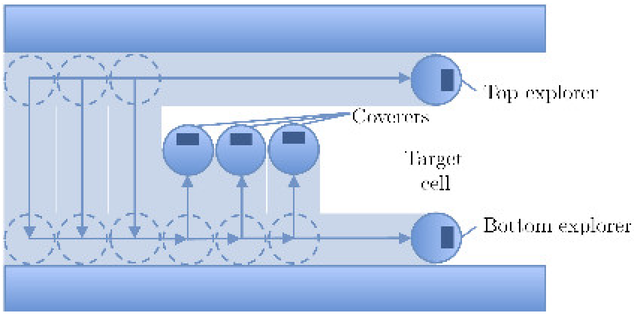


Figure 31: Explorer/coverer approach, where two explorer robots outline the top and bottom boundaries while the remaining robots (coverers) execute simple back-and-forth coverage of the target cell.

2.11.2 *Multi-robot Contact Sensor-based Coverage of Rectilinear Environments*

A strategy for covering rectilinear environments using robots equipped only with contact sensors was proposed by Butler et al. (2000). The strategy is based on CC_R, discussed in Section 2.5, and is called DC_R. DC_R decouples cooperation and coverage by executing CC_R individually on each robot and adding a higher-level coordinator, termed the "overseer", which is in charge of controlling the cooperation among the robots. The overseer operates in such a way that coverage directed by CC_R on each robot can continue without CC_R being aware that cooperation is occurring. A proof of completeness for DC_R is provided in this work.

2.11.3 *Multi-robot Spanning Tree Coverage*

The STC method was generalized to multi-robot teams using a heuristic approach by Hazon and Kaminka (2005). They termed their algorithm Multi-robot Spanning Tree Coverage (MSTC). Zheng et al. (2005) presented an improved (with respect to MSTC) method to find a coverage tree for a team of robots to cover known terrain. In this work they also provide an upper bound on the performance of a multi-robot coverage algorithm on known terrain, guaranteeing a performance of at most eight times the optimal cost. Their reported experimental results show their method to perform significantly better than MSTC. Agmon et al. (2006) propose a spanning tree construction algorithm that provides efficient paths in terms of distance. The spanning tree construction algorithm can be used as base for MSTC. An extension of MSTC to terrain with non-uniform traversability (that is, terrain where traversing certain areas is costlier than others) was presented by Zheng and Koenig (2007). Hazon et al. (2006) presented an on-line, robust version of MSTC. They show analytically that the algorithm is robust, guaranteeing coverage as log as a single robot is able to move. Empirical results validating the algorithm are reported.

A more recent off-line spanning tree-based multi-robot coverage method presented by Fazli et al. (2010) deals with the case where the robots have a limited visibility range. This approach is shown to be complete and robust with respect to robot failure.

2.11.4 *Multi-robot Neural-network-based Coverage*

Luo and Yang (2002) presented a straightforward adaptation of the biologically inspired neural network approach for coverage tasks to multi-robot scenarios where the robots see each other as moving obstacles. In a later work (Luo et al., 2003), an extension was provided to avoid deadlock situations between the robots. Their approach is validated in simulation.

2.11.5 *Multi-robot Graph-based and Boundary Coverage*

Extensions of the graph-based techniques discussed in Section 2.7 for coverage using multiple robots were provided by Xu (2011). In a previous multi-robot graph-based coverage approach, Easton and Burdick (2005) discuss a 2-dimensional boundary coverage algorithm for multiple robots. It is worth mentioning that, as in the majority of 3-dimensional coverage methods discussed in Section 2.8, this work focuses on covering only the boundary of the target environment. In the multi-robot boundary coverage problem, introduced in this work, a team of robots must inspect all points on the boundary of the 2-dimensional target environment. A motivating application of the multi-robot boundary coverage problem is inspection of separated blade surfaces inside a turbine. The boundary coverage problem is converted into an equivalent graph representation where a heuristic search is used to plan the inspection routes of every robot. The planned routes provide complete coverage of the boundary while balancing inspection load among the robots. The algorithm is validated in simulations.

2.11.6 *Bio-inspired Multi-robot Coverage*

Several muli-robot coverage path planning proposals have been presented which are inspired by behaviors found in nature. Many of them are inspired by ant behavior, using evaporating traces to achieve an emergent coverage behavior (Wagner et al., 1999, 2008; Menezes et al., 2007). In (Batalin and Sukhatme, 2002) two algorithms are presented which are based on the premise that to achieve coverage the team of robots must "spread out" over the environment. The authors note that "this premise is loosely inspired by the diffusive motion of fluid particles". Using these algorithms robots perform obstacle avoidance and at the same time are mutually repelled by each other within their sensor range. These bio-inspired works are validated in simulation, but their practical application has been very limited up to date.

2.11.7 *Multi-robot Coverage for Aerial Robotics*

A considerable body of research has addressed multi-robot coverage path planning for fleets of aerial robots, taking into account the particulars of this domain. In these works it is typically assumed that the vehicles fly at a safe altitude, and hence obstacles are not considered.

Ahmadzadeh et al. (2006) proposed a coverage algorithm for surveillance using a fleet of UAVs. Their proposal takes into account the limited maneuverability of the aerial platforms and visibility constraints on the body-fixed cameras imposed by the application at hand. The problem is posed using the integer programming formalism, which provides a convenient representation for the aforementioned constraints. The solution of the integer programming problem instance produces a control policy for the UAV fleet to accomplish the surveillance task operating within the constraint limits. The efficacy of this approach is validated by simulation and experimental results.

Maza and Ollero (2007) proposed a terrain coverage strategy using a heterogeneous fleet of UAVs. First, their method generates a polygonal partition of the target area. The partition takes into account the capabilities of each individual vehicle, such as flight endurance and range. Each polygon in the partition is assigned to an UAV which will cover it using a zigzag pattern. Each vehicle plans its zigzag pattern according to the geometric characteristics of its assigned polygonal area to determine a sweep direction that minimizes the number of turns. An important consideration in this work is low complexity of the algorithms used, seeking operation in near-real time. The proposed method is validated in simulation.

Targeting remote sensing in agriculture, Barrientos et al. (2011) presented an approach to area coverage using fleets of mini aerial robots. Regarding multi-robot coverage, they first present a task scheduler to partition the global target area into k non-overlapping subtasks for the k UAVs. By contrast with the work by Maza and Ollero (2007), this partition procedure is based on a negotiation process in which each robot claims covering as much area as possible, rather than on geometric considerations. After a partition is obtained, the wavefront algorithm discussed in 2.6.1 is used to cover each subarea.

2.12 DISCUSSION

In this survey, we have seen that the coverage path planning problem has been addressed using many different approaches. For planar spaces, the trapezoidal decomposition guarantees complete coverage for a known polygonal environment. An improvement to the trapezoidal decomposition is the "classical" boustrophedon decomposition, which generates shorter complete coverage paths for the same class of environments. The Morse-based cellular decomposition provides complete coverage paths for environments whose obstacle boundaries are differentiable. A method to detect the critical points that determine the cell boundaries using range sensor information allows to perform Morse-based cellular decomposition coverage on-line. Furthermore, Morse decomposition allows generation of different coverage patterns, such as spiral patterns, that can simplify path following for vehicles with motion constraints.

However, the Morse-based cellular decomposition method cannot handle rectilinear environments, and the cyclic rectangular paths used to detect all the critical points include retracing, which makes them longer than a standard zigzag path. This limitations are overcome by the landmark-based topological coverage approach. This method uses a cellular decomposition based on natural landmarks of the environment which determine the cell boundaries. An algorithm is given to perform coverage on-line on unknown environments using this technique.

For the particular case of robots with only contact sensors (i.e., with no range sensing capabilities) operating in rectilinear environments, the CC_R algorithm guarantees complete on-line coverage.

Grid-based methods such as the wavefront algorithm, the Spiral-STC algorithm and its derivatives, and the described neural-network-based and hexagonal decomposition approaches, provide complete coverage on a discretized representation of the target environment. However, the grid representation of the environment used is highly sensitive to localization error and incurs an exponential memory consumption. On the other hand, it is easy to create and operate with a grid map. It is worth noticing, as a unique capability among the reviewed methods, that the discussed neural network-based methods are able to handle environments with moving obstacles.

The work of Xu (2011) provides coverage algorithms for environments that can be represented as a graph, such as a street or road network.

Some methods aimed to cover 3-dimensional environments have been reviewed. Hert's algorithm can completely cover projectively planar 3D environments. However, details on how to detect the inlets used by the algorithm using sensor data are not provided, making it difficult to implement. Modular approaches, such as the coverage methods targeted at spray-painting tasks proposed by Atkar et al. or the simplified model of a urban environment used by Cheng et al. can achieve complete coverage of certain 3-dimensional environments. The coverage algorithm for arable farming by Jin et al. provides coverage of 3-dimensional environments taking into account application-specific constraints. However, in confined 3-dimensional areas where a robot cannot go through the spaces between component structures, or with occluded areas only visible from a reduced set of viewpoints, these modular approaches do not suffice. To overcome this limitation, Englot et al. introduced a sampling-based coverage algorithm to achieve complete sensor coverage of complex, 3-dimensional structures. The paths generated using this method are able to cover cluttered spaces where complex structures such as shafts and rudders are present. The approach is validated using triangular mesh models constructed using sensor imagery of real-world vessels. Building also upon the idea of sampling-based coverage, Papadopoulos et al. presented an algorithm that generates coverage paths for complex structures suitable for vehicles with differential constraints. Furthermore, their algorithm is proven asymptotically optimal with respect to a given cost function. These two later approaches constitute the state of the art in coverage of complex 3-dimensional structures due to the high complexity of the environments they can handle.

We have discussed several approaches to generate optimal coverage paths in planar spaces and to reduce localization error while performing coverage.

Finally, we have reviewed some multi-robot coverage methods in which time to completion is reduced by dividing the workload among the individual robot team members, besides providing increased robustness guarantees.

The features of the most relevant coverage path planning methods reviewed in this article are summarized in Table 1. For each method (rows), the table underlines (columns from left to right) its category, its approach, its main bibliographical references, whether it can be used on-line or not, the kind of environments it can handle and some remarks.

Probabilistic sampling-based algorithms have revolutionized the state of the art in path planning in the recent years, and they have proven to be extremely powerful as demonstrated in the work by Englot et al. on ship hull inspection. Therefore, exploiting these techniques opens the door to developing algorithms able to realize coverage tasks of unprecedented complexity. On the other hand, in real-world applications, a robot often does not have perfect knowledge about its location nor its environment. This is especially patent in the underwater domain, where the lack of ubiquitous absolute positioning such as GPS and the limited accuracy of typical AUV sensors dramatically constrain the precision of the vehicle's perception. In this situation, incorporating uncertainty in future location estimates in the planning phase can significantly improve motion performance. Although several research works have explored taking uncertainty into account in path planning problems, little attention has been paid to incorporating uncertainty in coverage path planning methods. Hence, this remains as an important open subject for further research.

survey summary

category	approach	reference(s)	on/off-line	environments	remarks
Classical Exact Cell Decomposition	Trapezoidal Decomposition	(Choset et al., 2005)	Off-line	Polygonal	Introduces the concept of using events to determine cell divisions, a concept a number of other approaches are based upon.
	Boustrophedon Decomposition	(Choset and Pignon, 1997)	Off-line	Polygonal	Generates less cells and hence shorter paths than the trapezoidal decomposition.
Morse-based Cell Decomposition	Morse Decomposition & Cycle algorithm	(Acar et al., 2002)	On-line	Polygonal and differentiable boundaries (non-rectilinear)	Allows for generation of different decomposition and coverage path patterns.
	Morse Decomposition + GVD	(Acar et al., 2006)	On-line	Polygonal and differentiable boundaries (non-rectilinear)	Avoids generation unnecessary zigzag paths in narrow environments.
Natural Landmark-based Topological Coverage	Landmark-based Coverage Algorithm	(Wong and MacDonald, 2003)	On-line	Generic planar obstacles	Handles a large variety of environments (including rectilinear environments).

Contact Sensor-based Coverage	Rectilinear decomposition (CC_R algorithm)	(Butler et al., 1999)	On-line	Rectilinear	Targeted for robots equipped only with contact sensors.
Grid-based Coverage	Wavefront Algorithm	(Zelinsky et al., 1993)	Off-line	Grid-discretized	Simple, easy to implement algorithm.
	Spiral-STC Algorithm	(Gabriely and Rimon, 2002)	On-line	Grid-discretized	Minimizes repeated coverage by visiting each grid cell only once.
	Neural Network	(Luo et al., 2002; Yang and Luo, 2004; Luo and Yang, 2008)	On-line	Grid-discretized	Handles dynamic obstacles.
	Hexagonal Grid	(Paull et al., 2012)	On-line	Grid-discretized	Maximizes information gain along the path.
Graph Coverage	Various graph search algorithms	(Xu, 2011)	On-line	Environmental constraints	Applies to environments that can be represented as a graph, such as a street or road network.

3-dimensional Coverage	2.5D covering algorithm	(Hert et al., 1996)	On-line	2.5D	Theoretically proven, but no details on detection of events provided.
	3D cellular decomposition	(Atkar et al., 2001)	On-line	Closed, orientable surfaces embedded in $\mathbb{R}^3$	Theoretically proven. Demonstrated in simulation on simple 3D surfaces.
	Hierarchical segmentation	(Atkar et al., 2009)	Off-line	Car-like parts	Targeted at coverage of automotive parts, uniform paint deposition.
	Planning on simplified 3D surfaces	(Cheng et al., 2008)	On-line	Urban environments	Suitable for covering urban structures with sufficient clearance.
	Coverage for Arable Farming	(Jin and Tang, 2011)	Off-line	Elevation maps	Minimizes application-specific costs.
	Random sampling-based	(Englot and Hover, 2012)	Off-line	Complex 3D structures	Allows for coverage of complex 3-dimensional structures such as a ship propeller.
	Random sampling-based	(Papadopoulos et al., 2013)	Off-line	Complex 3D structures	Can handle differential constraints and probabilistically guarantees optimality.

Optimal Coverage	Cell decomposition with variable sweep direction	(Huang, 2001)	Off-line	Polygonal	Takes into account “cell height” to select the optimal sweep direction.
	Trapezoidal decomposition + genetic algorithm	(Jimenez et al., 2007)	Off-line	Polygonal	A genetic algorithm quickly finds a specific coverage paths among the cells.
	Morse decomposition + optimal adjacency graph traversal	(Mannadiar and Rekleitis, 2010)	On-line	Polygonal and differentiable boundaries (non-rectilinear)	Finds an optimal walk through the adjacency graph.
Coverage under Uncertainty	Exploiting critical points	(Acar and Choset, 2002b)	On-line	Polygonal and differentiable boundaries (non-rectilinear)	Efficiently increases actual percent coverage achieved.
	Active SLAM	(Kim, 2012)	On-line	Ship hull	Decides when to revisit a salient feature when executing coverage to reduce uncertainty.
	Leap-frog strategy	(Tully et al., 2010)	On-line	Obstacles not considered	Robots in a team use each other as beacons alternatively.
	Modified boustrophedon paths	(Bretl and Hutchinson, 2013)	Off-line	Planar	Guarantees complete coverage under bounded position and velocity error.

Multi-robot Coverage	Boustrophedon-based	(Rekleitis et al., 2009)	On-line	Polygonal and differentiable boundaries (non-rectilinear)	Extension of Morse decomposition to multi-robot teams.
	Contact sensor-based	(Butler et al., 2000)	On-line	Rectilinear	Extension of CC_R using a decoupled high-level coordinator.
	Spiral-STC-based	(Zheng et al., 2005)	Off-line	Grid-discretized	Extension of Spiral-STC to multi-robot teams.
	Spiral-STC-based	(Hazon et al., 2006)	On-line	Grid-discretized	On-line extension of Spiral-STC to multi-robot teams.
	Neural-network-based	(Luo et al., 2003)	On-line	Grid-discretized	Robots see each other as moving obstacles.
	Boundary coverage	(Easton and Burdick, 2005)	Off-line	2D	Focused on coverage of obstacle boundaries.
	Bio-inspired	(Wagner et al., 2008) and others	On-line	2D	Validated in simulation, but of limited practical application.
	Aerial	(Ahmadzadeh et al., 2006) and others	Off-line	Aerial (obstacle-free)	Account for limited maneuverability and for load balancing in heterogeneous teams.

Table 1: Survey summary.

3

EFFICIENT 2D COVERAGE OF UNDERWATER ENVIRONMENTS

Hey, laser lips,
your mama was a snow blower.

— Number 5

"Short Circuit" (1986)

In this chapter we introduce a new 2D coverage method based on the Morse boustrophedon cell decomposition of Acar et al. (2002), discussed in Section 2.3, that provides AUV survey path planning with a principled way to account for obstacles. The key advantage of our proposed method is that it minimizes redundant coverage incurred when sweeping the ocean floor from a constant depth. After introducing the problem in Section 3.1, the method is detailed in Section 3.2. Section 3.3 presents results obtained in simulation experiments conducted with a real-world bathymetric dataset that show a significant increase on path efficiency in comparison with a standard boustrophedon coverage path.

3.1 INTRODUCTION

As mentioned earlier, many marine robotics applications require coverage of a region on the ocean floor. While performing this task, there are situations in which the vehicle is required to cover the target region navigating at a constant depth. This is the case of Autonomous Surface Vehicles (ASVs), that always navigate at the water surface level. Nonetheless, AUVs are also required to operate at constant depth in certain scenarios, for instance when GPS fixes via an antenna sticking out from the surface are required or when communication constraints with a surface vessel or with another AUV apply. As seen in Chapter 2, most existing coverage algorithms sweep the free space (the space free of obstacles) in the target region using lawnmower-like back-and-forth motions. This motion pattern is convenient in most applications since it is predictable and easy to implement. When using this lawnmower-type pattern the spacing between the back-and-forth laps is determined by the robot's sensor footprint. However, while covering the seafloor surface by navigating at a constant depth, the sensor's footprint varies depending on the distance to the ocean floor. Therefore, to ensure full coverage one would need to use the inter-lap spacing determined by the shallowest point (i.e., the maximum sea floor height) on the target surface, resulting in undesired, inefficient and redundant coverage overlapping among the back-and-forth laps. This problem is illustrated in Figure 32. Next, we propose a novel method based on the Morse cell decomposition (Acar et al., 2002) to minimize this redundant coverage. Note that we propose an off-line method that takes a bathymetric map as input to plan the coverage path. Only well under 10% of the ocean has been bathymetrically charted, therefore this method is only applicable in a limited subset of the oceanic environment. However, the availability of bathymetric charts is increasing as ocean exploration technology progresses. Therefore, so does the applicability of our method. Finally, our method does not take into account potential inaccuracies of the map. Thus its effectivity depends on the resolution and accuracy of the prior map.

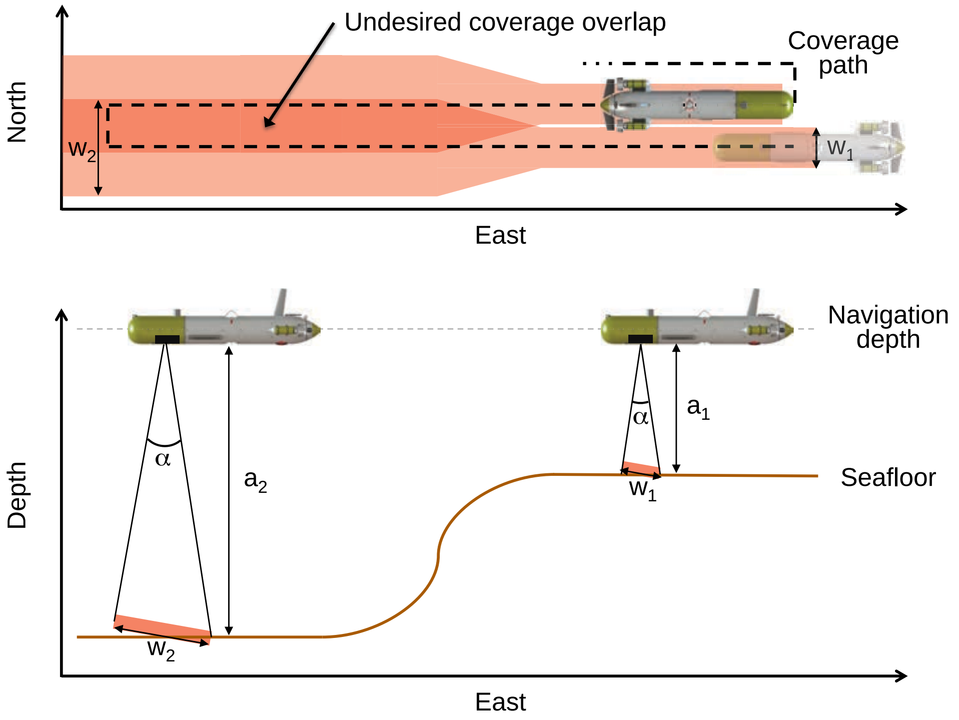


Figure 32: Coverage overlapping problem. As the torpedo-shaped AUV navigates from right to left with a sensor FOV angle α, vehicle altitude varies from a_1 to a_2. The FOV corresponding to a_2, w_2, is considerably wider than the FOV corresponding to a_1, w_1. As the inter-lap spacing is fixed and limited by w_1, undesired coverage overlapping appears.

3.2 EFFICIENT 2D COVERAGE PATH PLANNING FOR MARINE VEHICLES

The method we propose generates an efficient path to cover a projectively planar surface of interest on the seafloor, by means of an autonomous marine vehicle (namely an ASV or AUV) that navigates in a plane at a constant depth and above such surface. We assume the vehicle is equipped with a down-looking sensor with a certain FOV, able to image the seafloor from the navigating depth. Thus, a coverage path in the plane in which the vehicle navigates will be determined. Naturally, protruding parts of the underlying seafloor raising above the navigating plane are considered as obstacles that must be avoided.

Our method is detailed in Algorithm 1. We use information on an *a priori* bathymetric map of the target region to minimize the coverage overlapping. The coverage path planning process begins by segmenting the target surface in regions of similar depth (line 2). By segmenting the surface we seek to minimize the difference between the biggest and smallest possible sensor footprints in a region, and to maximize the inter-lap spacing as a result. Coverage of each segmented region is then addressed as an individual coverage path planning problem (line 3). The Morse decomposition method described in Section 2.3 is applied to each segmented region (line 4) and the sweep orientation to cover each cell is selected to be perpendicular to the principal sea floor gradient (line 6). By sweeping perpendicularly to the underlying sea floor gradient, the difference between the shallowest and the deepest point along a lap is minimized, and hence so is the coverage redundancy. In addition, the inter-lap spacing in the lawnmower-like paths used to cover each cell is maximized on a lap-by-lap basis (line 7), hence obtaining a shorter, more efficient coverage path.

Algorithm 1: Efficient 2D Coverage Path Planning for Marine Vehicles

```
   Input: Bathymetric map, $\mathcal{B}$ and workspace, $\mathcal{WS}$ (obstacle map)
 1 $cellPaths \leftarrow \emptyset$
 2 $regions \leftarrow$ SEGMENT($\mathcal{B}$)
 3 foreach $r \in regions$ do
 4     $cells \leftarrow$ MORSE_DECOMPOSITION($r, \mathcal{WS}$)
 5     foreach $c \in cells$ do
 6         $orientation \leftarrow$ SWEEP_ORIENTATION($c, \mathcal{B}$)
 7         $path \leftarrow$ MAXIMIZED_SPACING_PATH($c, \mathcal{B}, orientation$)
 8         $cellPaths \leftarrow cellPaths \cup path$
 9 $finalPath \leftarrow$ LINK_INDIVIDUAL_PATHS($cellPaths, \mathcal{WS}$)
10 return finalPath
```

Next, we describe our proposed method by showing the effect of every step on a real-world bathymetric dataset we introduce in Section 3.2.1. Sections 3.2.2 and 3.2.3 describe the environment and vehicle models, respectively, used in the coverage path planning process. Section 3.2.4 shows how the *a priori* map is segmented in regions of similar depth. Finally, Section 3.2.5 describes how a coverage path is generated for each segmented region by applying Morse decomposition, selecting an appropriate sweep orientation, and maximizing the inter-lap spacing on a lap-by-lap basis.

3.2.1 *Real-World Dataset*

We use a real-world dataset to demonstrate the successive steps of our coverage path planning method. This dataset is a bathymetric map constructed with data collected near the Formigues islands by CIRS members of the University of Girona in July 2009 (Hurtos, 2009). The Formigues islands are an archipelago of sixteen little islets located about 1300 m off the Canet cape in the Costa Brava in Girona, Catalonia (Spain). A photo of the Formigues islands is shown in Figure 33.

Figure 33: The Formigues islands. Image credit: Primeralinea SIE, S.L.

A 253 m width, 148 m height rectangular area was mapped, with its lower-left corner located at 41° 51′ 34.35″ N, 3° 10′ 38.30″ E. The map's abscissae and ordinates axis are aligned with the Earth's geographical latitude and longitude coordinate system. Depth in the mapped area ranges from about 5.5 m down to 16 m. The bathymetric data were obtained by means of a system composed of a down-looking multi-beam sonar, an Attitude and Heading Reference System (AHRS) and a GPS receiver. The whole system was attached to the hull of a vessel that was used to survey the area. For the purpose of demonstration of our coverage path planning method and to obtain a complete map, basic image processing techniques were applied to filter out noise originally present in the data and missing data points were interpolated. Figure 34 shows the post-processed bathymetric map.

3.2.2 *Environment Modeling*

We use an environment model in our coverage path planning method composed of the 2.5D surface of interest (namely the seabed) and the planar surface in which the vehicle navigates (the vehicle's workspace). We model the 2.5D surface of interest as a heightmap, $\mathcal{H}$, where the elevation values are negative downwards, as shown in Figure 34. The planar navigation surface, i.e. the vehicle's workspace, $\mathcal{WS}$, is modeled as an occupancy grid corre-

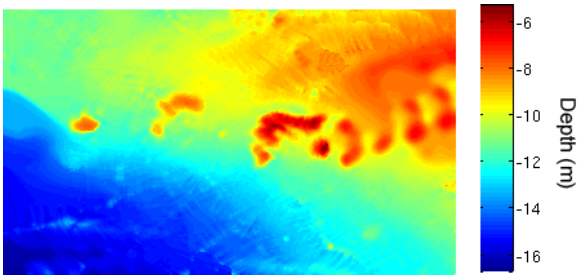


Figure 34: Bathymetric map obtained near the Formigues islands.

sponding to a horizontal slice of the heightmap at the constant vehicle depth, D, such that for a location (x, y) in the heightmap

$$\mathcal{WS}(x,y) = \begin{cases} 1 & \text{if } \mathcal{H}(x,y) \geqslant D \\ 0 & \text{if } \mathcal{H}(x,y) < D \end{cases} .$$

In other words, a point in the workspace is an obstacle (i.e. it is occupied) if the elevation of the surface of interest is above the vehicle's constant depth. Figure 35 shows the workspace obtained by slicing the target surface at $D = -8.5$ m.

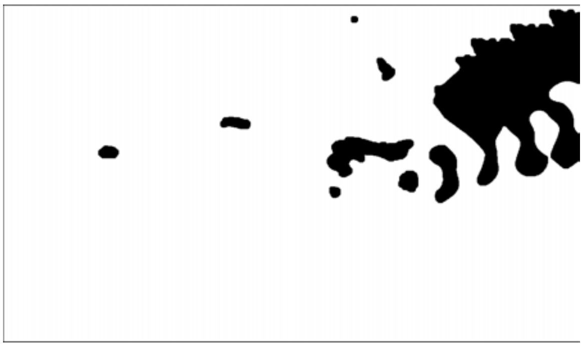


Figure 35: Planar workspace obtained by slicing the target heightmap at depth $D = -8.5$ m.

3.2.3 *Vehicle Modeling*

We model the vehicle as a fully actuated, point-mass system, i.e. having no volume nor weight. Nonetheless, the generated path can be followed by a vehicle with controllability in the surge (motion along the X axis) and heading (rotation about the Z axis) DOFs only, such as a typical torpedo-shaped AUV. It is reasonable to model the vehicle as a point-mass, since the workspace obstacles can be "grown" by an offset distance to accommodate the actual vehicle's radius. The vehicle has a down-looking sensor with a footprint determined by a certain viewing angle, α. It is assumed that, from any given point in the planar navigation surface, the sensor range can reach the underlying surface of interest.

3.2.4 *Surface Segmentation*

The first step of the algorithm (Algorithm 1, line 2) consists in segmenting the heightmap in n regions of similar depth. We use the K-means clustering algorithm (Lloyd, 1982) to obtain an initial segmentation. The number of regions, n, can be selected by the user on a trial-and-error basis, but it can be more convenient to automatically optimize said parameter using the Gap statistic criterion (Tibshirani et al., 2001). Then, we post-process the initial segmentation using the dilate and erode morphological operations (Serra, 1982) with an appropriate structuring element to smooth the region borders and to ensure that the regions are simply connected. Figure 36 shows the segmentation of the bathymetric map shown in Figure 34 with $n = 3$ regions.

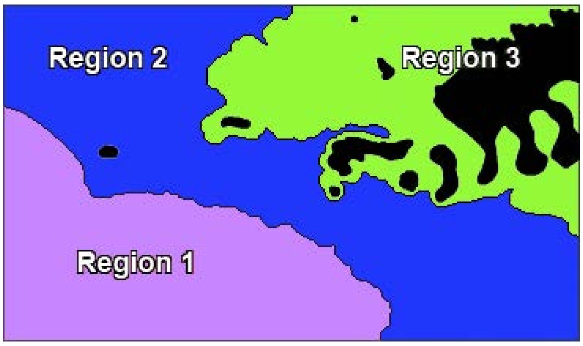


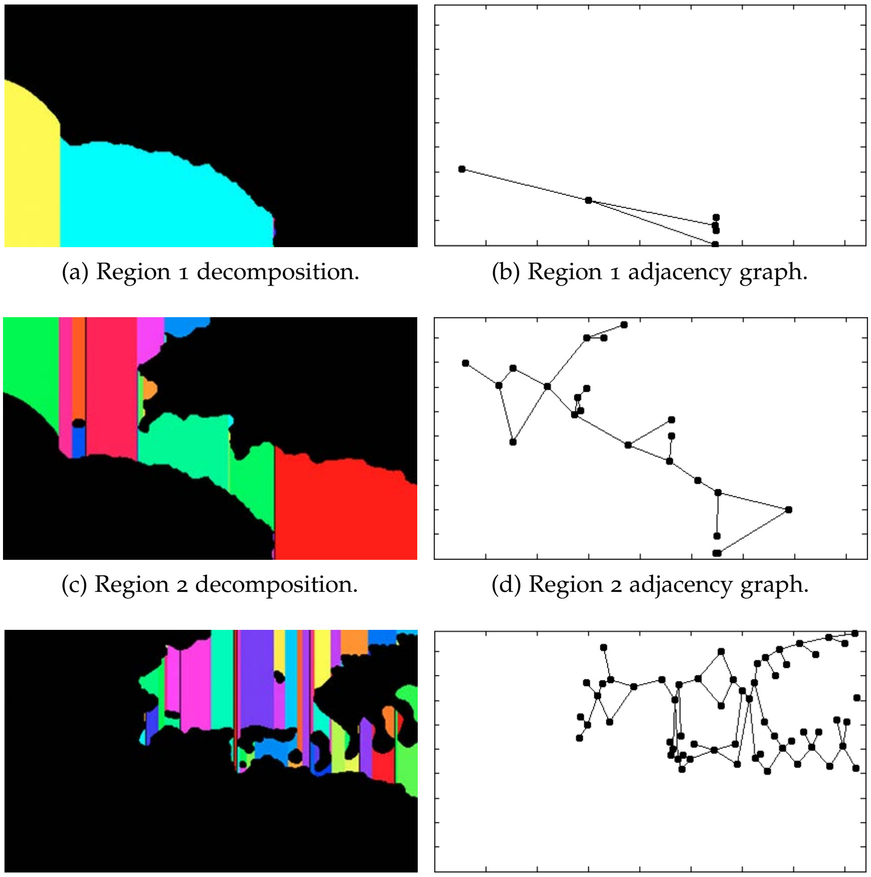
Figure 36: Segmentation of the bathymetric map shown in Figure 34 with $n = 3$ regions. The workspace obstacles are shown in black.

3.2.5 *Coverage Path Generation for Every Region*

Once the surface segmentation is obtained, we actually tackle n different coverage path planning problems, one for each region. As we mentioned, this contributes to minimize the coverage overlapping along the generated path. Each individual coverage path in a region is planned by adhering to the following steps. First, the Morse-based boustrophedon cell decomposition method discussed in Section 2.3 is applied to each region to obtain its cellular decomposition. Second, the sweep orientation in each cell is determined by the gradient of the underlying surface. Third, using the determined sweep directions, a boustrophedon path is generated to cover each cell where the inter-lap spacing is maximized on a lap-by-lap basis. Fourth and last, the individual paths in each region are concatenated to obtain the final coverage path.

3.2.5.1 *Region Cell Decomposition*

The cellular decomposition of each region is obtained by applying the Morse-based boustrophedon cell decomposition method (Algorithm 1, line 4). Figure 37 shows the cellular decompositions obtained for each region of the surface segmentation shown in Figure 36. Exhaustive walks through the adjacency graphs associated to the decompositions (also displayed in Figure 37) are computed. These exhaustive walks determine the order in which the cells are covered.


(a) Region 1 decomposition.

(b) Region 1 adjacency graph.

(c) Region 2 decomposition.

(d) Region 2 adjacency graph.

(e) Region 3 decomposition.

(f) Region 3 adjacency graph.

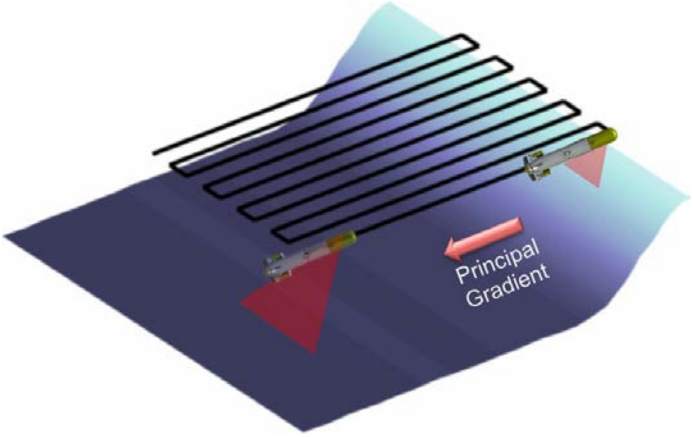
Figure 37: Cell decompositions and associated adjacency graphs of the segmented regions shown in Figure 36. Each cell is shaded with a distinct color.

3.2.5.2 *Sweep Orientation*

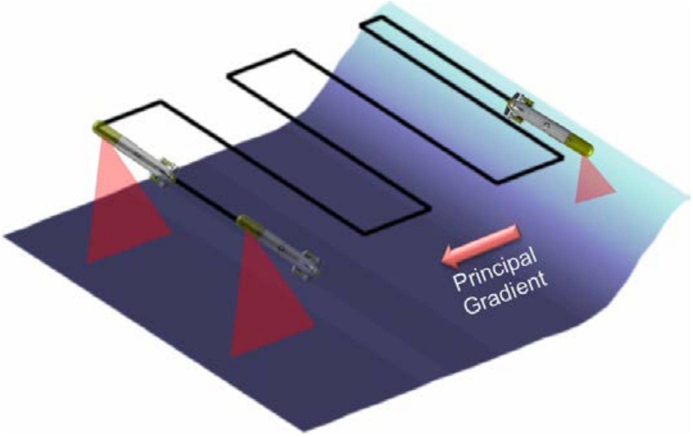
Once the cell decomposition of the region is obtained, we compute the sweep orientation of the individual boustrophedon paths used to cover each cell to be perpendicular to the principal seafloor surface gradient in the cell (Algorithm 1, line 6). The rationale behind this choice is that, as illustrated in Figure 38, navigating perpendicularly to the surface gradient allows for maximizing the inter-lap spacing, as steep bottom surface ascends or descends under a lap are minimized and hence the difference between the lowest and highest surface elevation under a lap is smaller.

The principal sea floor gradient P in a cell is computed as the mean surface gradient under the current cell using the *a priori* map:

$$P = \frac{\sum\limits_{\forall (x,y) \in C} \nabla \mathcal{B}(x_i, y_i)}{|C|}, \tag{3}$$



(a) Sweeping parallel to the terrain's principal gradient (longer coverage path, inefficient).



(b) Sweeping perpendicular to the terrain's principal gradient (shorter coverage path, desired).

Figure 38: Comparison of coverage parallel and perpendicular to the main gradient of the sea floor. When the vehicle sweeps the terrain at constant depth parallel to its principal gradient, the inter-lap spacing is limited by the FOV in the shallow region (a). By contrast, when sweeping perpendicularly to the principal gradient (b), the inter-lap spacing can be maximized on a lap-by-lap basis, leading to a more efficient coverage path.

where $\mathcal{B}(x_i, y_i)$ is the *a priori* map function that returns the elevation at point (x_i, y_i) on the map, C is the set of points on the map belonging to a given cell and the gradient $\nabla\mathcal{B}$ is given by definition as

$$\nabla\mathcal{B} = \frac{\partial\mathcal{B}}{\partial x}\vec{\imath} + \frac{\partial\mathcal{B}}{\partial y}\vec{\jmath}, \tag{4}$$

where $\vec{\imath}, \vec{\jmath}$ are the standard unit vectors in the X and Y axis, respectively.

Intuitively, P provides a measure of the orientation of the main slope in the terrain. We set the sweep orientation of a given cell to be perpendicular to P.

3.2.5.3 *Variable Inter-Lap Spacing Boustrophedon Paths*

Next, we generate the individual boustrophedon paths to cover each cell where the inter-lap spacing varies according to the minimum distance to the bottom surface under each lap (Algorithm 1, line 7). That is, the spacing between the current lap and the next lap is determined by the highest point on the seafloor surface under the current lap, where the minimum sensor FOV width occurs. By adhering to this variable inter-lap spacing strategy we guarantee full coverage of the cell while minimizing the redundant coverage among the laps, hence obtaining a shorter, more efficient path. Figure 39 shows coverage paths obtained for each cell in the decompositions shown in Figure 37.

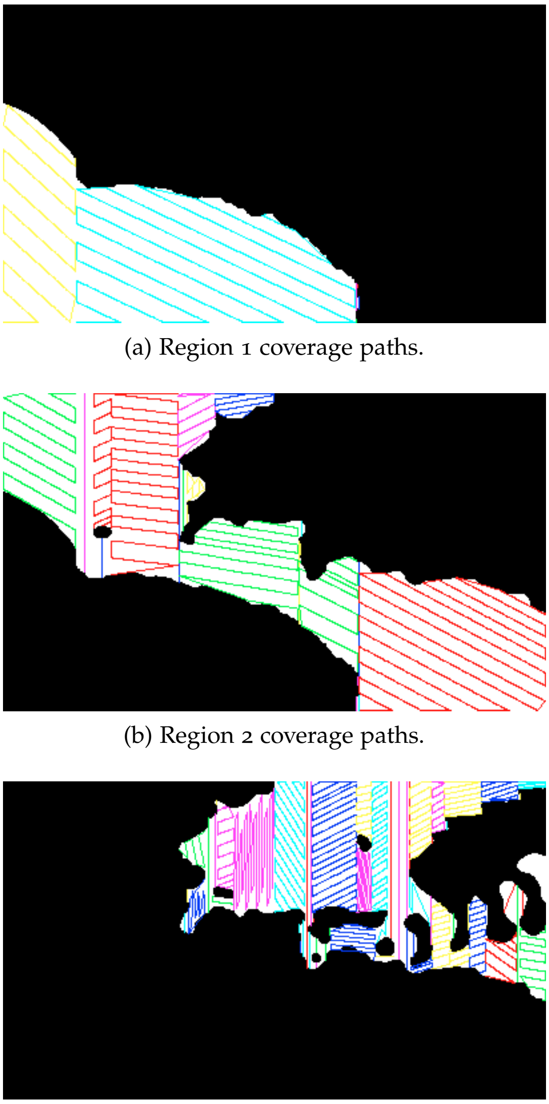
3.2.5.4 *Final Coverage Path*

The three steps we just described are applied to all the regions in the surface segmentation. Finally, we concatenate the individual coverage paths computed for each cell in the decompositions (Algorithm 1, line 9). The well-known "start-to-goal" path planner A* (see Section C.1 in the appendix for an introduction to A*) is used to compute a collision-free path from the last point of a cell coverage path to the first point of the next cell's coverage path. The same planner is used to connect the coverage paths of each segmented region. Figure 40 shows the final coverage path obtained after connecting all the coverage sub-paths in comparison to the standard boustrophedon coverage path.

3.3 RESULTS

To validate our coverage method, we compare our variable inter-lap spacing approach to the standard Morse-based boustrophedon decomposition for planar spaces. Seeking a fair comparison, we use the smallest possible sensor footprint in each cell to determine the inter-lap spacing (that is, the footprint obtained when the vehicle is located over the highest point on the seafloor surface).

Our comparison is twofold. On one hand, we quantitatively measure both paths in terms of length. On the other hand, we provide coverage density

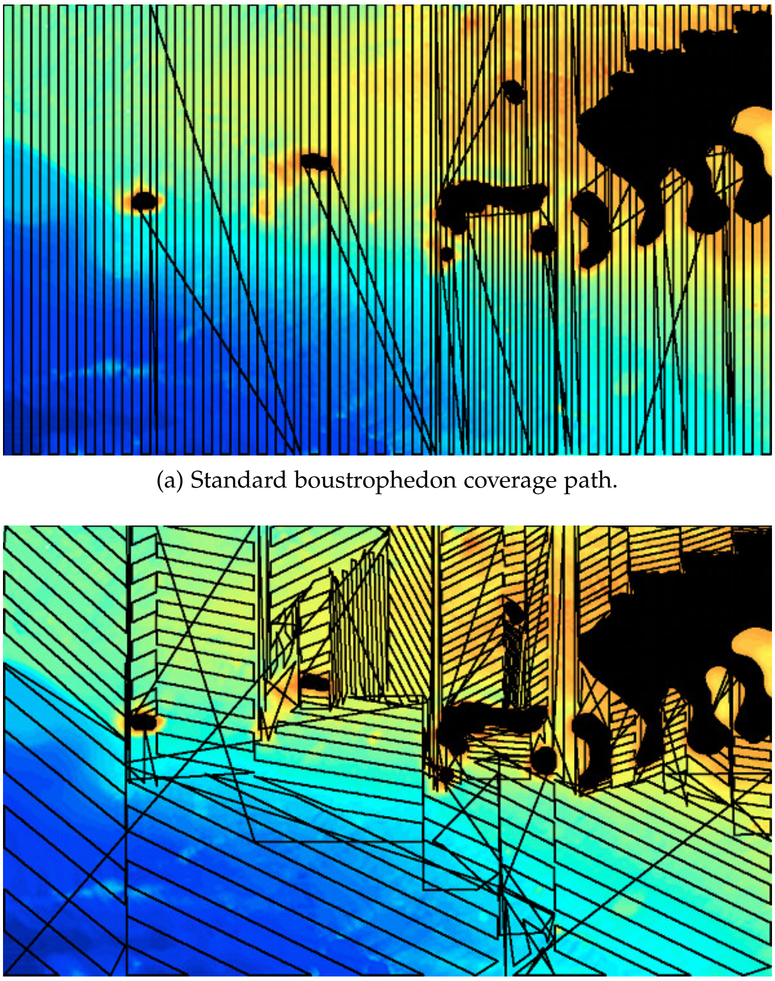


(a) Region 1 coverage paths.

(b) Region 2 coverage paths.

(c) Region 3 coverage paths.

Figure 39: Coverage paths for each cell of the cellular decompositions shown in Figure 37.



(a) Standard boustrophedon coverage path.

(b) Coverage path planned using our method.

Figure 40: Standard boustrophedon coverage path and final coverage path planned using our method for the workspace shown in Figure 35 overlapped on the target bathymetric map.

maps generated by following the paths computed by each method and taking into account the vehicle's sensor footprint in every point. In other words, the coverage density maps show, for a given path on the workspace, how many times has every point on the workspace been covered.

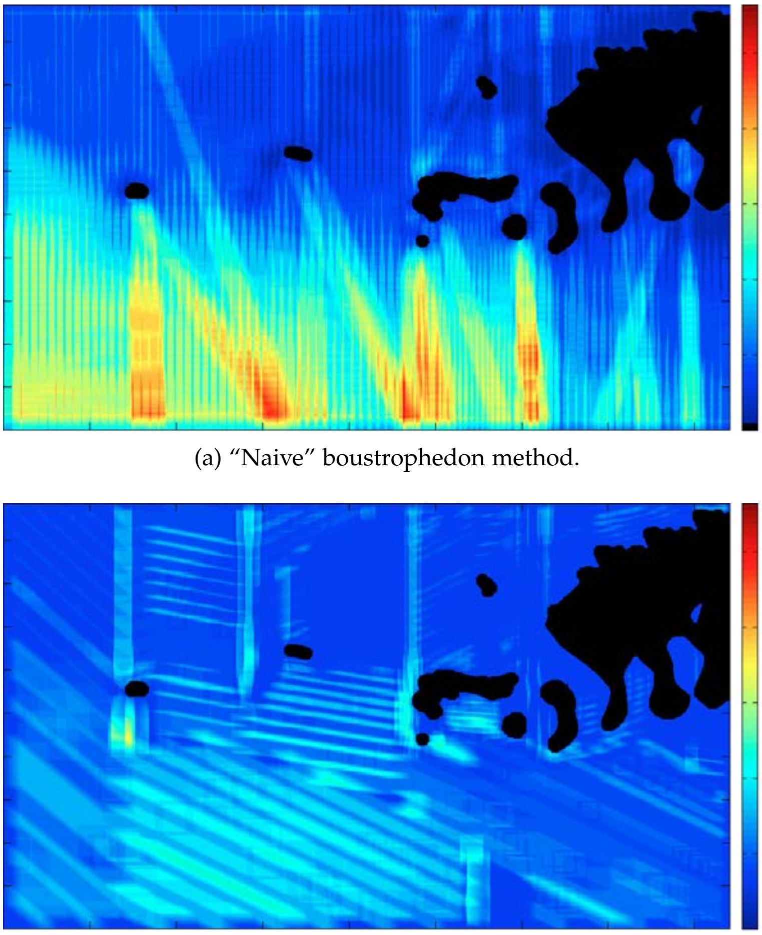
Table 2 shows the quantitative comparison of both methods. The vehicle navigates at depth $D = -8.5$ m and the target seafloor surface is segmented in $n = 3$ regions.

METHOD	PATH LENGTH
Constant inter-lap spacing	15846.08 m
Variable inter-lap spacing	10349.63 m

Table 2: 2D coverage path length comparison.

These results show that, for the real-world environment considered in this work, our method performs approximately 34% better than the boustrophedon approach for planar spaces in terms of path length.

Figure 41 shows the comparison of the coverage density maps obtained using the "naive" boustrophedon decomposition method and the novel approach proposed here. Ideally, each point in the target space should be covered once and hence be represented in a dark blue color. The coverage density map corresponding to the "naive" boustrophedon method (Figure 41a) shows a high amount of overlapping, denoted by yellow and red colors. By contrast, the coverage overlapping is highly reduced with our method, as denoted by the blue colors on Figure 41b.



(a) “Naive” boustrophedon method.

(b) Our method.

Figure 41: Coverage density map comparison.

4

UNCERTAINTY-DRIVEN 2D COVERAGE OF UNDERWATER ENVIRONMENTS

Bodies are for hookers and fat people!

— Bender

"Futurama", "A Head in the Polls" (1999)

AUV navigation techniques such as Terrain-Relative Navigation (TRN) or SLAM, which seek to match sensor measurements to an *a priori* map or to previously visited locations, respectively, are key to enhancing the autonomy of AUVs. Inherently, when using such techniques the uncertainty of the vehicle position estimates at every time step is critically affected by the characteristics of the terrain being traversed. For instance, high variability terrain provides discriminative measurements that lead to estimating the vehicle location with high probability, whereas a flat, feature-less terrain typically leads to a more uncertain, lower probability estimate. However, current survey path planning techniques for AUVs do not take this into account when planning the mission, leading to potentially very inaccurate data products (such as maps or 3D reconstructions). After detailing this problem in Section 4.1, this chapter presents a novel survey path planning technique for area coverage which minimizes the robot's position uncertainty along the planned path in Section 4.2. Reducing this uncertainty leads to a better position estimate, and hence to more accurate data products resulting from AUV surveys. Note that, rather than avoiding obstacles, which can be achieved with the method presented in Chapter 3, the focus in this chapter is to reduce the uncertainty associated to the planned coverage path. The proposed technique especially targets bathymetric mapping applications and respects application constraints such as the desire to survey in parallel tracks. We present results of our method in Section 4.3 using real-world datasets collected off the coasts of Catalonia (Spain), Greece and Australia and in real-world sea trials with the GIRONA 500 AUV in the Catalan coast. We evaluate the expected robot's position uncertainty along the planned paths with a particle filter and assess their associated mapping performance using a bathymetric mapping algorithm. Results show that our method offers benefits over a standard lawnmower-type path both in terms of position uncertainty and map quality.

4.1 INTRODUCTION

Due to the absence of a ubiquitous global localization system such as GPS in underwater environments, AUV navigation is confined to these three primary methods: (1) dead-reckoning, (2) time of flight acoustic navigation, and (3) geophysical navigation techniques. The most obvious and longest established technique is dead-reckoning, which consists in integrating vehicle velocity measurements from a sensor such as a Doppler Velocity Log (DVL) to obtain new position estimates. The problem with exclusive reliance on dead reckoning is that position error increases without bound as the distance traveled by the AUV increases. Since acoustic energy propagates well in the ocean, acoustic transponders can be deployed and used as beacons to guide the motion of an AUV. Long BaseLine (LBL), in which a vehicle triangulates its position from acoustic ranges within a network of transponders, and Ultra-Short BaseLine (USBL) acoustic navigation, in which a sonar array is employed to determine the range and bearing to the vehicle, are the primary time of flight acoustic navigation systems used today. However, these systems typically require mooring transponders to the ocean floor and the intervention of one or several surface vehicles, which is costly. For these reasons, advances in environmentally-based, geophysical navigation techniques such as TRN and SLAM has rendered this techniques popular and successful in AUV navigation. Terrain relative, or landmark relative navigation matches realtime sensing to a terrain or landmark map (topographic, magnetic, gravitational or of other geodetic data) to determine vehicle position. SLAM exploits sensing capabilities of robots to correct for accumulated odometric error by localizing the robot with respect to landmarks in the environment. These techniques eliminate the need for additional infrastructure and bound position error growth, which are prerequisites for truly autonomous navigation. Since these techniques rely on the robot's perception of the environment, the path followed by the robot during area coverage critically affects the quality of the position estimates.

For scientific data collected in the ocean to be useful, the location where the data were collected needs to be known accurately. A key task in underwater robotics clearly affected by this requirement is bathymetric mapping. Recall that bathymetric mapping is the measurement of underwater depth of lake, river or ocean floors. Bathymetric mapping supports safe navigation, helps protect and monitor marine areas of biological interest and is key to geology, archaeology and military applications, to name a few (Escartin et al., 2008; Bingham et al., 2010; Yoerger et al., 2000; Williams, 2010). Indeed, incorporating uncertainty when planning a survey path for an AUV mapping mission can lead to more accurate maps. This is because bathymetric mapping algorithms rely on the vehicle pose estimates during a mission to build the map. Therefore, the more accurate the vehicle pose estimates are the more accurate the resulting map will be.

While the general problem of path planning under uncertainty has been addressed in several research works, little attention has been paid to incorporating uncertainty when planning paths for area coverage. In fact, a considerable body of recent research has addressed the general problem of motion planning under uncertainty. However, these works address the "start-to-goal" path planning problem rather than area coverage. Some uncertainty-aware coverage algorithms were already discussed in Section 2.10, but they are limited to planar environments and do not account for the aforementioned particulars of underwater environments and bathymetric mapping.

Regarding path planning under uncertainty, many researchers propose extensions to the sampling-based Rapidly-exploring Random Tree (RRT) and Probabilistic Roadmap (PRM) path planning algorithms (LaValle and Kuffner, 2000; Kavraki et al., 1996) to handle uncertainty. (See Section C.2 in the appendix for an introduction to these algorithms.) The RRT extensions by Melchior and Simmons (2007) and Kewlani et al. (2009) explicitly handle uncertainty associated with terrain parameters (e.g., friction). By taking this uncertainty into account these planners try to avoid rough terrain. However, sensing or state observation uncertainty is not considered in these works. Huang and Gupta (2008) combined an extension to the RRT algorithm with a particle-based SLAM algorithm used to expand the tree. This integrated approach explicitly accounts for sensor, localization and environment uncertainty in the planning stage. In a later work, Huang and Gupta (2009) extended the PRM algorithm to compute paths of minimum probability of collision in the context of a robotic arm with uncertainty in its mobile base location. Generalizations of the RRT and PRM algorithms were proposed by Chakravorty and Kumar (2011) to obtain hybrid hierarchical motion planners that are robust to the motion uncertainty and to the uncertainty in the environment. However, the generalizations proposed in this work assume perfect knowledge about the state of the robot.

Other path planners focus on the uncertainty in the map of the environment to generate paths with minimum probability of collision with obstacles (Missiuro and Roy, 2006; Burns and Brock, 2006; Guibas et al., 2008; Nakhaei and Lamiraux, 2008).

Active perception algorithms increase robot localization efficacy by specifically considering the expected uncertainty of the localization algorithm while planning the next control input the robot will receive (Burgard et al., 1997; Roy et al., 1999; Valencia et al., 2012). Particularly related to the underwater domain are the next-best-view visual SLAM approach by Kim (2012) and the active localization technique using multibeam sonar by Fairfield and Wettergreen (2008). However, these algorithms select a control action to minimize uncertainty at the next time step, but do not optimize over an entire path.

Another class of approaches use Markov Decision Processes (MDPs) with motion uncertainty to define a global control policy over the entire robot's workspace, providing a connection between planning and control (Alterovitz et al., 2007). In order to also include sensing uncertainty, Partially Observable Markov Decision Processes (POMDPs) can be used (Kurniawati et al., 2008; Candido and Hutchinson, 2010; van den Berg et al., 2012). Although POMDPs are theoretically satisfactory, these approaches require the discretization of the environment, and as a result they suffer from scalability problems.

Some planners seek to maximize the probability of success or rather to minimize an expected cost by taking into account the sensing uncertainty (Pepy and Lambert, 2006; Gonzalez and Stentz, 2009; Prentice and Roy, 2009; Platt et al., 2010; Carrillo et al., 2012). However, these approaches, either implicitly or explicitly, assume that maximum likelihood measurements are received from the sensors. As a result, the probability distributions of the robot's state are only approximated. By considering the controller used to execute the path, van den Berg et al. (2011) could compute the true *a priori* probability distributions of the robot's state along its future path. By using these probability distributions, their method can select a path among several candidates such that maximizes the probability of arrival to the goal and at the same time minimizes the probability of collision.

Most approaches mentioned above assume the belief of the robot to be Gaussian, parametrized by mean and covariance. This restricts the type and degree of uncertainty that can be considered. Nonetheless, some recent non-Gaussian approaches have been proposed (Candido and Hutchinson, 2010; Platt et al., 2012), and so is the particle-filter-based approach we present in this work.

In relation to the graph structure we use in our method to represent parallel tracks, coverage path planning algorithms for environments that can be represented as a graph, such as a street or road network, were presented by Xu (2011), as described in Section 2.7. Nonetheless, uncertainty is not considered in these algorithms.

In summary, no uncertainty-aware path planning algorithms account for the application constraints of bathymetric mapping, such as the desire to survey in parallel tracks and to avoid turns on the target area to maximize the quality of sonar readings and to find useful loop-closures for a mapping algorithm. In fact, off-the-shelf AUV survey design tools typically plan a lawnmower-type path on the target area completely ignoring uncertainty. Optionally, one or more crossing tracks are then appended seeking to provide loop-closures for the mapping process. However, these crossing tracks are placed arbitrarily, again ignoring uncertainty.

Aiming to bridge this gap, we present an off-line survey path planning technique which takes into account the robot's motion and sensing uncertainty and seeks to minimize this uncertainty along the planned path. Multibeam sonars provide noisy, highly corrupted range measurements under pronounced orientation changes. Therefore, our method operates on a parallel track basis to confine turns to the boundaries of the target area. We compute the saliency for every point of an *a priori* bathymetry of the target area* using the saliency map (Itti et al., 1998), a tool borrowed from the Computer Vision community. Based on the saliency, we provide an algorithm to decide the order in which to trace the parallel tracks to minimize uncertainty while also keeping extra path length into account. Once the order is determined, the algorithm uses a particle filter with the *a priori* bathymetry and

* It is common in marine robotics applications to have prior knowledge of the target area in the form of low resolution bathymetry. The objective of a mapping mission is usually to obtain a more refined data product.

simulated multibeam sonar measurements to estimate the robot's position uncertainty. Whenever the uncertainty after a parallel track exceeds a user-provided threshold, a crossing track through a salient area is inserted, seeking to reduce uncertainty and to find useful loop-closures for mapping. This contrasts with traditional survey path planning methods, which concatenate arbitrarily placed crossing tracks to a lawnmower-type path.

Most existing uncertainty-aware path planning techniques assume the belief of the robot to be Gaussian. By contrast, our algorithm uses particle filtering methods with an *a priori* map of the target area and simulated multibeam sonar measurements to keep track of the robot's belief. This allows the consideration of arbitrary beliefs, given a sufficient number of particles. This capability is essential when performing bathymetry-based localization, which often leads to multi-modal posteriors.

We test our algorithm in simulation using real-world datasets collected off the Formigues islands in Catalonia (Spain); the Santorini island in Greece; Tasmania in Australia and in real-world sea trials with the GIRONA 500 AUV off the Costa Brava in Sant Feliu de Guíxols, Girona, Catalonia. We calculate the position uncertainty along the planned paths using terrain-aided particle filter localization and compare them to standard lawnmower-type paths. Additionally, we compare the mapping performance of a path planned using our method to a standard survey path on one of the datasets using a bathymetric mapping algorithm. Results show that our method offers benefits in terms of position uncertainty and map quality over a standard lawnmower-type path.

4.2 UNCERTAINTY-DRIVEN SURVEY PATH PLANNING

As stated earlier, our method deals with the application constraints of surveying the target area in parallel tracks and avoiding turns in the target area in order to maximize the quality of the sonar readings. We therefore operate on a parallel track basis by constructing a graph representing the parallel tracks required to cover the target area, which we call the "coverage graph". Then, we plan a survey path in the two following steps:

1. Find the best possible order in which to cover the parallel track edges of the coverage graph which minimizes the overall uncertainty along the path;

2. Insert crossing track edges in the path found in the first step if, after tracing a parallel track, the uncertainty surpasses a given threshold.

Finding the order in which to trace the parallel track edges raises two important concerns that need to be addressed. First, note that finding the optimal coverage path implies dealing with $n!$ candidate solutions, for a coverage graph with n edges, which is an intractable problem (Ausiello et al., 1999). Therefore, finding the optimal solution is computationally infeasible and some heuristic must be applied in order to find a good approximation in reasonable time. Second, commonly used heuristics do not apply to this problem due to the expansion and contraction of uncertainty (that is, the uncertainty throughout the path is non-monotonic). We address these concerns by determining the parallel track order based on the saliency of the terrain, which can be computed quickly. Then, we keep track of the robot's belief uncertainty along the determined path using a particle filter. When the estimated uncertainty surpasses a user-provided threshold, a crossing track that visits salient locations of the terrain is inserted, seeking to reduce the uncertainty.

Next, we first describe the construction of the coverage graph in Section 4.2.1. Then, we discuss how the saliency map is used to compute the average saliency associated to each parallel track and to determine salient locations upon which to trace crossing tracks (Section 4.2.2). The vehicle and measurement models and the particle filter algorithm used to keep track of the robot's position uncertainty are described in Section 4.2.3 and Section 4.2.4, respectively. Finally, we describe our proposed survey path planning algorithm (Section 4.2.5), which builds upon the coverage graph, the saliency map, the models and the particle filter.

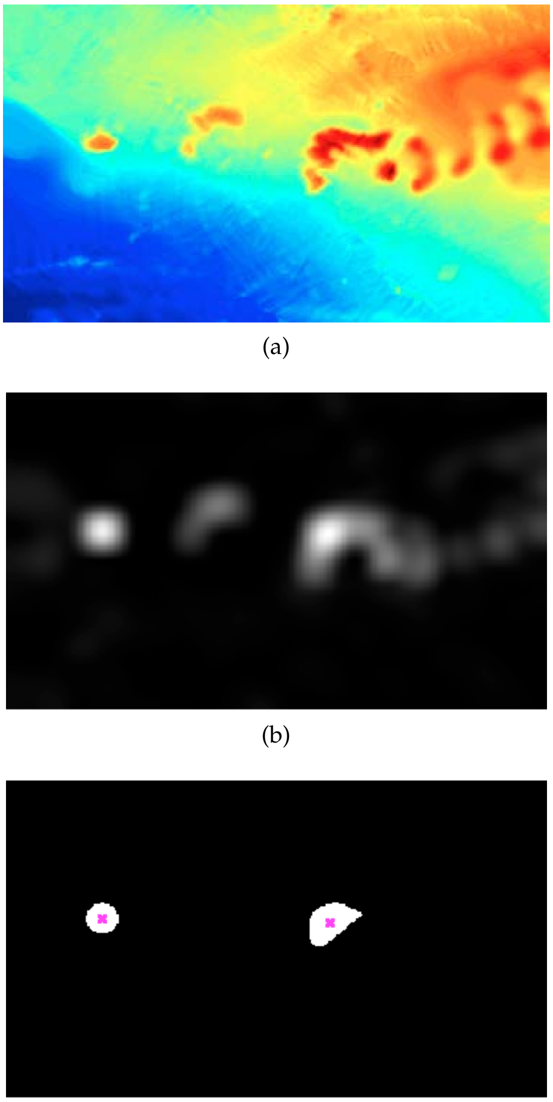
4.2.1 *Coverage Graph Construction*

We construct a coverage graph consisting of equally spaced, parallel edges (tracks) with vertices lying on their endpoints. The vertices on each side of the parallel edges are then linked with the other vertices on the same side by vertical edges, forming a connected graph. The altitude from the seafloor together with the sonar swath aperture determine the inter-track spacing. The coverage graphs on the datasets we later use to test our algorithm can be seen in Section 4.3 below.

4.2.2 *Saliency Calculation*

When using the terrain's elevation profile for localization and/or mapping, we observe that profile measurements are less uncertain where the terrain is more salient. Based on this observation, in this work we propose to use the saliency map (Itti et al., 1998) over the *a priori* bathymetry as an estimation of the effect of the terrain on the robot's belief uncertainty. The saliency map assigns a saliency score to every pixel in an image (the bathymetry in this case).

We use the saliency map in two respects. First, for each parallel track, we compute the average saliency score in the region determined by the bathymetry sonar footprint and the parallel track itself. Second, we determine key salient points through which our algorithm will later trace crossing tracks. We select regions surpassing a user-provided saliency threshold δ on the saliency map and take the weighted centroid of each high-saliency region as a key point. As far as at least one region is identified, our method can deal with many high-saliency regions in the map, since always the closest to the current robot position is visited by the algorithm. Figure 42 shows the saliency map, its corresponding segmentation ($\delta = 0.5$) and key salient points for a real-world bathymetric dataset.



(a)

(b)

(c)

Figure 42: Saliency computation for a bathymetric map (a): saliency map (b) and saliency map segmentation with $\delta = 0.5$ (c). The weighted centroids of each segmented region are marked with an "x".

4.2.3 *Vehicle and Measurement Models*

Given a bathymetric map, $\mathcal{B}$, and a path to be analyzed defined as a sequence of K 3-dimensional waypoints, $\Pi = [x_0, y_0, z_0]^\top, [x_1, y_1, z_1]^\top, ..., [x_K, y_K, z_K]^\top$, we define a vehicle model and a measurement model as follows.

4.2.3.1 *Vehicle Model*

The state vector s_t of the vehicle model is the 3 DOFs vehicle position at path step k:

$$s_k = [x_k, y_k, z_k]^\top. \tag{5}$$

This state vector is updated according to a constant-velocity vehicle model

$$s_k = f(s_{k-1}, u_k) + \mathcal{N}(0, \sigma_f), \tag{6}$$

$$f(s_{k-1}, u_k) = s_{k-1} + u_k, \tag{7}$$

where σ_f is additive Gaussian noise and u_k is the control vector at step k, in this case determined by the last and current path steps:

$$u_k = \Pi_k - \Pi_{k-1}. \tag{8}$$

4.2.3.2 *Measurement Model*

We model a typical multibeam sonar providing an array of beams spread in a downward-facing swath perpendicular to the vehicle's direction of travel. At path step k, the vector of range measurements corresponding to the beams is given by $\mathbf{r}_k = [r_{k,1}, \dots, r_{k,N}]^\top$ and the measurement model for each beam i is given by

$$r_{k,i} = \mathcal{B}_i(x, y) - d_k + \mathcal{N}(0, \sigma_r), \forall 1 \leqslant i \leqslant N, \tag{9}$$

where $\mathcal{B}_i(x, y)$ is the map elevation at the point where the sonar beam i intersects the map surface, N is the number of beams, d_k is the vehicle's depth and σ_r is measurement noise which is assumed to be Gaussian. We simulate the sonar beams by shooting multiple rays against the map and computing their intersections. In order to speed-up the ray intersection computation, we use an axis-aligned bounding boxes data structure.

4.2.4 *Particle Filter*

We use a particle filter based on the the sequential importance resampling filter (Ristic et al., 2004) to estimate the position and uncertainty of the robot along a given path Π. The distribution on the state s_k is approximated by the weighted set of M particles $s_k^{(i)}, w_k^{(i)}, i \in [1, M]$ as

$$p(s_k|\mathbf{R}_k) = \sum_{i=1}^{M} w_k^{(i)} \delta_{s_k^{(i)}}(s_k), \tag{10}$$

where $\mathbf{R}_k = \mathbf{r}_{0:k}$. The particle weights are recursively updated according to the equations (Ristic et al., 2004)

$$\hat{w}_k^{(i)} = w_{k-1}^{(i)} \frac{p(s_k^{(i)}|s_{k-1}^{(i)})g(\mathbf{r}_k|s_k^{(i)})}{q(s_k^{(i)}|\mathbf{s}_{k-1}^{(i)}, \mathbf{R}_k)}, \tag{11}$$

$$w_k^{(i)} = \frac{\hat{w}_k^{(i)}}{\sum_{i=1}^{M} \hat{w}_k^{(i)}} \tag{12}$$

where the prior $p(s_k^{(i)}|s_{k-1}^{(i)})$ is given by Equation (6), $q(\cdot|\cdot)$ represents the proposal distribution and $g(\cdot|\cdot)$ represents the likelihood function. Here, we use the prior distribution as the proposal distribution which results in simplification of the weight update. The particles are sampled according to

$$s_k^{(i)} \sim \mathcal{N}(f(s_{k-1}^{(i)}, u_k), \sigma_f^2 \mathbb{I}_3), \quad i \in [1, M]. \tag{13}$$

The likelihood function is given by

$$g(\mathbf{r}_k|s_k^{(i)}) = \mathcal{N}(\mathbf{r}_k; \hat{\mathbf{r}}_k^{(i)}, \sigma_r^2 \mathbb{I}_N) \tag{14}$$

where $\hat{\mathbf{r}}_k^{(i)}$ is the vector of expected elevations:

$$\hat{\mathbf{r}}_k^{(i)} = [\hat{r}_{k,1}^{(i)}, \ldots, \hat{r}_{k,N}^{(i)}], \tag{15}$$

$$\hat{r}_{k,i} = \mathcal{B}_i(x_k^{(i)}, y_k^{(i)}) - d_k. \tag{16}$$

Resampling with replacement is carried out at each time to limit the degeneracy of the particles.

In this work, we are interested only in the uncertainty of the robot's belief rather than in the position estimate. We estimate the uncertainty by evaluating the trace of the sample covariance of the distribution $p(s_k|\mathbf{R}_k)$ as

$$\text{tr}(\Sigma_k) = \text{tr}(\frac{1}{M-1} \sum_{i=1}^{M} (s_k^{(i)} - \bar{s}_k)(s_k^{(i)} - \bar{s}_k)^T). \tag{17}$$

4.2.5 *Survey Path Planning Algorithm*

Our survey path planning algorithm addresses the aforementioned intractability and application constraints with a saliency-based heuristic. It first sorts the n parallel track edges in two groups of $\frac{n}{2}$ or $\frac{n}{2} - 1$ edges each: one with the highest saliency edges and one with lowest saliency edges (the edges being scored as described in Section 4.2.2). Then it alternatively selects one edge from each group to construct the path. The idea behind this heuristic is that the uncertainty growth incurred by a low saliency edge will be compensated by the high saliency of the next edge, avoiding high uncertainty peaks. To take also path length into account, the closest edge on the next group is selected at every step.

To further bound the uncertainty, the algorithm estimates the robot's position uncertainty by tracing the parallel tracks in the order determined by the heuristic using the particle filter. If, after a track, the uncertainty surpasses a user-provided threshold α, a crossing track through the closest key salient point is inserted before continuing on the next parallel track.

It is worth noticing that our heuristic, inherently, does not guarantee an optimal path with respect to uncertainty. However, it tackles the intractability of the planning problem by producing a low uncertainty solution, as demonstrated by our experimental results (see Section 4.3 below). On the other hand, small values of the uncertainty threshold α can lead to longer paths due to the addition of multiple crossing tracks seeking to reduce the uncertainty. Nonetheless, our results show that even reasonably restrictive values of α do not lengthen the resulting path significantly. The choice of α strongly depends on the uncertainty tolerance of the target application. Evaluation of the effect of several values of α on the path produced by the algorithm can be used to determine a good fit for the application at hand.

The survey planning algorithm is detailed in Algorithm 2. The algorithm takes as input a coverage graph as a list of edges and an *a priori* bathymetry and the uncertainty and saliency thresholds as parameters. The parallel track edges are classified in two groups according to their average saliency using the saliency map in line 2 (a low saliency group, E_L, and a high saliency group, E_H). Key salient points on the map are identified in line 3. The particle filter's particles, P, and weights, W, are initialized according to an initial distribution (line 4). The coverage path to be constructed, $\hat{\Pi}$, is initialized in line 5. The algorithm runs while there are edges to be processed in any of the groups. In line 8 the closest edge from each group is alternatively selected according to the h flag and removed from the group. The selected edge is appended to the path (line 10) and analyzed using the particle filter (line 11). If the uncertainty computed by the particle filter, $\mathrm{tr}(\Sigma_k)$, surpasses the α threshold, a crossing track through the closest key salient point and toward the next edge to be covered is concatenated to the path (lines 12-15). Note that, in line 13, the appropriate edge is accessed, but it is not removed from the group. This is in contrast with line 8, where an edge is selected and removed from the group. Lastly, the planned coverage path, $\hat{\Pi}$, is given as output in line 16.

4.3 RESULTS

We show the effectiveness of our proposed method in regions of interest of four different real-world bathymetric datasets collected at sea. For each dataset, we generate a survey path using our method and compare its performance to a standard lawnmower-type path. We perform the comparison using both trajectories executed in simulation and in a real-world experiment conducted at sea with the GIRONA 500 AUV (see Chapter A in the Appendix for an introduction to the GIRONA 500 AUV). To quantitatively evaluate the effect of the trajectories on the robot's belief uncertainty, we use the particle filter algorithm presented above to keep track of the sample covariance of the particles as given in Equation 17. Additionally, on one of the datasets, we compare the effect on mapping performance of the uncertainty-

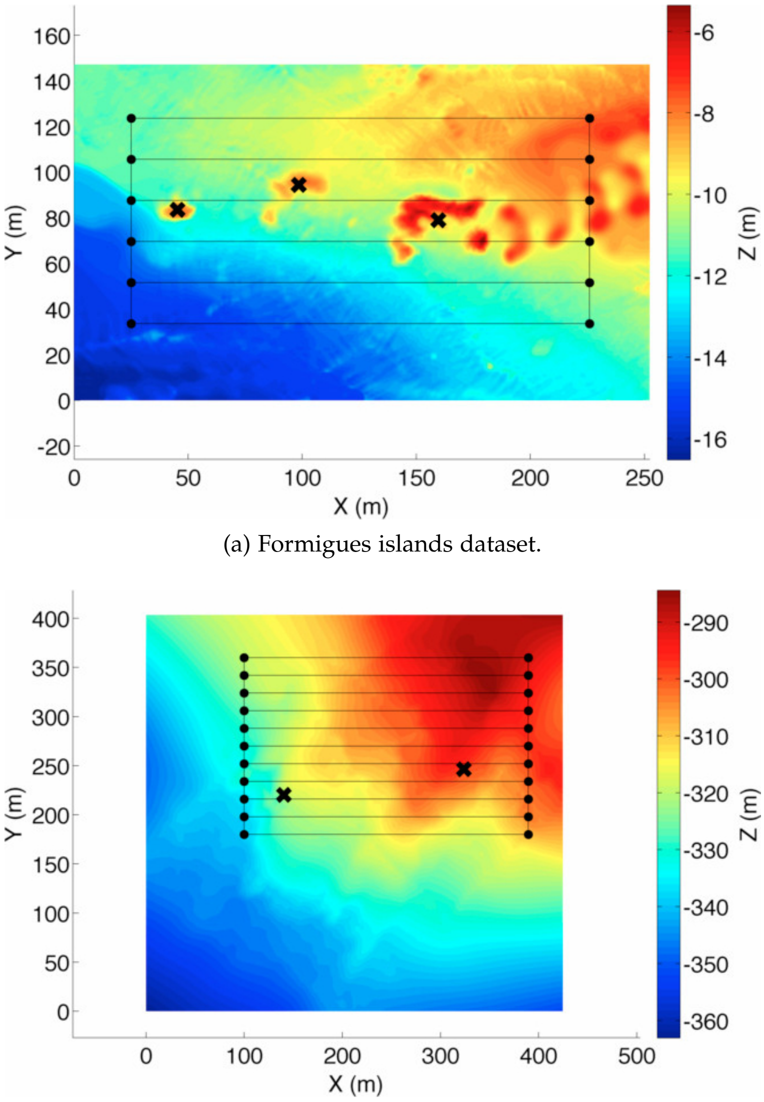
Algorithm 2: Uncertainty-driven Survey Path Planning

```
Input: List of parallel track edges in the coverage graph, $E$.
A priori bathymetry, $\mathcal{B}$.
Parameters: Uncertainty and saliency thresholds: $\alpha$, $\delta$.
$\mathcal{S} \leftarrow$ SaliencyMap($\mathcal{B}$)
$(E_L, E_H) \leftarrow$ ClassifyEdges($E$,$\mathcal{S}$)
$\Lambda \leftarrow$ KeySalientPoints($\mathcal{S}$,$\delta$)
$(P, W) \leftarrow$ InitParticleFilter()
$\hat{\Pi} \leftarrow \emptyset$
$h \leftarrow$ true
while not $E_L$.empty() and $E_H$.empty() do
    $e \leftarrow$ PopClosestNextEdge($E_L$,$E_H$,$h$)
    $h \leftarrow$ not $h$
    $\hat{\Pi}$.append($e$)
    $(P, W, \Sigma_k) \leftarrow$ ParticleFilter($e$,$P$,$W$)
    if $\mathrm{tr}(\Sigma_k) > \alpha$ then
        $n \leftarrow$ GetClosestNextEdge($E_L$,$E_H$,$h$)
        $c \leftarrow$ BuildCrossingTrack($e$,$n$,$\Lambda$)
        $\hat{\Pi}$.append($c$)
return $\hat{\Pi}$
```

driven paths planned using our method to a standard path. Next, we first describe the four bathymetric datasets we use to validate our method (Section 4.3.1). We then perform simulation experiments evaluating uncertainty and mapping performance with three of the four datasets. Lastly, using the remaining dataset, we evaluate our method in terms of uncertainty with data obtained in real-world AUV surveys.

4.3.1 *Bathymetric Datasets*

The four bathymetric datasets we use to evaluate our proposed planning method were collected: (1) near the Formigues islands off the Costa Brava, Girona, Catalonia (Spain), in the Mediterranean Sea; (2) in the Santorini caldera, Greece, in the Aegean Sea; (3) off the island of Tasmania, Australia, in the Pacific Ocean; and (4) at "El Jardí" and "l'Amarrador" area nearby the harbor of Sant Feliu de Guíxols in the Costa Brava, Girona, Catalonia (Spain), in the Mediterranean Sea. These four bathymetric datasets are shown in Figure 43 with their coverage graphs and their key salient points, obtained using a saliency threshold $\delta = 0.5$. We found out, empirically, that this threshold value successfully identifies the salient locations on all the datasets used in these experiments. The parallel tracks of the coverage graph keep a constant 6 m altitude from the bottom in all datasets except for "El Jardí" and "l'Amarrador" area, where they keep a constant 5 m depth. The inter-track spacing is determined by the footprint of a down-looking 120° swath aperture multibeam sonar modeled after the Imagenex Delta T device the real-world GIRONA 500 AUV is equipped with. The Formigues islands dataset (Figure 43a) has been introduced already in Section 3.2.1. A brief description of the remaining three datasets is given below. The Formigues, Santorini and Tasmania datasets are used in the simulation experiments. The "El Jardí" and "l'Amarrador" dataset is used in the real-world experiments.



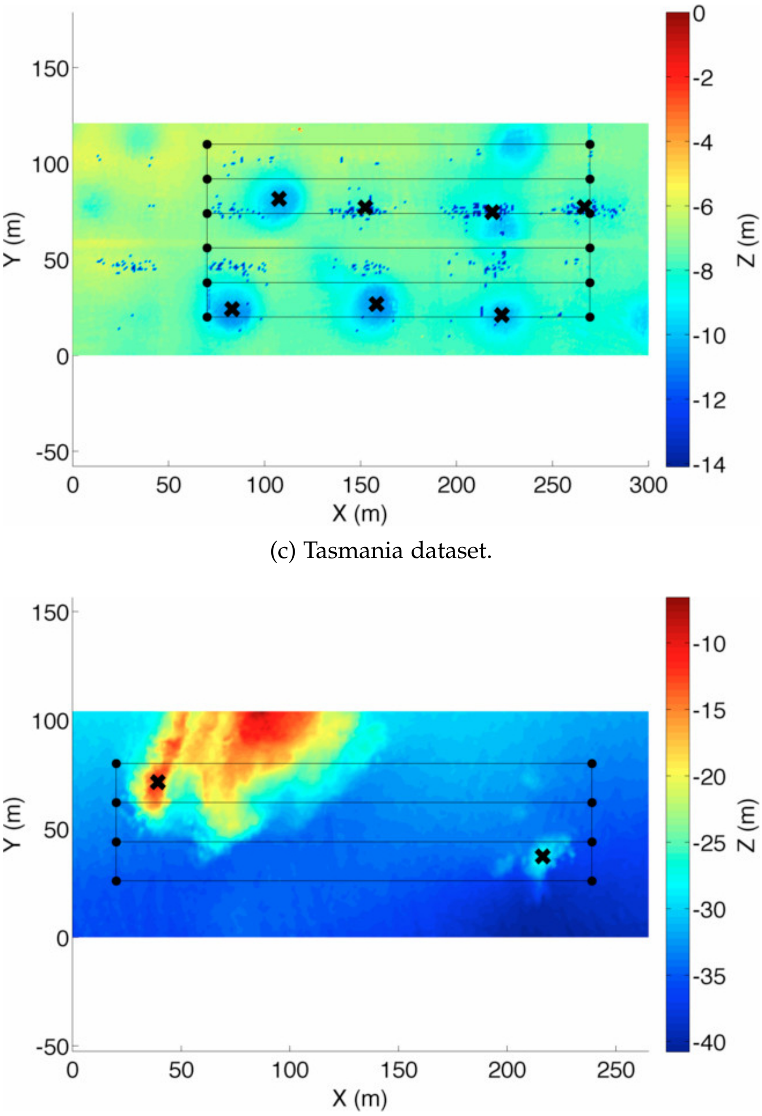
(a) Formigues islands dataset.

(b) Santorini dataset.

Figure 43: Bathymetric datasets used to test our algorithm with their corresponding coverage graph and key salient points (marked as "x").

4.3.1.1 *The Santorini Caldera Dataset*

The Santorini caldera bathymetric map (Figure 43b) was obtained during the Caldera 2012 sea trials, which took place from July 13th to July 23rd that year within the caldera of the Santorini island in Greece. These sea trials were part of a joint project involving an international and multidisciplinary team, formed by the Institut de Physique du Globe de Paris (France), the University of Girona (Catalonia, Spain), the Woods Hole Oceanographic Institute



(c) Tasmania dataset.

(d) “El Jardí” and “l’Amarrador” dataset.

Figure 43: Bathymetric datasets used to test our algorithm with their corresponding coverage graph and key salient points (marked as “x”).

(USA) and the University of Athens with infrastructure support from the Hellenic Centre of Marine Research (Greece). During the Caldera 2012 trials, the GIRONA 500 AUV was used for characterization of hydrothermal activity within the caldera via mapping and for collection of other oceanographic data. In one of the missions, GIRONA 500 AUV gathered this bathymetric dataset in an area of high volcanic activity in the vicinity of the caldera. The vehicle mapped the area navigating at a safe altitude of 15 m from the bottom. The mapped area is 427.5 m by 406.5 m, with depths ranging from 284 m to 363 m.

4.3.1.2 *The Tasmania Pockmarks Dataset*

The Tasmania dataset (Barkby et al., 2012), shown in Figure 43c, was kindly provided by the School of Aerospace Mechanical and Mechatronic Engineering of the University of Sydney. The experiment was performed in 2009 with the Sirius AUV (Williams et al., 2009) off the coasts of Tasmania island, in Australia. The mapped area features several geological pockmarks, which are craters in the seabed caused by fluids (gas and liquids) erupting and streaming through the sediments.

4.3.1.3 *"El Jardí" and "l'Amarrador" Dataset*

This dataset maps "El Jardí" and "l'Amarrador", two popular diving sites located approximately 1 Km off the harbor of Sant Feliu de Guíxols in the Costa Brava of Girona, Catalonia (Spain). "El Jardí" (featured in the top-left of Figure 43d) is a submerged rock formation emanating from the coastal cliffs and extending throughout an area of 150 m by 100 m, approximately. "l'Amarrador" (featured in the bottom-right of Figure 43d) consists of an isolated underwater boulder approximately 12 m high, rising from 40 m depth up to 28 m. Both sites feature a rich diversity of marine life. Although we had performed bathymetric surveys in the area using GIRONA 500, we used a bathymetric map obtained from a surface vessel instead. This way the data could be georeferenced via GPS, which is important since this is the dataset we use in our real-world experiments at sea. As in the Formigues dataset, a vessel equipped with the very same multibeam sonar as GIRONA 500, GPS and AHRS was used to survey the area.

4.3.2 *Simulation Experiments*

Next, we present simulation experiments conducted using the Formigues, Santorini and Tasmania datasets. We first present an evaluation in terms of position uncertainty on all three datasets and then, in addition, an evaluation of mapping performance using the Formigues dataset.

4.3.2.1 *Position Uncertainty Results*

We run our algorithm on each dataset, using the coverage graphs depicted in Figure 43, with an uncertainty threshold $\alpha = 20$. We compare the paths planned using our method with a standard lawnmower-type path, the construction of which was discussed in Section 2.2 and is well-documented in the literature (Lumelsky et al., 1990; Acar et al., 2002). We append the same number of equally-spaced crossing tracks to the standard survey path as crossing tracks are inserted by our algorithm. Figures 44, 45 and 46 show the standard and uncertainty-driven coverage paths for the Formigues, Santorini and Tasmania datasets, respectively.

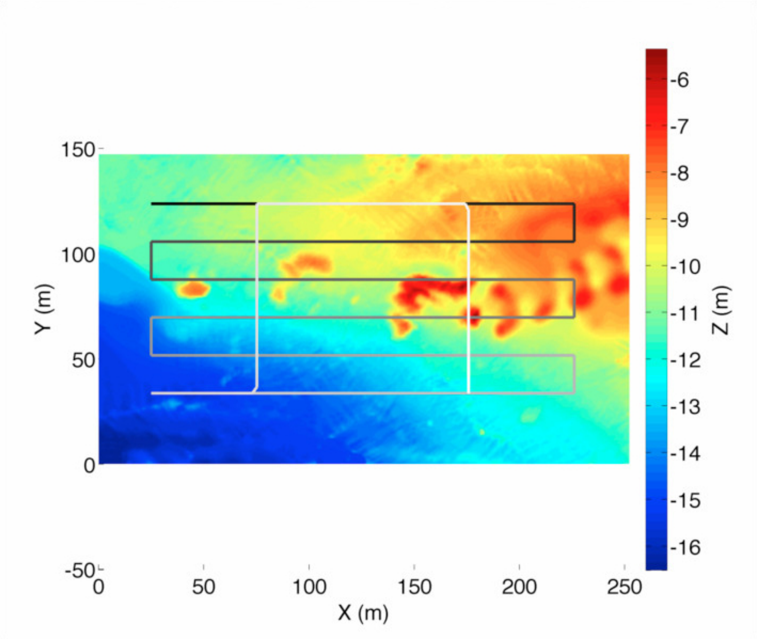
Figure 47 shows the belief uncertainty, $\mathrm{tr}(\Sigma_k)$, and its mean vs. path length for a standard lawnmower-type path and a path planned using our proposed method on each dataset. It can be observed that our method produces a path with a lower average uncertainty than the standard lawnmower-type path. We also note that our method tends to avoid the high uncertainty peaks associated to the standard lawnmower-type paths. Regarding path length, our crossing track insertion procedure lengthens the path due to the requirement of visiting a (potentially distant) key salient point. However, the extra length is compensated by the enhancement in navigation quality.

4.3.2.2 *Mapping Results*

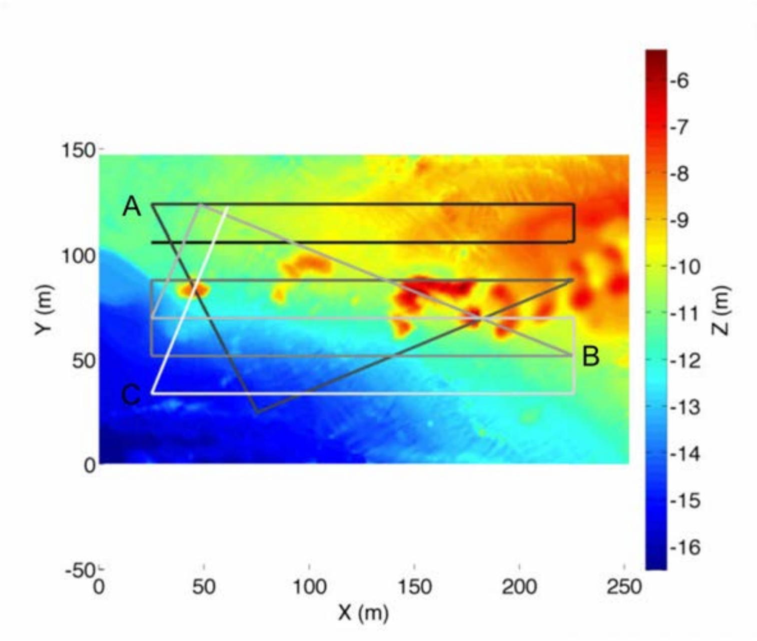
We next compare the mapping performance associated to a path planned using our uncertainty-driven method to the mapping performance of a standard survey path on the Formigues islands dataset.

We do so by executing the paths in simulation using UWSim (Prats et al., 2012a), an open source tool for visualization and high-fidelity simulation of underwater robotic missions. UWSim allows us to use models of the GIRONA 500 AUV, a multi beam sonar and a navigation sensor suite to collect a complete bathymetric dataset in simulation. Moreover, UWSim is seamlessly integrated with GIRONA 500's control architecture (Palomeras et al., 2012), which means that the very same software that runs onboard the AUV in real missions is used in our simulations. See Chapter A in the Appendix for more details about UWSim and GIRONA 500's control architecture.

After executing the paths in simulation we apply a mapping algorithm to the multibeam sonar data collected along the paths. Once the maps are constructed we assess their quality and compare the results obtained using each type of path. We use the mapping algorithm by Zandara et al. (2013) to build the bathymetric map and assess the map error. The application of this mapping algorithm to the data collected in simulation was carried out by colleague Albert Palomer at CIRS, coauthor of the aforementioned paper. The outcomes of the mapping algorithm are presented here for the purpose of demonstrating the effectiveness of the proposed uncertainty-driven coverage path planning approach. The map error is calculated as the standard deviation of the map points inside each and every cell of the 3-dimensional grid forming the map. The maps obtained with our method and with the standard survey path are shown in Figure 48, whereas Figure 49 shows their associated map errors. The path planned using our method produces a higher quality map, with an average map error of 0.0893 m in contrast to the average error of 0.1136 m produced by the standard survey path.

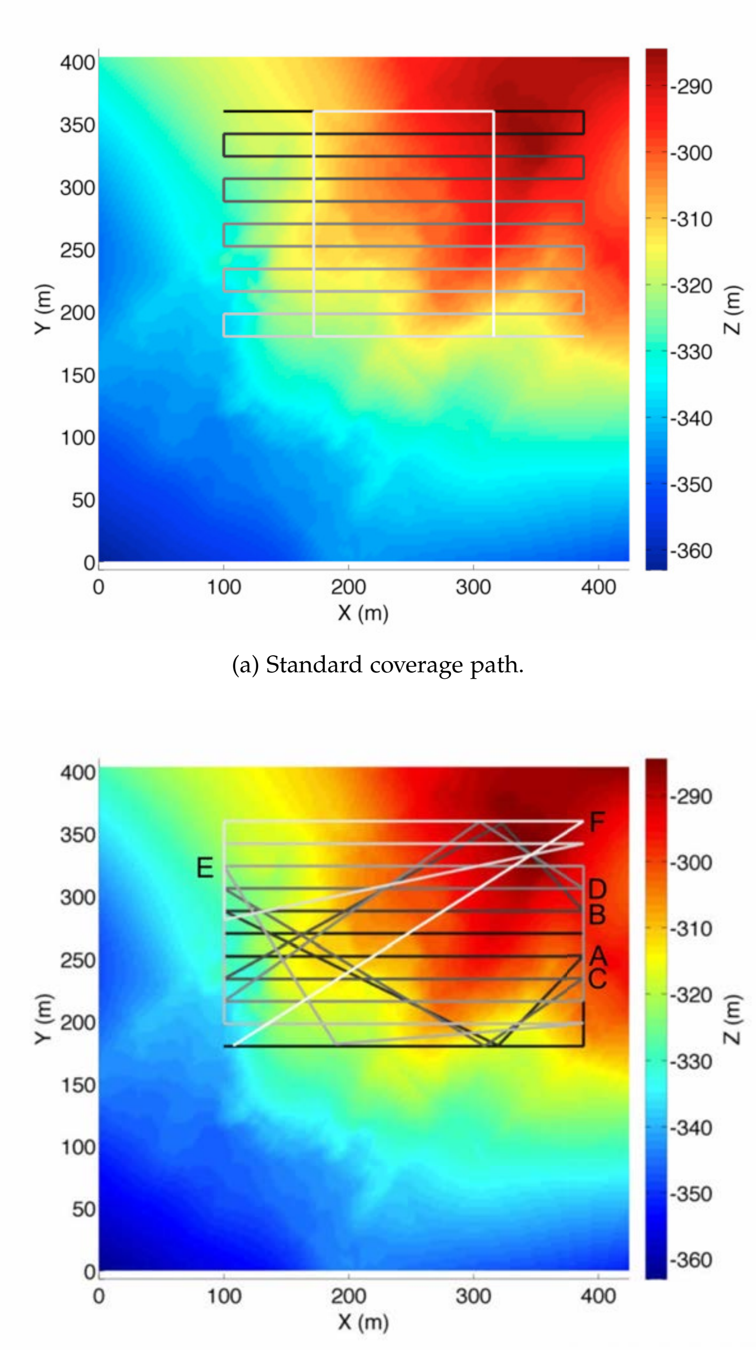


(a) Standard coverage path.



(b) Uncertainty-driven coverage path.

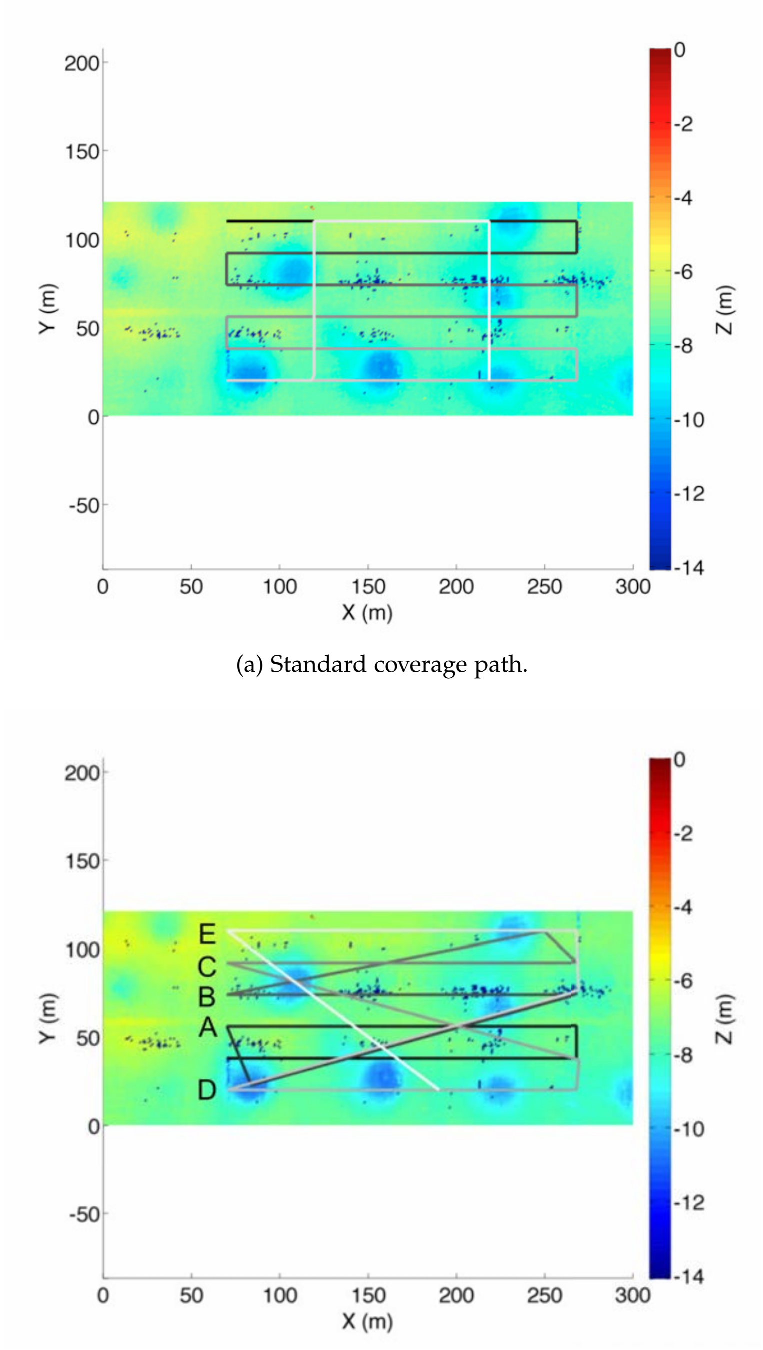
Figure 44: Distance-colored (from black to white) standard coverage path and uncertainty-driven coverage path planned for the Formigues islands dataset. $A, \ldots, C$ indicate, sequentially, the locations where the uncertainty surpasses the threshold and a crossing track is added to the path.



(a) Standard coverage path.

(b) Uncertainty-driven coverage path.

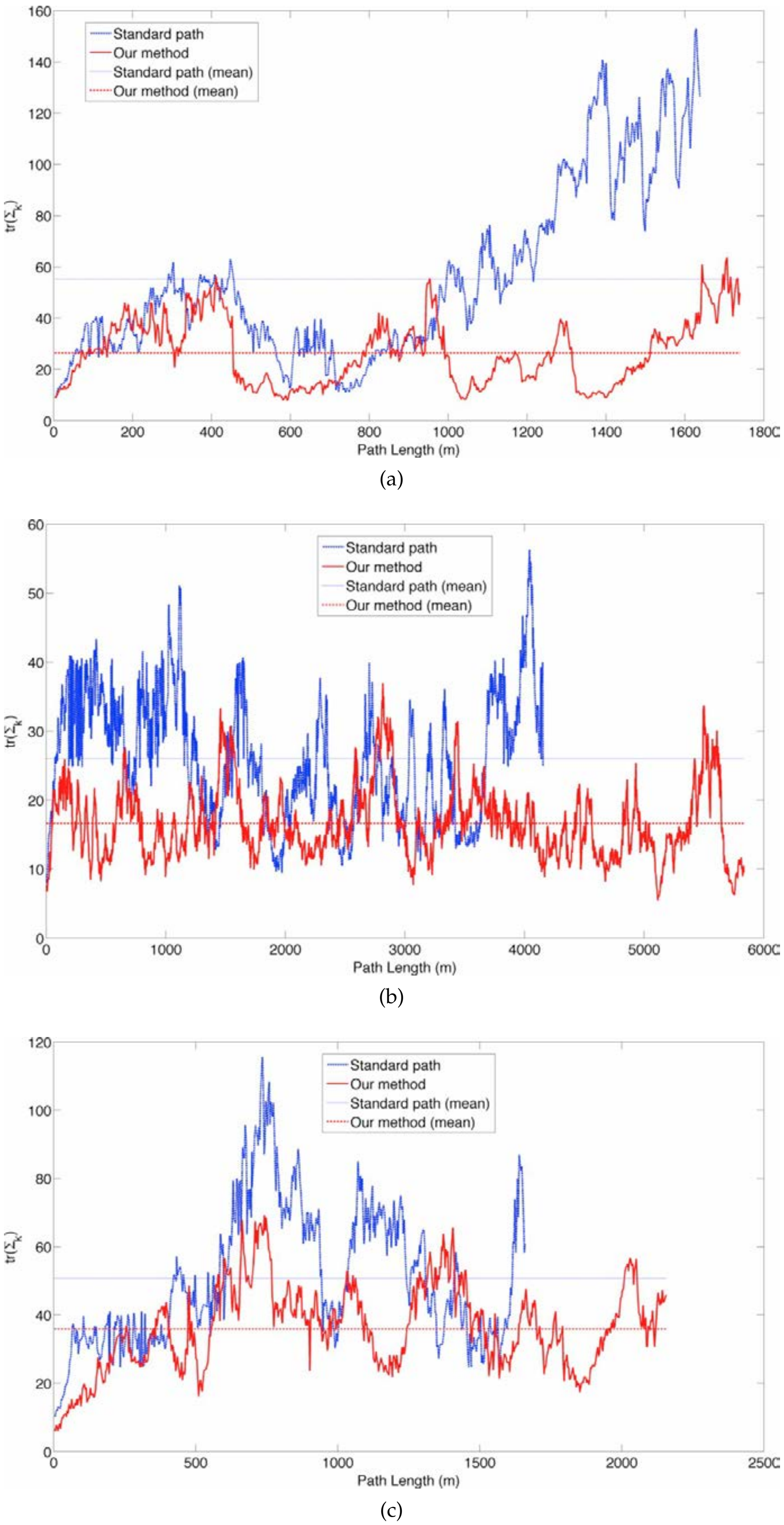
Figure 45: Distance-colored (from black to white) standard coverage path and uncertainty-driven coverage path planned for the Santorini caldera dataset. $A, \ldots, F$ indicate, sequentially, the locations where the uncertainty surpasses the threshold and a crossing track is added to the path.



(a) Standard coverage path.

(b) Uncertainty-driven coverage path.

Figure 46: Distance-colored (from black to white) standard coverage path and uncertainty-driven coverage path planned for the Tasmania pockmarks dataset. $A, \ldots, E$ indicate, sequentially, the locations where the uncertainty surpasses the threshold and a crossing track is added to the path.

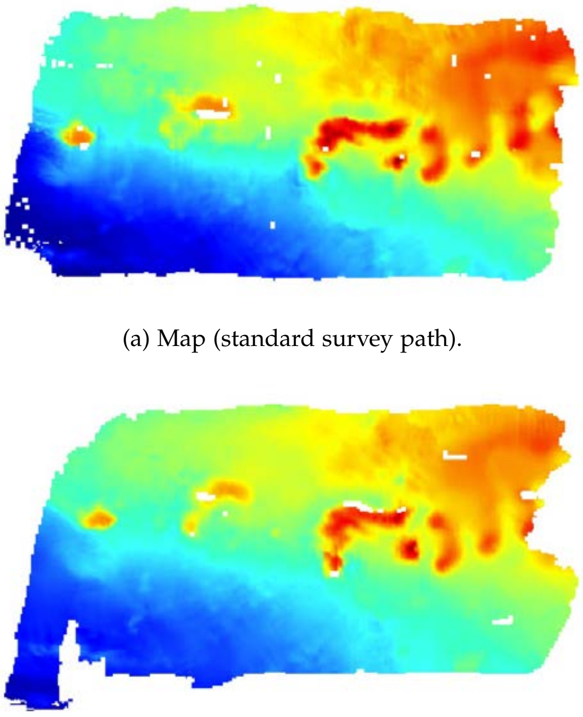


(a)

(b)

(c)

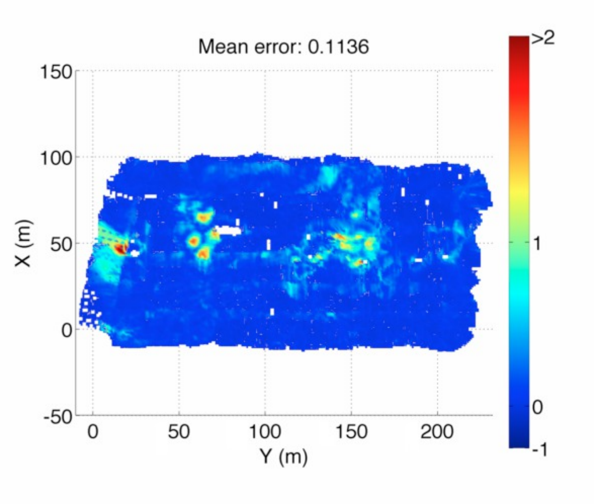
Figure 47: Belief uncertainty, $\text{tr}(\Sigma_k)$, and mean vs. path length for a standard lawnmower-type path and a path planned using our proposed method on each dataset: (a) Formigues islands dataset, (b) Santorini Caldera dataset, (c) Tasmania pockmarks dataset.



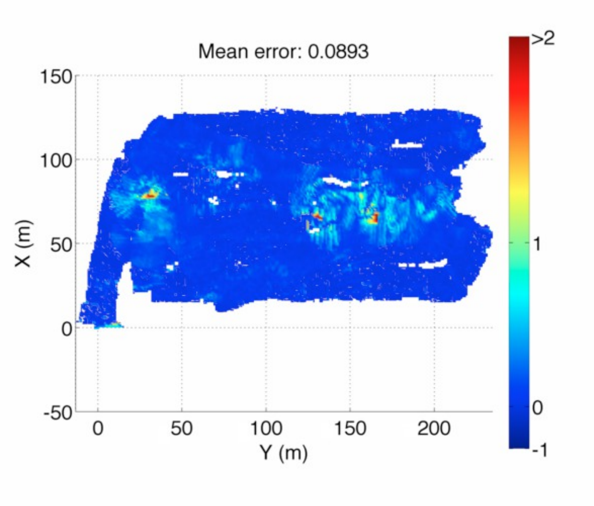
(a) Map (standard survey path).

(b) Map (our method).

Figure 48: Mapping results for the Formigues islands dataset: map obtained with a standard lawnmower-type path (a) and map obtained with the path planned using our method (b).



(a) Mapping error (standard survey path).



(b) Mapping error (our method).

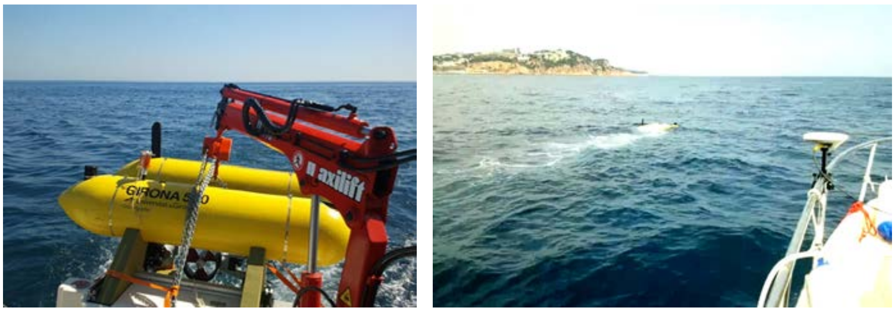
Figure 49: Mapping results for the Formigues islands dataset: map error obtained with a standard lawnmower-type path (a) and map error obtained with the path planned using our method (b).

4.3.3 *Real-world Validation*

We validate our method in the real world with the GIRONA 500 AUV at "El Jardí" and "l'Amarrador" area, which is shown in Figure 50. As in the simulation experiments reported above, we apply our algorithm to the bathymetric dataset, using the coverage graph depicted in Figure 43d, with an uncertainty threshold $\alpha = 20$. Nonetheless, by contrast, we compare the paths planned using our method with a standard lawnmower-type path by actually executing them *in situ* with the GIRONA 500 AUV and evaluating their associated uncertainty. Rather than using simulated navigation and multibeam measurements (as in the experiments discussed above), here we use the actual multibeam measurements and trajectory estimation formed onboard the vehicle during the mission to evaluate the effect of the planned paths on the robot's belief. That is, we feed these real-world data (multibeam measurements and trajectory estimation) to our particle filter in order to perform the evaluation.

The sea trials we next report on took place in October 2013. Figure 51 shows GIRONA 500 being deployed at "El Jardí" and "l'Amarrador" area during the trials.

Figure 50: Panoramic view of "El Jardí" and "l'Amarrador" area from GIRONA 500's support surface vessel.



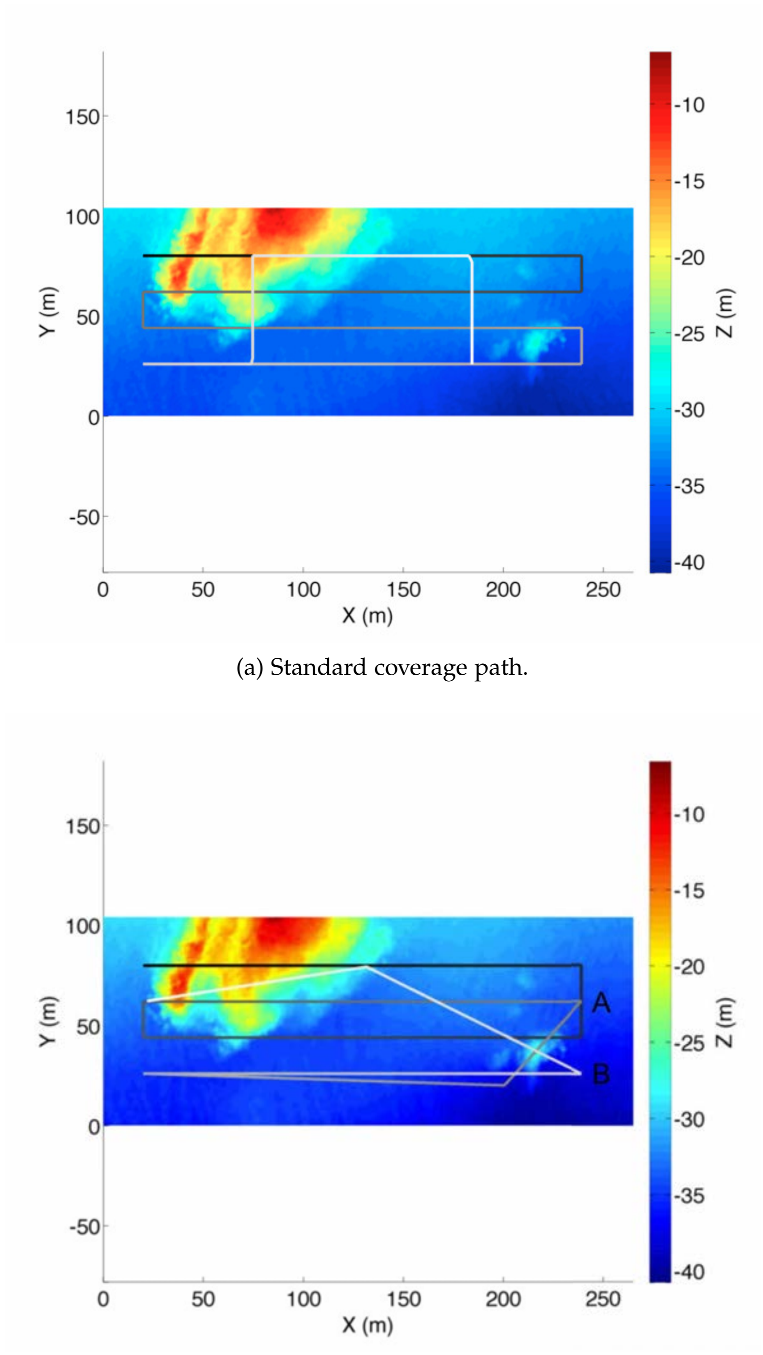
(a) Deployment from the support vessel.

(b) GIRONA 500 initiating a survey.

Figure 51: Deployment of the GIRONA 500 AUV during the uncertainty-driven coverage path planning trials.

A standard coverage path and an uncertainty-driven coverage path were planned and executed in the target area. The standard and uncertainty-driven planned paths are shown in Figure 52. To provide the big picture of the geographical location where these experiments take place, Figure 53 shows the uncertainty-driven coverage path overlaid on satellite imagery. Figure 54 shows the actual execution of each planned path by the vehicle, as estimated by its localization system using its onboard sensors. Note the small differences from the planned path due to the path following controller used to maneuver the vehicle. Nonetheless, these differences are not significant in relation with the scale of the survey.

The belief uncertainty associated with each path is shown in Figure 55. Note how, as in the simulation experiments, the average uncertainty of the path planned with our method is kept below that of the standard path. In addition, the standard path presents a high uncertainty peak at approximately 900 m caused by the traversal of two roughly feature-less tracks, whereas the uncertainty-driven path avoids this undesirable effect.



(a) Standard coverage path.

(b) Uncertainty-driven coverage path.

Figure 52: Distance-colored (from black to white) standard coverage path and uncertainty-driven coverage path planned for the "El Jardí" and "l'Amarrador" dataset. A, B indicate, sequentially, the locations where the uncertainty surpasses the threshold and a crossing track is added to the path.

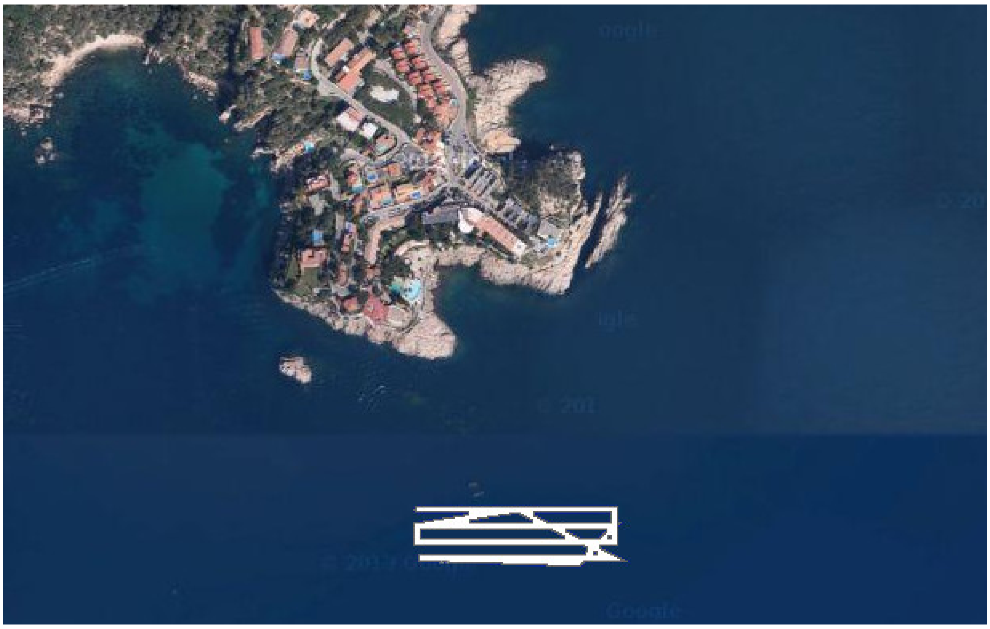
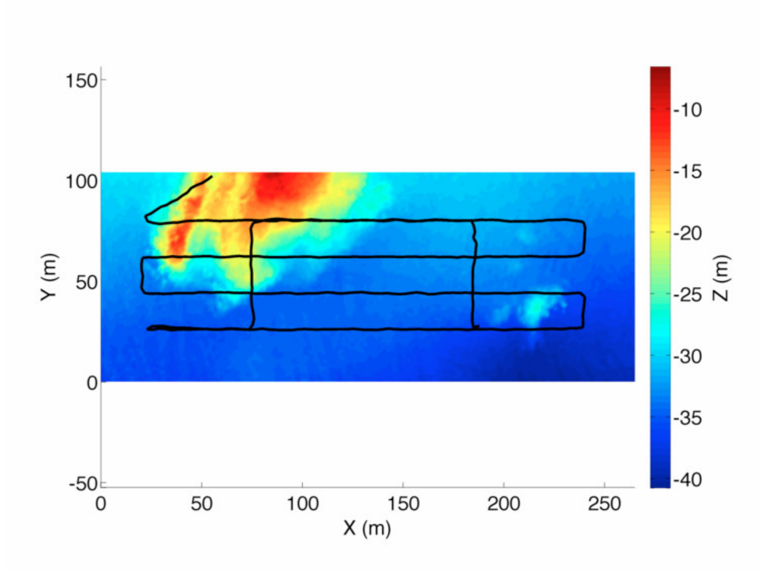
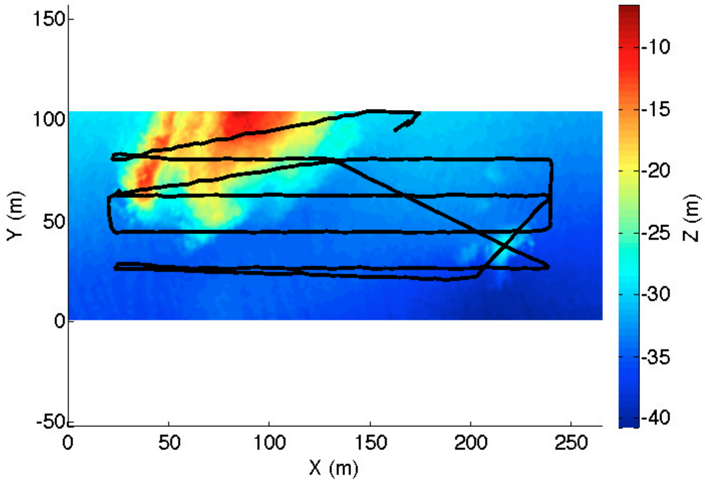


Figure 53: Uncertainty-driven coverage path for "El Jardí" and "l'Amarrador" dataset plotted on satellite imagery from Google Maps.



(a) Execution of the standard coverage path.



(b) Execution of the uncertainty-driven coverage path.

Figure 54: Path execution of the planned paths by the GIRONA 500 AUV during "El Jardí" and "l'Amarrador" experiments as estimated using the vehicle's onboard sensors. Both paths include an initial phase in which the vehicle navigates to the path's first waypoint from its deployment location.

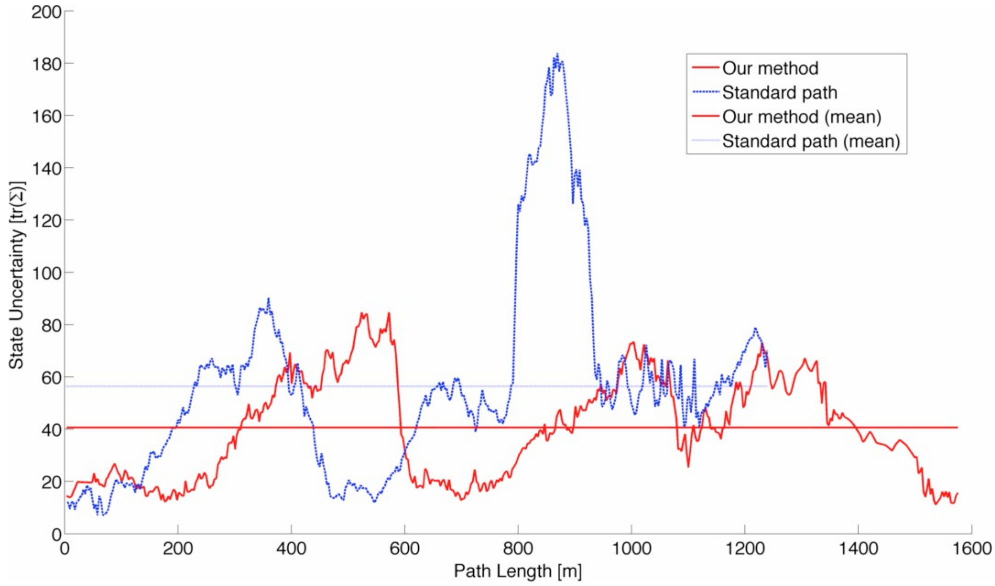


Figure 55: Belief uncertainty vs. path length for a standard lawnmower-type path and a path planned using our proposed method executed by GIRONA 500 at "El Jardí" and "l'Amarrador" area.

5

3D COVERAGE OF COMPLEX STRUCTURES IN UNDERWATER ENVIRONMENTS

Beware of bugs in the above code;
I have only proved it correct, not tried it.

— Donald E. Knuth (1979)

Up to this point, this thesis has addressed coverage problems in 2D scenarios. However, there are plenty of structures on the ocean floor, natural and man-made as well, that can not be successfully surveyed in detail without resorting to a 3D workspace. Examples include coral reefs, ship wrecks, breakwater structures in harbors or the continental slope, to name but a few. In this chapter, we present 3D* coverage path planning algorithms for inspection of such structures. After introducing the 3D coverage path planning problem in the context of underwater robotics in Section 5.1, we present an off-line coverage path planning algorithm that generates a complete coverage path suitable for covering sea floor areas including 3D structures (Section 5.2). The algorithm identifies high-slope regions on an *a priori* bathymetric map and provides different path patterns suiting both effectively planar and high-slope regions, completely covering the target area. Most coverage algorithms rely on the assumption of an idealized path execution. However, the vehicle suffers from position error and environmental disturbances during the path execution, rendering such an assumption unrealistic. This is especially patent when navigating amidst complex 3D structures, where the threat of collision increases. Rather than relying on an idealized execution of the planned path, we present two approaches to deal with discrepancy between a nominal 3D coverage plan and the actual situation the AUV faces *in situ* during the coverage task. The first approach (Section 5.3) is reactive and uses range sensor measurements to follow cross-section profiles of the target surface in realtime. The second approach (Section 5.4) uses realtime replanning to reshape the nominal coverage path according to the actual target structure perceived on the site via the vehicle's onboard sensors. Finally,

* Strictly speaking, our algorithms operate mostly in a 2.5D scenario (a bathymetric map), although after realtime path execution they enable a full 3D perception of the target environment. Bearing that in mind and that coverage of 2.5D environments requires a 3D workspace, we use the term "3D coverage" throughout this chapter.

we present an experimental validation of the proposed methods using the real-world AUV GIRONA 500 in Section 5.5.

5.1 INTRODUCTION

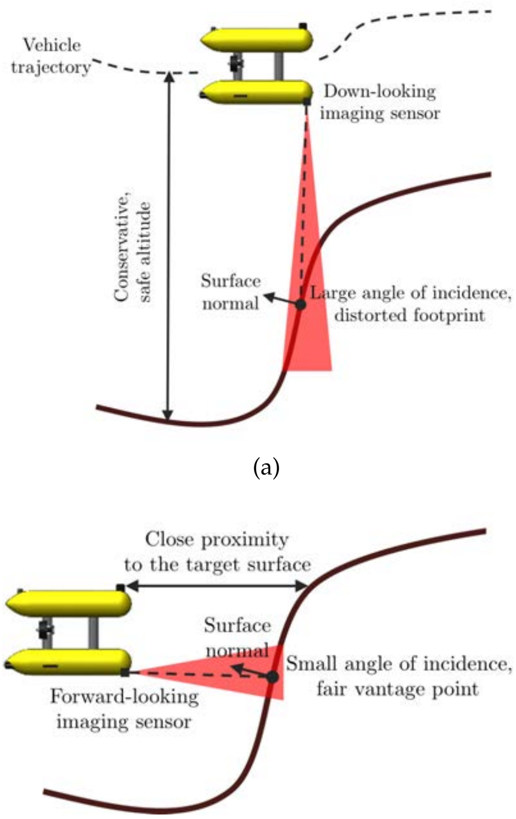
At present, in most AUV survey missions the vehicle covers an area with its payload sensors by following a pre-planned lawnmower-like survey path while keeping a safe altitude from the sea bottom. This is a valid approach for seafloor areas that are effectively planar at the survey scale. However, flying at a conservative altitude imposes serious limitations for a number of emerging applications requiring fine-scale seafloor surveys of rugged terrain in close proximity that allow acquisition of high-resolution imagery or even object grasping. Examples include monitoring of cold water coral reefs, oil and gas pipeline inspection, harbor and dam protection and object search and recovery. Therefore, techniques that allow the AUV to maneuver in close proximity to the seabed without compromising vehicle safety are desired.

On the other hand, following the elevation profile of the seabed does not provide satisfactory results when surveying rugged, high-relief terrain such as coral reefs or ship wrecks. These sites present very steep slopes that cannot be imaged with acceptable quality from an overhead point of view. As illustrated in Figure 56, a survey at a safe altitude from the bottom provides an askew angle of incidence with respect to the bottom's surface normal, which results in poor imaging. It is rather desired that the AUV places its sensor so that a viewing angle close to the surface normal of the target structure is achieved.

Therefore, in order to meet these requirements, flying at a conservative distance from the seabed is no longer an option. The AUV must rather navigate amidst bulges sticking out of the bottom. Obviously, this increases the threat of collision.

Approaches to plan a coverage path accounting for obstacles have been proposed in the literature, as discussed in Chapter 2. However, they typically assume perfect knowledge of the environment or perfect sensing, which are unrealistic assumptions in the vast majority of scenarios and especially in underwater environments, even when using techniques such as TRN or SLAM for enhanced localization. This limits real-world application of those approaches to very constrained, controlled environments. Hereby, the rest of this chapter will present algorithms that seek to push the envelope in this matter.

It is worth noting that the algorithms in this chapter have been designed with a hovering-capable AUV in mind. With such a capability it is easier to orientate the vehicle's static sensors to provide a more rich perception than with typical torpedo-shaped vehicles. However, when using torpedo-shaped vehicles to follow our planned paths a pan and tilt unit can be used to enable a proper orientation of the sensor footprint.


Vehicle trajectory
Down-looking imaging sensor
Conservative, safe altitude
Surface normal
Large angle of incidence, distorted footprint
Close proximity to the target surface
Surface normal
Small angle of incidence, fair vantage point
Forward-looking imaging sensor

(a)

(b)

Figure 56: Askew angle of incidence provided by constant, safe altitude survey (a) in contrast to the fair viewing angle obtained by imaging the surface in parallel to its normal (b).

5.2 COVERAGE PATH PLANNING ON BATHYMETRIC MAPS

Our proposed off-line coverage path planning method operates on a bathymetric map, $\mathcal{B}(x,y)$, provided as input. For every point (x,y) on the mapped area, $\mathcal{B}(x,y)$ returns its depth. The proposed coverage path planning method is a three-step process, as illustrated in Figure 57. First, the terrain (bathymetric map) is classified into high-slope regions and effectively planar regions. Next, a coverage path that follows the horizontal cross-sections of the surface is generated in the high-slope (3D) regions using a slicing algorithm we put forth. Finally, a coverage path is planned for the remaining effectively planar (2D) regions of the target surface. This later coverage path is generated using the Morse-based boustrophedon cellular decomposition approach (introduced in Section 2.3), where the high-slope regions are treated as obstacles.

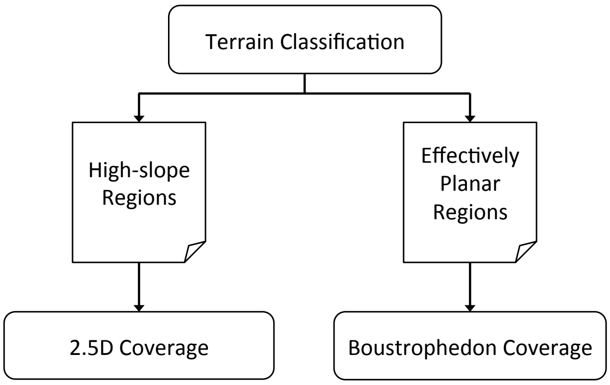


Figure 57: Diagram of the proposed coverage path planning algorithm for bathymetric maps.

5.2.1 *Terrain Classification*

We classify the terrain into high-slope and effectively planar regions using a "slope map", $S(x,y)$. The slope map is calculated for the mapped area as the norm of the gradient of $\mathcal{B}$, that is:

$$S(x,y) = \|\nabla\mathcal{B}\| = \left\|\frac{\partial\mathcal{B}}{\partial x}\vec{i} + \frac{\partial\mathcal{B}}{\partial y}\vec{j}\right\|,$$

where $\vec{i},\vec{j}$ are the standard unit vectors in the X and Y axis, respectively. Then, we apply a user-defined slope threshold, δ_s, to S to obtain an initial binary classification:

$$T(x,y) = \begin{cases} 1 & \text{if } S(x,y) \geqslant \delta_s \\ 0 & \text{if } S(x,y) < \delta_s \end{cases}.$$

The choice of δ_s strongly depends on the scale of the mapped area. A valid option is to normalize S into the $[0\ldots1]$ range and select $\delta_s = 0.5$.

In order to filter out small regions and potential artifacts in the initial classification, we apply the dilate and erode morphological operations (Serra, 1982) to T using an appropriate structuring element.

Let us illustrate this terrain classification process with the simple example shown in Figure 58. First, according to the target surface (Figure 58a), the slope map is computed and normalized into the $[0 \ldots 1]$ range (Figure 58b). Then, a slope threshold $\delta_s = 0.5$ is applied to the slope map to obtain an initial classification (Figure 58c). Lastly, morphological operations are used to obtain the final classification (Figure 58d). Note how the morphological operations eliminate the holes of the high-slope regions in the initial classification. However, a hole (a planar region "inside" a high-slope region) can be also covered by the coverage algorithms described below. The subsets of the original surface corresponding to each class (high-slope and effectively planar) are shown in (Figures 58e and 58f).

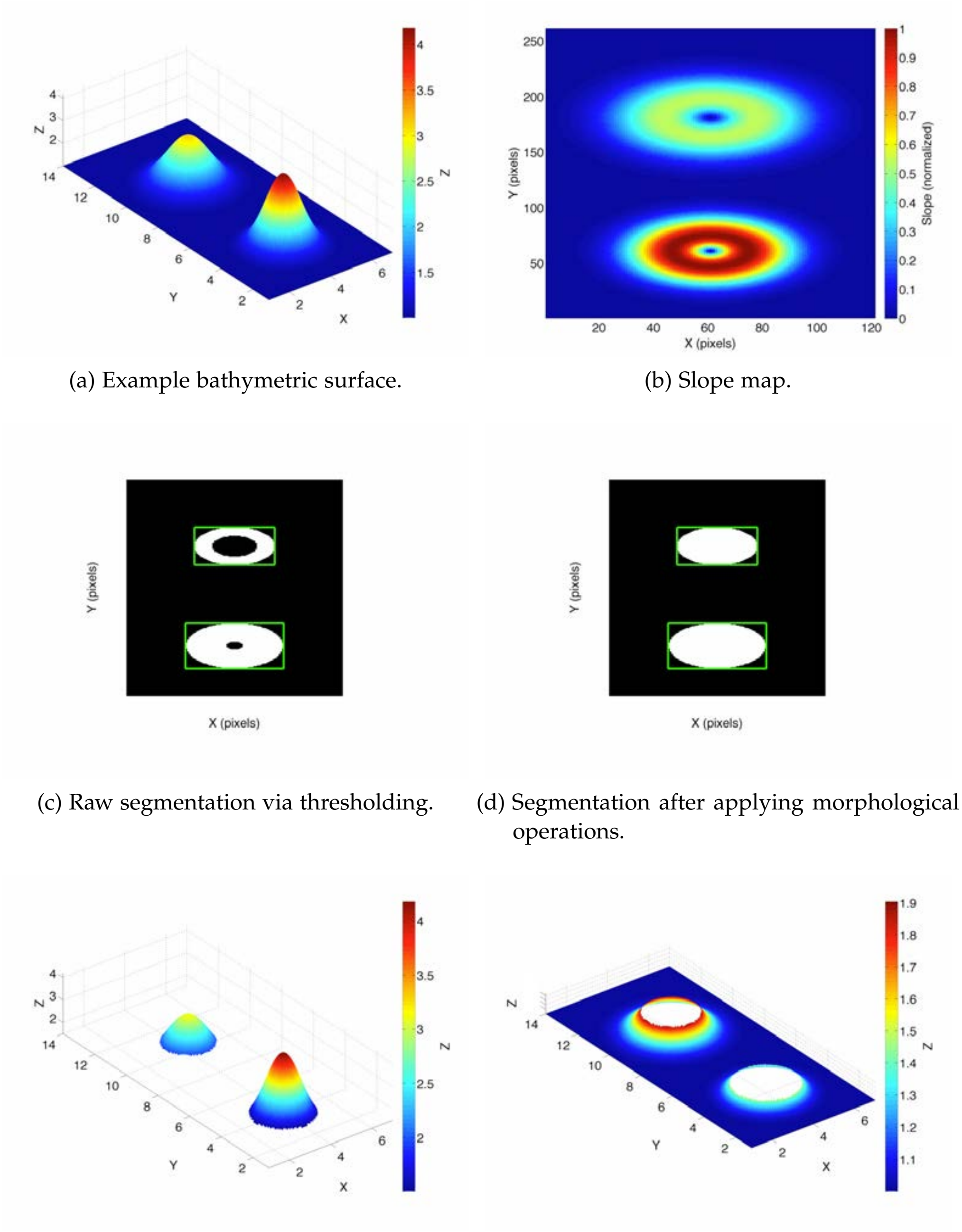
5.2.2 *Covering High-Slope Regions using a 2.5D Slicing Algorithm*

We propose a slicing algorithm to generate an in-detail coverage spath for each identified high-slope region. The proposed algorithm draws inspiration from the algorithm of Atkar et al. (2001). The main idea is to intersect a horizontal slice plane with the target surface at incremental depths, and then link these intersections. The resulting intersections correspond to contour lines, or level curves, of the target bathymetric surface, accumulating data contour-by-contour along the vertical spatial dimension of the workspace. As a result, the path enables acquisition of a clear and continuous data product, simplifying the tasks of post-processing and analysis for both humans and automated procedures.

Consider a robot equipped with a limited FOV sensor. The sensor FOV is determined by an aperture angle, α, and a maximum range r_{max}, as shown in Figure 59. To image the target surface with the sensor, the robot navigates at a user-defined fixed offset distance, $\Omega < r_{max}$, from the target surface. The sensor footprint on the target surface determines the spacing between successive slice planes, $\Delta\lambda$ (where λ is the current slice plane depth). Note that the footprint extent depends on the curvature of the target surface on the imaged area. We approximate the footprint extent as a circle of radius $r = \Omega \tan \frac{\alpha}{2}$, and therefore $\Delta\lambda = 2r$.

The slicing algorithm is detailed in Algorithm 3. The algorithm is applied to each identified high-slope region of the bathymetric map. For each high-slope region, the algorithm takes as input the corresponding subset of the bathymetric map $\mathcal{B}_r$ (for the example described above, the subset depicted in Figure 58e), the offset distance, Ω, and the slice plane spacing, $\Delta\lambda$.

An example run of the algorithm on the high-slope regions of Figure 58e is illustrated in Figure 60. First, the algorithm initializes the horizontal slice plane depth, λ, as the minimum depth (the shallowest) in $\mathcal{B}_r$ (line 1). The set of intersection edges, E, is initialized as empty in line 2. The algorithm runs at incremental values of λ until λ surpasses the maximum depth in $\mathcal{B}_r$ (line 3). That is, from the shallowest down to the deepest point in $\mathcal{B}_r$



(a) Example bathymetric surface.

(b) Slope map.

(c) Raw segmentation via thresholding.

(d) Segmentation after applying morphological operations.

(e) Areas classified as high-slope.

(f) Areas classified as effectively planar.

Figure 58: Example of terrain classification into high-slope and effectively planar regions on a synthetic bathymetric surface. The white regions in (c) and (d), delimited by their bounding boxes (in green), correspond to high-slope areas; the black regions correspond to effectively planar areas.

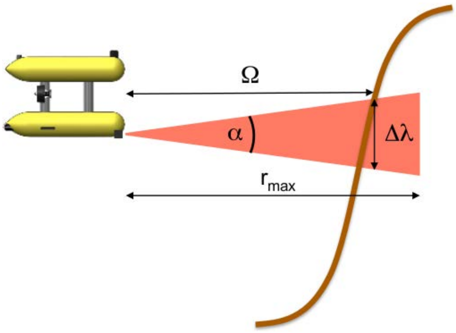


Figure 59: Sensor FOV of a robot located at an offset distance Ω from the target surface. The sensor footprint determines the distance between slice planes, Δλ.

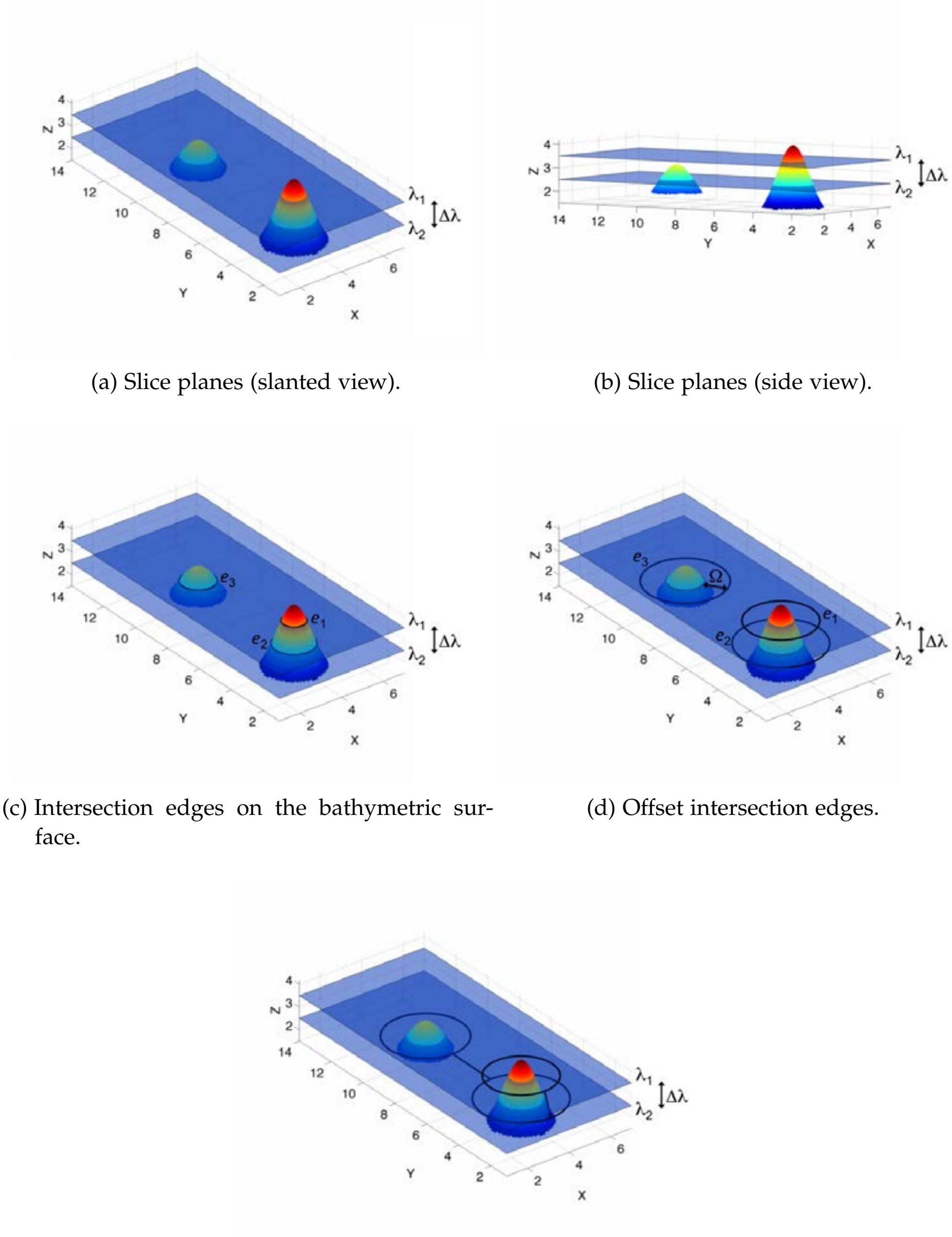
(see Figures 60a-60b). At each depth level, a horizontal plane is intersected with the bathymetric surface (line 4). The function Intersect() returns the set of edges composing the intersection, as illustrated in Figure 60c. The intersection edges of the current slice plane are added to E. Recall that these edges correspond to the level curves of $\mathcal{B}_r$ at the current depth λ, and that these edges are not necessarily closed. In fact, these intersections can be determined using an off-the-shelf contour drawing algorithm, such as that implemented by MATLAB's CONTOURC function.

Notice that, when the while loop exits, the intersection edges lie exactly on the bathymetric surface. To obtain a coverage path at the desired offset distance, the edges are then projected onto the offset surface by OffsetEdges(), which projects all points in the edges outward from the target surface by an offset distance Ω, as shown in Figure 60d. To do so, each point in the original edge is offset along the projection of the bathymetric surface normal vector on the corresponding horizontal plane.

Lastly, the final coverage path is generated by linking all edges in the set (function LinkEdges(), line 6). The function LinkEdges() links all edges in E in a greedy manner. Starting at the first edge in the set, belonging to the first depth level, it will link it to the closest edge in the set, and do so repeatedly for all remaining edges until all of them have been linked. To link a given pair of edges, LinkEdges() simply traces a straight line path from the first edge to the second. If the straight line intersects the bathymetric surface, it traces the projection of the line on the bathymetric surface. The result of this linkage procedure is shown in Figure 60e.

5.2.3 *Covering the Effectively Planar Regions using the Boustrophedon Decomposition Algorithm*

Since high-slope regions will be covered by the slicing algorithm, we generate a coverage path for the remaining effectively planar regions treating the (already addressed) high-slope regions as obstacles. To do so, we use the Morse-based boustrophedon decomposition coverage algorithm (Section 2.3).



(a) Slice planes (slanted view).

(b) Slice planes (side view).

(c) Intersection edges on the bathymetric surface.

(d) Offset intersection edges.

(e) Final coverage path.

Figure 60: Application of the slicing algorithm on an example bathymetric surface. The target high-slope regions are intersected with the slice planes (a)-(b), producing a set of intersection edges (c). The intersection edges are then offset by the desired offset distance Ω (d). Lastly, the final coverage path is obtained by linking the offset edges (e).

Algorithm 3: Slicing Algorithm

```
Input: High-slope region of a bathymetric map, $\mathcal{B}_r(x, y)$
Parameters: Offset distance, $\Omega$. Slice plane spacing, $\Delta\lambda$.
$\lambda \leftarrow \min \mathcal{B}_r(x, y) + \Delta\lambda$
$E \leftarrow \emptyset$
while $\lambda < \max \mathcal{B}_r(x, y)$ do
    $E \leftarrow E \cup \mathtt{Intersect}(\lambda, \mathcal{B}_r)$
    $\lambda \leftarrow \lambda + \Delta\lambda$
$E \leftarrow \mathtt{OffsetEdges}(E, \Omega)$
$p \leftarrow \mathtt{LinkEdges}(E)$
return $p$
```

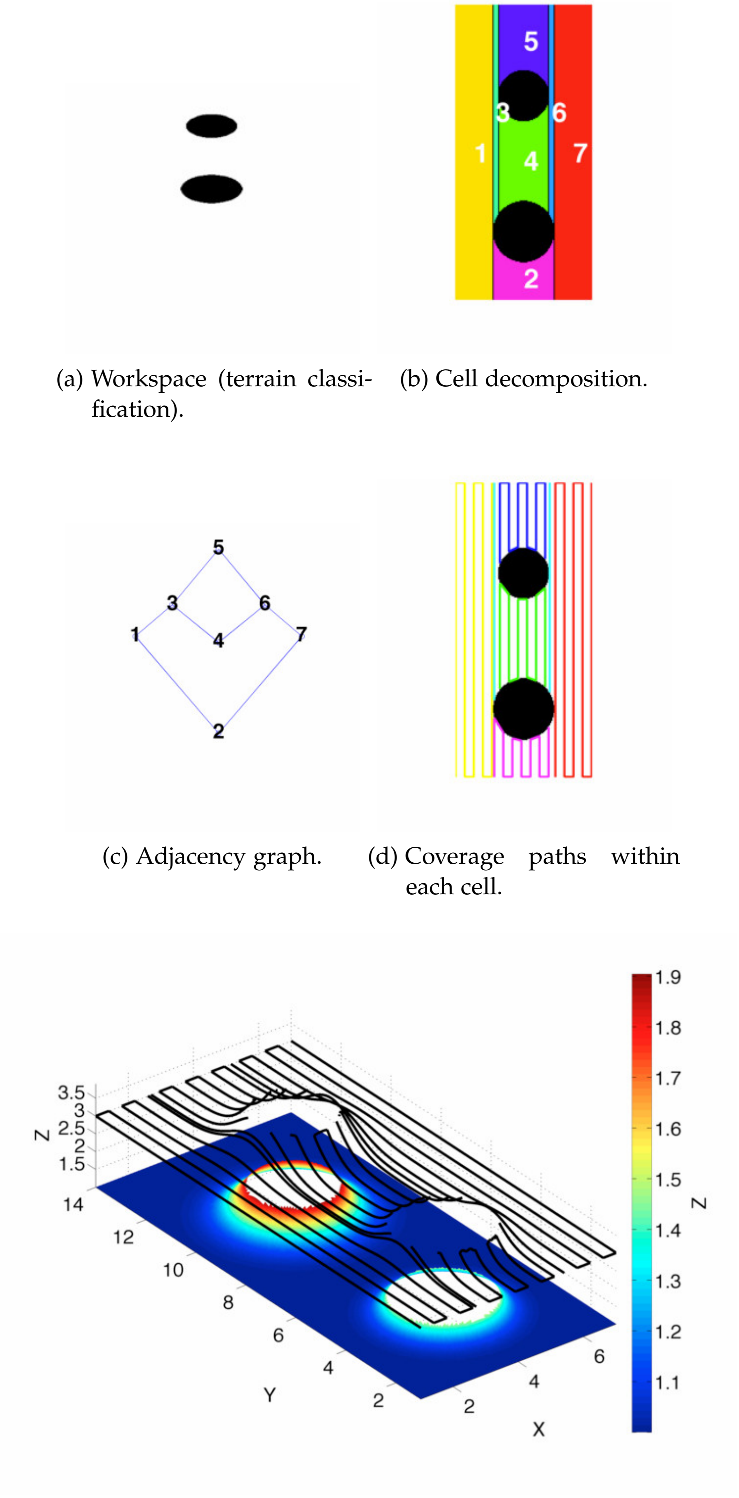
We apply the boustrophedon decomposition to the 2D workspace composed by the terrain classification procedure, where high-slope regions represent obstacles. Following on the example introduced above, Figure 61 shows the execution of the algorithm. The workspace (Figure 61a) is decomposed into cells (Figure 61b), which are encoded as an adjacency graph (Figure 61c). Then, an exhaustive walk through the graph is determined to obtain the order in which to cover the cells, and finally individual coverage paths are generated within each cell (Figure 61d) and linked according to the exhaustive walk. Notice how the boustrophedon path maintains a constant offset distance from the bottom.

5.2.4 *Final Coverage Path*

The union of both coverage paths (the coverage path for high-slope areas and the coverage path for effectively planar areas) provides full coverage of the target bathymetric surface, using different coverage patterns according to the terrain's slope and providing the AUV with suitable viewpoints for imaging tasks. Figure 62 shows the coverage paths for both high-slope and effectively planar areas generated on the example target surface introduced above.

5.3 3D COVERAGE WITH SENSOR-BASED PROFILE FOLLOWING

As stated earlier, assuming an idealized execution of the planned path for 3D coverage is not a feasible option. Due to error in the *a priori* map used to plan the path, position error accumulated during the path execution and environmental disturbances, the vehicle is exposed to a threat of collision with the target structure. Next, we propose a method to address this problem based on profile estimation and following using range sonar information in realtime. The key idea of the method is, as in Algorithm 3, to follow the cross-section profiles of the target structure, such as those resulting from the intersections of the slicing plane with the structure in Figure 60. However, rather than following a coverage path planned off-line consisting of



(a) Workspace (terrain classification).

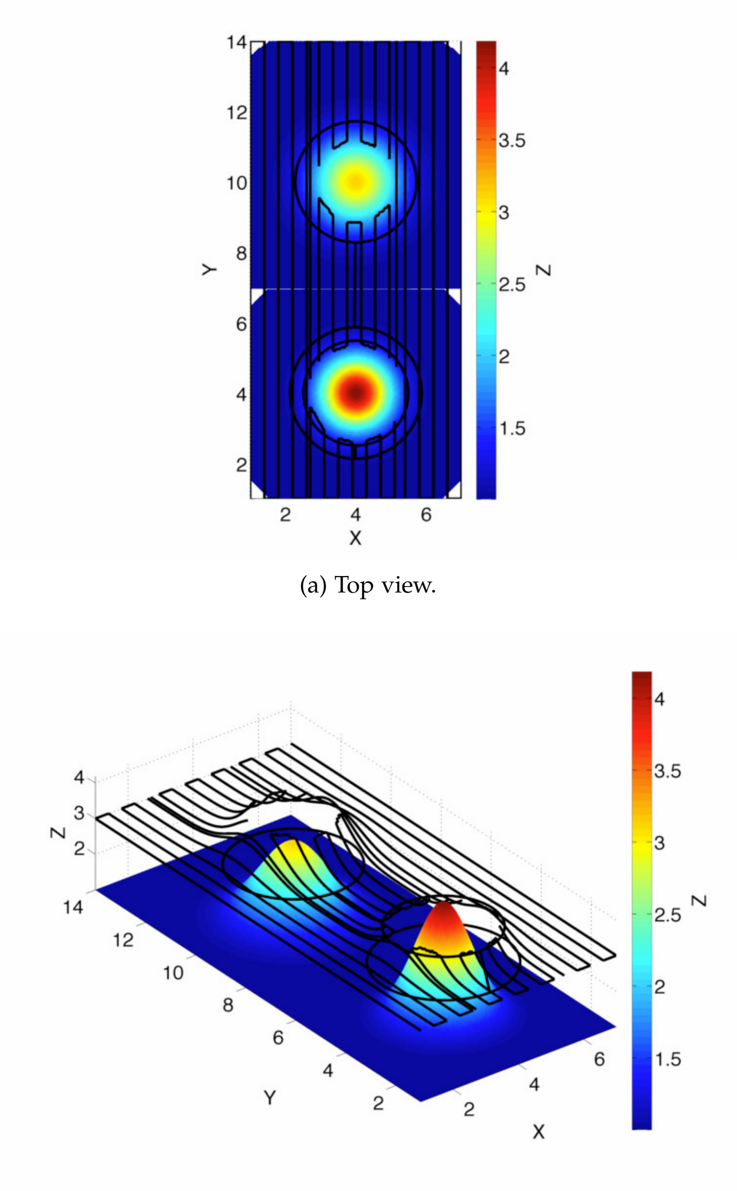
(b) Cell decomposition.

(c) Adjacency graph.

(d) Coverage paths within each cell.

(e) Final coverage path for the effectively planar regions.

Figure 61: Application of the Morse-based boustrophedon decomposition algorithm for coverage of effectively planar areas on an example bathymetry.



(a) Top view.

(b) Slanted view.

Figure 62: Coverage path for both high-slope and effectively planar areas.

a list of waypoints and relying on the vehicle's navigation accuracy to do so, we propose to follow each actual contour perceived in situ using range measurements from a profiling sonar. That way the AUV does not need to rely on near-perfect localization and control, being able to survey the target structure using its range sonar measurements.

The bottom following problem, that is, following the vertical profile of the sea bottom, described as "maintaining a fixed altitude above an arbitrary surface whose characteristics may or may not be known" (Bennet et al., 1995), has been addressed in prior work using different sensors, such as high-frequency pencil beam profiling sonars (Caccia et al., 1999) or a pair of altimeters to estimate the profile (Caccia et al., 2003). In regard to AUV control, a variety of different schemes have been proposed for bottom following. Creuze et al. (2001) proposed a seabed-following trajectory generation algorithm for torpedo-shaped vehicles. Their algorithm computes trajectories using simple geometric functions and interpolation curves called "semi-forced cubic splines." Melo and Matos (2012) presented a basic guidance approach to provide depth and pitch references to, as in the aforementioned work, a torpedo-shaped vehicle. Other works propose more complex control techniques such as nonlinear output regulation (Adhami-Mihosseini et al., 2011). Recently, Houts et al. (2012) presented a technique for aggressive seabed terrain following. Estimates from a terrain-based navigation system are used to anticipate the terrain. As in this work, their method pursues closer proximity seabed following. However, their technique is concerned only with the seabed's vertical profile. Karras et al. (2013) proposed a robust control scheme for wall profile (horizontal) following using sonar measurements.

Nonetheless, little attention has been paid to following horizontal profiles, as required by our coverage method. Moreover, the existing approaches for bottom following are basically targeted at traditional torpedo-shaped AUVs, and do not take into account the hovering capabilities of recent AUV developments. This limits the profile following fidelity provided by these approaches.

By exploiting hovering capabilities of recent AUV developments, our proposed method is able to survey the target structure in close proximity while avoiding collision. Moreover, besides horizontal profile following, it can be used for vertical sea floor profile following as well. The proposed method builds and maintains a local map of the immediate AUV surroundings using sonar range information. The map allows to filter out noisy sonar range measurements and provides a convenient representation to estimate the profile using previously sensed ranges as new data becomes available. An estimate of the local profile to follow is obtained by means of simple linear regression. Then we use horizontal and vertical profile following controllers which react to the profile estimates and provide the appropriate commands to the vehicle's thrusters. We next describe the profile estimation (Section 5.3.1) and profile following (Section 5.3.2) schemes. Then, we detail how these profile estimation and following techiques are integrated with the slicing algorithm introduced above and with the control architecture of GIRONA 500 to achieve 3D coverage on-line (Sections 5.3.3 and 5.3.4).

5.3.1 *Profile Estimation*

Our profile estimation method is targeted at AUVs equipped with a pencil-beam sonar or a sonar array providing range information in a circular sector of 180°. Although we focus on AUVs equipped with sonar in this article, the same strategy we propose can be used with other range-and-bearing sensors, such as stereo cameras. We use a two-module profile estimation process. First, a mapping module uses the sonar range information to construct and maintain a local map of the profile to follow. Second, a profile estimation module performs linear regression on certain regions of the local map to obtain an estimate of the profile around the vehicle.

5.3.1.1 *Local Mapping*

We use the sonar range information to construct and maintain a local, probabilistic occupancy-grid map of the structure profile to follow. The probabilistic map serves two main purposes. First, it allows to deal with noisy measurements by assigning a higher likelihood to cells that have been consistently reobserved as either occupied or free and hence filtering out outliers. Second, it provides a convenient representation upon which to perform profile estimation at a higher rate than the sensed ranges arrival rate (the later being typically slow in the case of mechanically scanning devices).

In fact, depending on configuration settings, mechanically scanning pencil-beam sonars take up to several seconds to provide a scan. It is the case of the pencil-beam sonar we use in our experiments (see Section 5.5 below), which takes up to 4 seconds to provide a 180° scan. Waiting for a whole sector scan to become available delays the subsequent profile estimation process on the recently collected sonar beams. We tackle this problem by feeding subsequent small sub-sector scans (of about 10°, depending on configuration) to the mapping framework. By doing so we obtain more frequent map updates, enabling the profile estimation module to promptly incorporate recent sonar readings. To correct the distortion induced by the vehicle motion, we use the vehicle position estimates obtained during the scan time. (The reader can refer to Appendix B for more details on the sonar technology used in this work.)

It is worth noting that, even if using a fast sensor (such as a multibeam sonar or a stereo camera) to perceive the profile, the probabilistic map representation is still beneficial in dealing with noisy measurements and obtaining a reliable perception as the environment is reobserved.

We use the Octomap Hornung et al. (2013) probabilistic mapping framework to construct and maintain our 3D map, which uses an octree map compression method to keep the 3D model compact and quickly accessible. Figure 63 shows a snapshot of a local map constructed in our laboratory's pool.

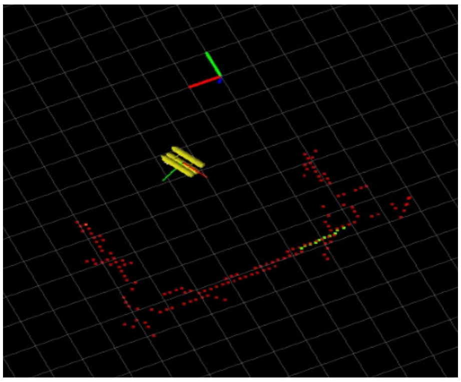


Figure 63: Local mapping module running on our laboratory's pool. The lightest (green) points show the latest incorporated sonar sub-scan and the red points show the occupied cells (with probability > 0.5) in the map. A real-sized model of the GIRONA 500 AUV and the world and vehicle coordinate frames are also shown.

5.3.1.2 *Profile Estimation on Regions of Interest*

The profile estimation module operates on different regions of interest on the map, providing a local profile estimation for each region. Then, these estimates are appropriately used by the profile following modules. The horizontal profile estimation module operates on two regions, one in the front of the vehicle and one on the right of the vehicle, as depicted in Figure 64.

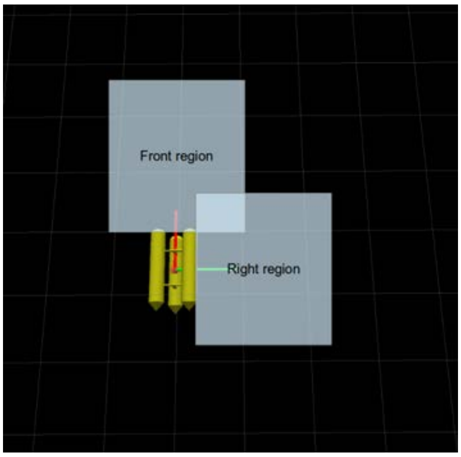


Figure 64: Regions of interest for horizontal profile estimation.

The vertical profile estimation module operates on the three regions shown in Figure 65: one below the vehicle, termed "bottom region"; one in front of the vehicle, termed "cliff region"; and one above the vehicle, termed "ceiling region".

The size (width and height) of these regions is determined by the size of the vehicle and by the desired offset distance at which the profile needs to be followed. The 3D coordinates of the occupied cells (with probability > 0.5) of the map are projected on a 2D horizontal or vertical plane depending on

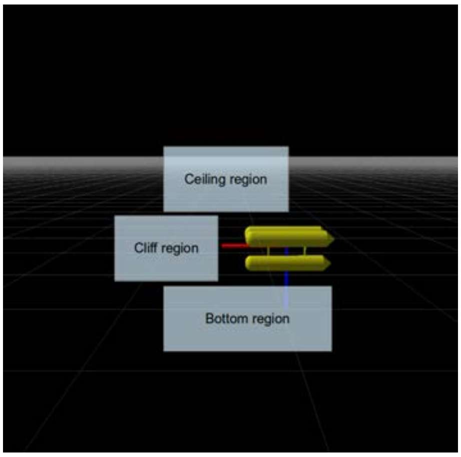


Figure 65: Regions of interest for vertical profile estimation.

the current profile following mode. Then the profile is estimated on the 2D projection.

We estimate the local profile on each projected map region of interest by fitting a line to the points in the region. A line $y = ax + b$ is fitted to the N points (x_i, y_i) in the region via least squares, with slope a and intercept b given as follows:

$$a = \frac{\sum y_i - b \sum x_i}{N},$$

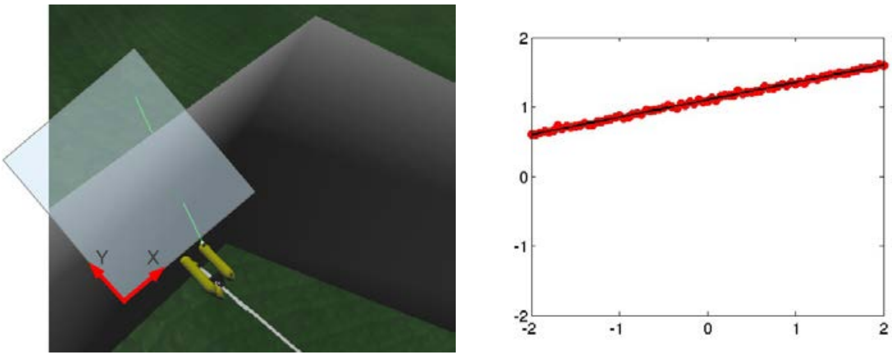
$$b = \frac{N \sum (x_i y_i) - (\sum x_i)(\sum y_i)}{N \sum x_i^2 - (\sum x)^2}.$$

More complex interpolation methods, such as splines Bartels et al. (1987), could be used for profile estimation. These techniques could be beneficial when the AUV is following a profile from far enough so it perceives a large piece (in the order of several meters) of the profile after one sensor scan. However, our target application (inspection of underwater structures) requires following the profile in close proximity, and therefore only a small piece of the profile is visible to the vehicle at any given time. In this situation the profile can be effectively approximated as a straight line. Moreover, line fitting can be computed quickly and provides a compact two-parameter representation.

Additionally, we compute the average distance from the points to the vehicle's reference frame, $\hat{d}$. Figure 66 shows the profile estimation procedure operating on a 3D structure in a simulated environment.

5.3.2 *Profile Following*

We have designed and implemented two profile following modules, one for horizontal profile following and one for vertical profile following. It is assumed that the vehicle is hovering-capable and can be controlled in the surge (X), sway (Y), heave (Z) and heading (yaw) DOFs. The modules use



(a) Simulated environment.

(b) X-Y plot showing the plane-projected map points and the estimated line corresponding to the scene shown in (a).

Figure 66: Profile estimation.

the output from profile estimation and are designed as a set of coordinated behaviors which use proportional-integral (PI) controllers to generate speed commands in the appropriate DOFs. The speed commands generated by the behaviors are not sent directly to the vehicle's thrusters, but rather sent to a low-level velocity controller which merges and coordinates all the commands sent from all the modules of the robot's control architecture.

5.3.2.1 *Horizontal Profile Following*

The horizontal profile following module consists of a single behavior operating on the surge (X), sway (Y) and heading (yaw) DOFs. It assumes that an external module controls the vehicle to keep a certain constant depth.

The module seeks to advance the vehicle along the frontal region profile while keeping a certain desired distance, δ_h, from the frontal profile and facing perpendicularly to the profile (that is, keeping the profile slope, a, close to 0). The vehicle is advanced along the profile counter-clockwise (from an overhead viewpoint).

As shown in Figure 67, a PI controller operating on the X DOF is fed the frontal distance error, $e_{hf} = \delta_h - \hat{d}$, which it seeks to keep to 0. A second PI controller on the heave DOF is fed the slope, a. When both e_{hf} and a are below user-provided tolerances, the vehicle advances in the Y DOF at constant speed, V_y. Together with V_y, the desired speeds in the X and yaw DOFs (V_x and V_{yaw}, respectively) are sent as setpoints to the vehicle's low-level velocity controller.

The module keeps track of the average distance to the profile in the right region. If a profile is detected on the right region and the distance to that profile is less than δ_h, it turns the robot 90° clockwise. This effectively moves the front region to where the right region was located, allowing the robot to successfully react to highly non-convex profiles. Indeed, using this strategy avoids performing profile estimation in regions where a linear fit is difficult to be reliably maintained (such a sharp "V" in the profile). After the 90° turn, the profile following continues. This procedure is illustrated in Figure 68.

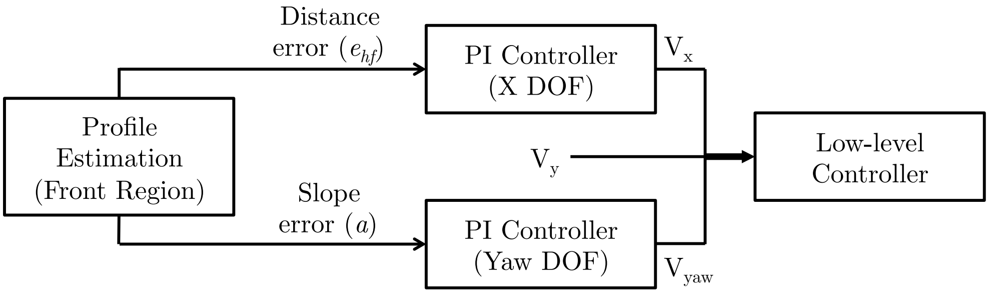


Figure 67: Horizontal profile following block diagram.

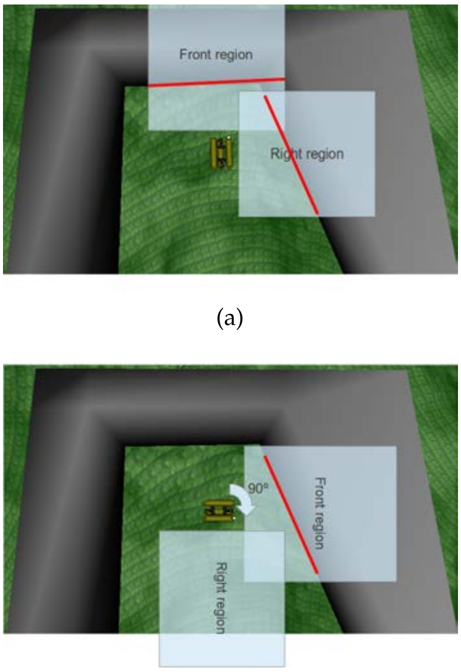


Figure 68: 90° turn maneuver on detection of a profile in the right region during horizontal profile following.

Very acute corners on the profile are seen by the profile estimation module as discontinuities. In practice, that means that the profile vanishes in presence of such corners, from the point of view of the robot's perception. In case that, during profile following, the profile estimation module fails to detect the profile, the module steers the vehicle to trace a circular trajectory with radius equal to δ_h (the vehicle's surge axis facing always its center). The robot traces the trajectory for a minimum sector angle (provided as a parameter), ignoring any potential profile estimation information. Then, it continues tracing the circular trajectory until the profile is detected. This allows the robot to continue profile following on the presence of very acute angles. This profile recovery maneuver is illustrated in Figure 69.

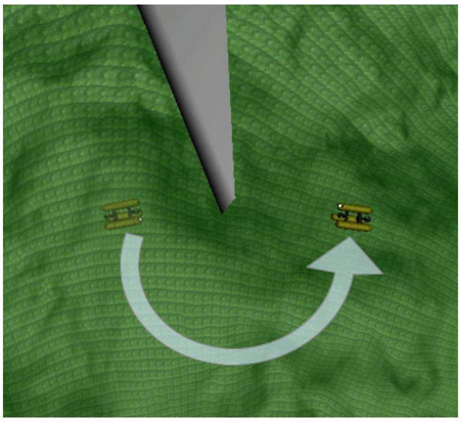


Figure 69: Horizontal profile recovery maneuver.

5.3.2.2 *Vertical Profile Following*

The vertical profile following module is implemented as a set of three coordinated behaviors: bottom following, cliff following and ceiling following. It is assumed that the robot is kept heading at a constant angle by an external module during vertical profile following.

Figure 70 shows a diagram of the vertical profile following module. A multiplexor selects which of the three behaviors takes over according to the profile estimation, that is, it enables the behavior for which a profile is detected in its associated region. If more than one behavior meets the condition, the behavior with the most recent associated profile estimation takes over. PI controllers in the X and Z DOFs are used by each behavior to keep the robot at a constant average distance, δ_v, from the profile (that is, they seek to keep $\delta_v - \hat{d} = 0$).

To avoid steering the vehicle backwards in the surge axis, the ceiling following behavior makes the vehicle turn 180° in yaw when it is engaged. Then, it proceeds steering the vehicle forward and keeping the appropriate distance from the ceiling. When the behavior is disengaged, it reverts the initial turn. This procedure allows the robot to surge forward in a direction well suited for its hydrodynamic configuration.

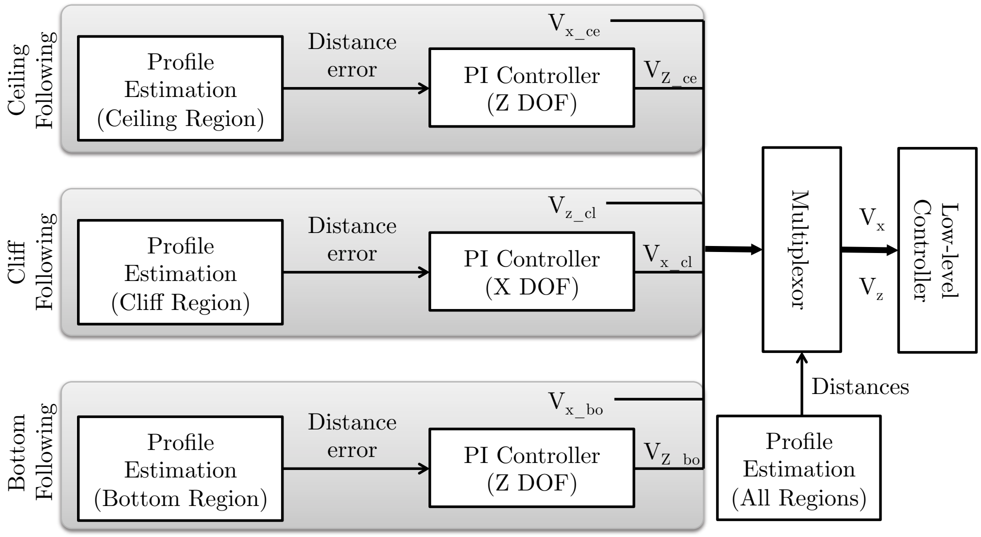


Figure 70: Vertical profile following block diagram.

5.3.3 *Integration with Coverage Path Planning*

In integrating our profile following scheme with coverage path planning, we make use only of the horizontal profile following part. The vertical profile following part has been introduced to show the applicability of our technique to the classical bottom following problem. To achieve a 3D coverage task with our profile following techniques we apply a simple two-step process. First, we use an *a priori* bathymetric map and the slicing algorithm to determine a coverage plan consisting of the inter-profile spacing, the number of cross-section profiles to be followed and the order in which they will be followed. Second, we execute profile following on-line on each determined cross-section profile of the target structure using the scheme presented above in Sections 5.3.1 and 5.3.2.

By applying the slicing algorithm (Algorithm 3), we obtain M intersection edges $e_{\lambda_i,1}, e_{\lambda_i,2}, \ldots, e_{\lambda_i,M}$ for each horizontal plane λ_i (recall Figure 60). A starting point where the robot will initiate horizontal profile following is assigned to each intersection edge. Once the cross-section profiles to be covered and their starting points have been determined, the AUV proceeds to cover them using our profile estimation and following method on-line. Initially, the AUV approaches the first starting point by diving towards it from the surface. Then, the horizontal profile following behavior is engaged and the coverage task begins. The vehicle's onboard navigation system is used to estimate its trajectory and determine when a complete cross-section profile has been completed. When the profile is entirely covered, the vehicle approaches the starting point of the next profile and repeats the profile following process until all profiles determined in the off-line planning phase have been covered.

5.3.4 *Integration with GIRONA 500's Control Architecture*

As detailed in Appendix A, GIRONA 500 is driven by a control architecture termed Component-Oriented Layer-based Architecture for Autonomy (COLA2), developed at CIRS. A simplified diagram of COLA2 highlighting its components involved in profile following for inspection tasks is shown in Figure 71. To address an inspection task, the off-line coverage planning module first generates a coverage plan using the slicing algorithm. The coverage plan is fed to the Task Execution component, which decides when to engage each profile following component according to the plan. The profile following components send commands to the low-level velocity controller, which then generates and sends velocity setpoints to the vehicle's thrusters. The navigation and mapping modules estimate the vehicle's trajectory and map its environment, respectively. These trajectory estimates and local maps are used by the profile estimation component to estimate the profile to be followed and feed it to the profile following components.

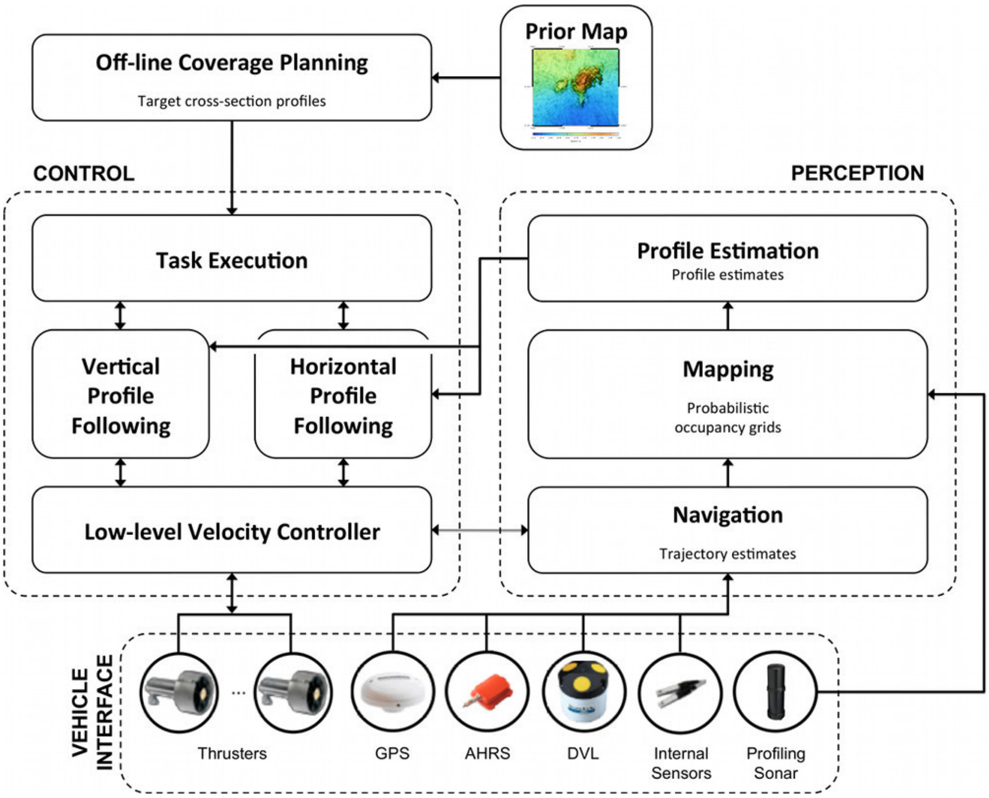


Figure 71: GIRONA 500's control architecture components for coverage tasks using profile following.

5.4 3D COVERAGE WITH REALTIME REPLANNING

Next, we present a new 3D coverage path planning method for inspection of complex structures on the ocean floor using AUVs which does not rely on the unrealistic assumption of an idealized path execution. Here, rather than directly reacting to range sensor measurements as in Section 5.3, we propose a replanning algorithm to reshape the path according to the actual target structure perceived on site using the vehicle's onboard sensors. Initially, the method takes as input a nominal coverage path planned off-line using the algorithm presented above in Algorithm 3. Recall that the planned coverage path follows the structure contours on the map at uniformly spaced depths maintaining a fixed offset distance from the target surface, accumulating data contour-by-contour along the vertical spatial dimension of the workspace. To handle the vehicle's position, environmental, and control uncertainty during path execution, we use stochastic trajectory optimization to adapt the initially planned coverage path in realtime. The resulting path is smooth and provides successful coverage under bounded position error.

5.4.1 *Problem Description*

Recall that the problem we tackle in this work calls for planning and executing a path that allows an AUV to pass its sensor payload over all points of the exterior boundary surface of a 3D underwater structure charted as a bathymetric map. The AUV is able to perceive the target structure with range sensors providing measurements at least in the vehicle's horizontal plane. When the AUV executes the path, its position estimate in the X (surge) and Y (sway) DOFs at time t, (x_t, y_t), is given by its true position $(\hat{x_t}, \hat{y_t})$ subject to a random error ϵ_t:

$$(x_t, y_t) = (\hat{x_t}, \hat{y_t}) + \epsilon_t, \tag{18}$$

where we assume ϵ_t is bounded by

$$|\epsilon_t| \leqslant \epsilon_{max} \forall t. \tag{19}$$

In a real-world AUV, ϵ_t is typically brought about by GPS error while at the surface and by dead-reckoning drift while underwater. The upper bound ϵ_{max} can be determined according to the accuracy of the AUV's navigation sensors and by the mission duration. Note that we assume the position errors in the remaining DOFs are negligible.

Given ϵ_{max}, the objective is for the AUV to provide full coverage of the exterior boundary surface of the 3D target structure.

5.4.2 *Choosing an Appropriate Offset Distance*

A key point of our method is the choice of the offset distance Ω in the offline coverage path planning phase. Of course, Ω must be greater than the vehicle's radius to avoid collisions, assuming a rigid vehicle modeled as a sphere. Ω must also lie within the payload sensors range limits. But, as we will discuss below, for our realtime replanning strategy to succeed Ω must be chosen such that for an error $|\epsilon_t| \leqslant \epsilon_{max}$, the coverage path does not intersect the target surface. This has two implications. On one hand, Ω must be greater than ϵ_{max}. On the other hand, sufficient clearance between our target structure and potential neighboring structures is required.

5.4.3 *Realtime Replanning*

Once a nominal coverage path has been planned off-line, we propose an iterative replanning method to adapt the path in realtime according to range sensor information. To obtain a convenient representation of the environment for our replanning method we incrementally construct and maintain a 3D map of the target structure onboard the vehicle in realtime using range data. As in the sensor-based approach described in Section 5.3, we use the Octomap (Hornung et al., 2013) probabilistic mapping framework. At each iteration, our algorithm operates on the section of the nominal path yet to be processed within a given range from the robot. That piece of the nominal path is then reshaped using a trajectory optimization algorithm that, given an appropriate cost function, produces a smooth trajectory that keeps the vehicle at the desired offset distance from the actual target structure. The vehicle then executes the optimized trajectory. The process repeats until the end of the nominal path is reached. Next, we first describe the trajectory optimization algorithm (5.4.3.1) and the cost function we use in the optimization process (5.4.3.2). Then, building upon those two elements, we detail our coverage path replanning algorithm (5.4.3.3).

5.4.3.1 *Stochastic Trajectory Optimization*

We use the Stochastic Trajectory Optimization for Motion Planning (STOMP) algorithm (Kalakrishnan et al., 2011) to reshape the nominal coverage path so it adapts to the actual target structure perceived on site via onboard sensors. STOMP explores the space around an initial trajectory by generating noisy trajectories, which are then combined to produce an updated trajectory with lower cost in each iteration. Consider the example in Figure 72, where a cost designed to repel obstacles is used. At each iteration, the trajectory is updated to obtain a lower cost, achieving the effect of keeping it away from the obstacle (shaded in blue in Figure 72).

STOMP optimizes a cost function based on a combination of smoothness and application-specific costs, such as obstacles (as in the example above), constraints, or motor torques. An important characteristic of this algorithm is that it does not use gradient information, and so general costs for which derivatives are not available can be included in the cost function.

STOMP considers trajectories of a fixed duration, T, discretized into N waypoints equally spaced in time. For simplicity in the notation, we describe the algorithm for a 1-dimensional trajectory in Algorithm 4; this extends naturally to multiple dimensions. This 1-dimensional trajectory is represented as a vector $\theta \in \mathbb{R}^N$. The algorithm takes as input the start and goal positions (which are kept constant during the optimization process), an initial trajec-

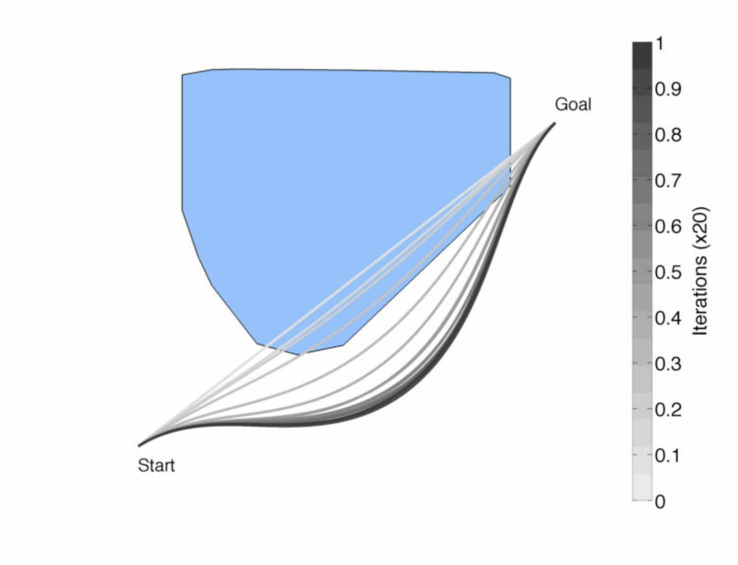


Figure 72: Example execution of the STOMP algorithm.

tory from start to goal (which can be as simple as a straight line) and a cost function (which we detail below in 5.4.3.2 for our case).

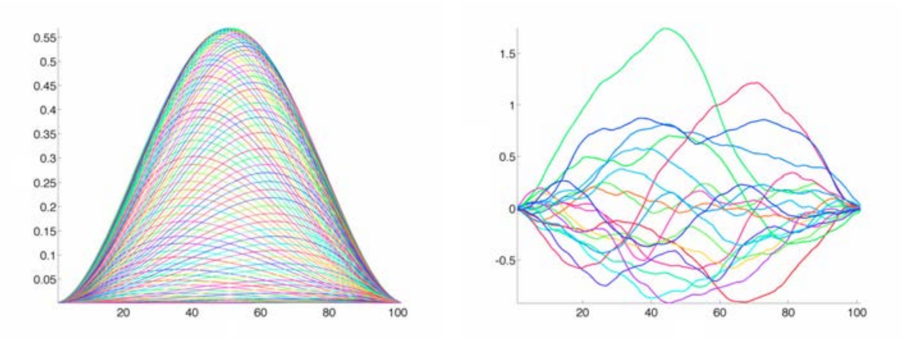
STOMP represents smoothness costs as a positive semi-definite matrix θ, such that $\theta^\top \mathbf{R}\theta$ is the sum of squared accelerations along the trajectory. The accelerations are obtained by means of a second-order finite difference matrix that when multiplied by the position vector θ, produces accelerations $\ddot{\theta}$:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & & 0 & 0 & 0 \\ -2 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & -2 & 1 & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & & 1 & -2 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & -2 \\ 0 & 0 & 0 & & 0 & 0 & 1 \end{pmatrix} \tag{20}$$

$$\ddot{\theta} = \mathbf{A}\theta \tag{21}$$

$$\ddot{\theta}^\top \ddot{\theta} = \theta^\top (\mathbf{A}^\top \mathbf{A})\theta. \tag{22}$$

Selecting $\mathbf{R} = \mathbf{A}^\top \mathbf{A}$ ensures that $\theta^\top \mathbf{R}\theta$ respresents the sum of squared accelerations along the trajectory. In each iteration of STOMP (Algorithm 4), first a set of K noisy trajectories is generated by sampling the noise from a zero mean normal distribution with $\mathbf{R}^{-1}$ as its covariance matrix (line 2). Figure 73 shows a representation of the $\mathbf{R}^{-1}$ matrix and of the noisy trajectories generated to explore the space around the initial trajectory. This keeps the generated trajectories smooth and does not allow them to diverge from the start or goal.



(a)

(b)

Figure 73: STOMP's trajectory exploration. (a) Each curve depicts a column/row of the symmetric matrix $\mathbf{R}^{-1}$. (b) 20 random samples of $\boldsymbol{\epsilon}$, drawn from a zero mean normal distribution with covariance $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}} = \mathbf{R}^{-1}$.

For each trajectory, its cost per time-step $S(\tilde{\theta}_{k,i})$ is computed (line 4). Based on this cost, a probability $P(\tilde{\theta}_{k,i})$ is assigned to each trajectory, per time-step (line 5). The parameter λ regulates the sensitivity of the exponentiated cost, and is optimized to maximally discriminate between the experienced costs by computing the exponential term in line 5 as:

$$e^{-\frac{1}{\lambda}S(\tilde{\theta}_{k,i})} = e^{-h\frac{S(\tilde{\theta}_{k,i})-\min S(\tilde{\theta}_{k,i})}{\max S(\tilde{\theta}_{k,i})-\min S(\tilde{\theta}_{k,i})}}, \tag{23}$$

with $h = 10$ as suggested by Kalakrishnan et al. (2011). The update for each time-step is computed in line 7 as the probability-weighted combination of the noisy trajectories for that time-step. In line 8 the update is smoothed using the $\mathbf{M}$ matrix, which is formed by scaling each column of $\mathbf{R}^{-1}$ such that the largest element in the column has magnitude $1/N$. Multiplication by $\mathbf{M}$ ensures that the updated trajectory remains smooth. Finally, the trajectory parameter vector is updated in line 9 and the cost for the updated trajectory is computed in line 10.

5.4.3.2 *Cost Function*

Our cost function seeks to keep all the points in the optimized trajectory at the desired offset distance Ω from the target structure. The distance between a point x and the boundary surface of the target structure S is the shortest distance between x and all points s_i in S. Such distance is given by the following function:

$$d(x, S) = \min_{s_i \in S} ||x - s_i|| \,. \tag{24}$$

Algorithm 4: STOMP

Input:

- Start and goal positions x_0 and x_N.
- An initial 1-D discretized trajectory vector θ.
- A state-dependent cost function $q(\theta_i)$.

Precompute:

- $\mathbf{A}$: second-order finite difference matrix (Equation 20).
- $\mathbf{R}^{-1} = (A^\top A)^{-1}$.
- $\mathbf{M} = \mathbf{R}^{-1}$, with each column scaled such that the maximum element is $1/N$.

```
while not convergence of trajectory cost $Q(\theta)$ do
    Create K noisy trajectories, $\tilde{\theta}_1, \ldots, \tilde{\theta}_K$ with parameters $\theta + \epsilon_k$, where $\epsilon_k \sim \mathcal{N}(0, \mathbf{R}^{-1})$
    for $k = 1 \ldots K$ do
        $S(\tilde{\theta}_{k,i}) \leftarrow q(\tilde{\theta}_{k,i})$
        $P(\tilde{\theta}_{k,i}) \leftarrow \frac{e^{-\frac{1}{\lambda}S(\tilde{\theta}_{k,i})}}{\sum_{l=1}^{K}[e^{-\frac{1}{\lambda}S(\tilde{\theta}_{l,i})}]}$
    for $i = 1 \ldots (N-1)$ do
        $[\delta\tilde{\theta}]_i \leftarrow \sum_{k=1}^{K} P(\tilde{\theta}_{k,i})[\epsilon_k]_i$
    $\delta\theta \leftarrow \mathbf{M}\delta\tilde{\theta}$
    $\theta \leftarrow \theta + \delta\theta$
    Update trajectory cost $Q(\theta) = \sum_{i=1}^{N} q(\theta_i) + \frac{1}{2}\theta^\top \mathbf{R}\theta$
return $\theta$
```

We define our cost function so it penalizes the difference between the distance from the trajectory points to the target surface and the desired offset distance:

$$q(\theta) = \sum_{t=0}^{T} |d(\theta_t, S) - \Omega|, \tag{25}$$

where $d(\theta_t, S)$ is calculated according to the current map. Recall that the additional smoothness cost $\theta^\top \mathbf{R}\theta$ is already incorporated in Algorithm 4.

5.4.3.3 *Realtime Coverage Path Replanning Algorithm*

We propose an iterative realtime replanning algorithm that uses range sensor data to reshape the nominal path to the actual target structure perceived *in situ*. The resulting path is smooth and keeps the desired offset distance Ω from the target structure. Recall that our replanning algorithm assumes that for an error $|\epsilon_t| \leqslant \epsilon_{max}$, the nominal coverage path does not intersect the actual target surface. The algorithm reshapes, in each iteration, the section of the nominal path yet to be processed within a range R from the vehicle's position. The magnitude of R must be smaller than the maximum sensor range used to perceive the target structure since the environment is still unknown beyond that limit. Once optimized, the vehicle executes the path assuming the environment's map representation does not change for the duration of the path execution. Therefore, R can be chosen to regulate the length of the replanning "steps" so newly incorporated range measurements providing a more up-to-date map can be taken into account early in the next iteration.

Algorithm 5 details our realtime coverage path replanning algorithm. In each iteration, the algorithm takes the section of the nominal path composed of all unprocessed waypoints within the given range R from the vehicle (lines 4-8). Next, an initial trajectory is built based on this path section (line 9). We do so by first building an initial geometric path. To construct this initial geometric path, the last waypoint (the most distant from the vehicle) in the current nominal path section is projected along the surface normal so it lies at the desired distance Ω from the target structure. This step is necessary because the goal of the initial trajectory remains constant during the optimization process. Then, the initial path is composed by: 1) a straight line connecting the current vehicle position to the first waypoint of the current path section; 2) the current path section itself; and 3) a straight line connecting the last waypoint of the current path section to its projection along the surface normal. This initial path is then discretized into time-steps to obtain an initial trajectory. This initial trajectory generation procedure is illustrated in Figure 74.

Next, the initial trajectory is optimized using the STOMP algorithm (line 10). The optimization takes place in the vehicle's horizontal (X-Y) plane, leaving the vertical (Z) coordinates of the nominal path unchanged. The current map $\mathcal{M}$ is passed as an argument to compute the cost function given in

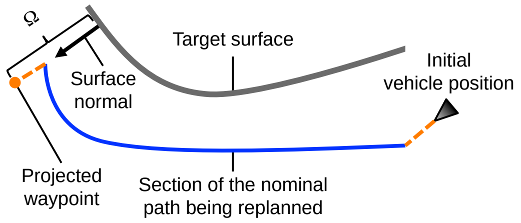


Figure 74: Illustration of the initial trajectory construction in a replanning step (top view).

Equation 25. Finally, the optimized trajectory is executed (line 11) and the process repeats until the end of the nominal path is reached.

Algorithm 5: Realtime Coverage Path Replanning

Input:

- Nominal coverage path as a list of K waypoints $w_0 \dots w_K$.
- Current environment's map, $\mathcal{M}$.
- Replanning step range, R.

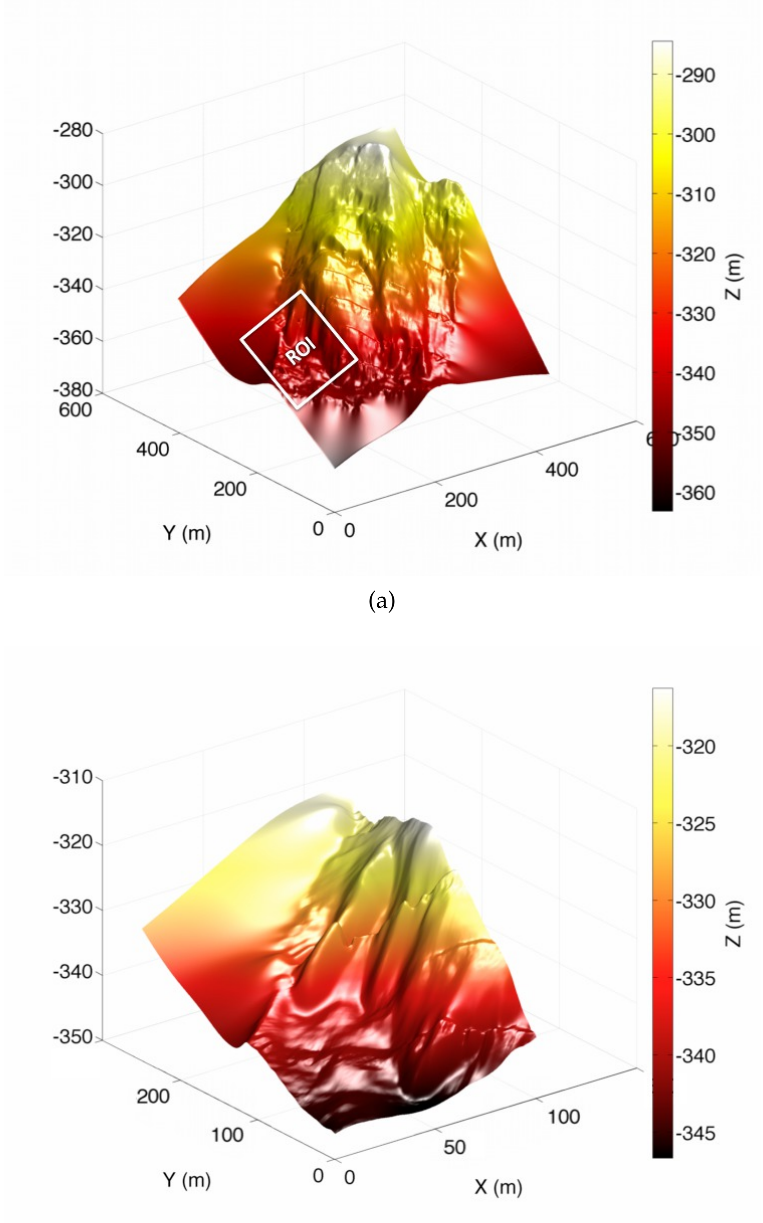
```
1  Navigate to initial waypoint $w_0$
2  $i \leftarrow 0$
3  while $i < K$ do
4      $x \leftarrow$ GetRobotPosition()
5      $pathSection \leftarrow \emptyset$
6      while Distance($x$, $w_i$) $< R$ and $i < K$ do
7          $pathSection$.append($w_i$)
8          $i \leftarrow i + 1$
9      $\theta \leftarrow$ InitialTrajectory($pathSection$, $x$)
10     $optimizedTrajectory \leftarrow$STOMP($\theta$, $\mathcal{M}$)
11     Execute($optimizedTrajectory$)
```

5.5 EXPERIMENTAL OUTCOMES

We next present experimental outcomes to validate the 3D coverage path planning techniques introduced above. Section 5.5.1 discusses results obtained with our off-line 3D coverage path planning method only, Section 5.5.2 discusses results obtained with our sensor-based profile following approach and Section 5.5.3 discusses results obtained with our realtime replanning approach. Finally, Section 5.5.4 presents a combination of all these techniques applied to the same target scenario in both simulation and real-world experiments to achieve complete coverage.

5.5.1 *Off-line 3D Coverage Outcomes*

We validate our off-line 3D coverage path planning technique using the Santorini caldera dataset introduced in Section 4.3.1.1. A 100 m by 250 m Region Of Interest (ROI) on the mapped area is selected for further in-detail inspection. The selected ROI comprises the boundary of the caldera, and is therefore of high geological interest. Figure 75 shows the bathymetric maps of the entire mapped area and of the selected ROI.



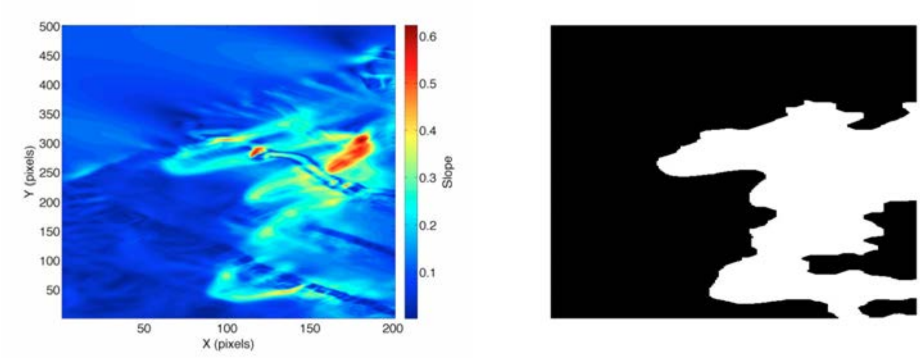
(a)

(b)

Figure 75: 3D view of the Santorini caldera bathymetric dataset: (a) complete bathymetric map of the Santorini caldera with the selected ROI indicated by its approximate bounding box; (b) bathymetric map of the selected ROI.

We apply our proposed 3D coverage path planning technique to the selected ROI of the bathymetric map introduced above. The objective is to obtain a coverage path on the ROI at an offset distance $\Omega = 2\,\text{m}$ from the target surface to collect imaging data with the GIRONA 500 AUV. (Recall the reader can refer to Appendix A for an introduction to the GIRONA 500 AUV.) It is assumed that the vehicle uses a 60° aperture angle camera to image the surface.

The slope map, $S(x, y)$, of the ROI is shown in Figure 76a, with values ranging between 0.001 and 0.622. The slope map is classified using a threshold $\delta_s = 0.5$, which yields a single high-slope region after applying the described morphological operations as shown in Figure 76b.



(a)

(b)

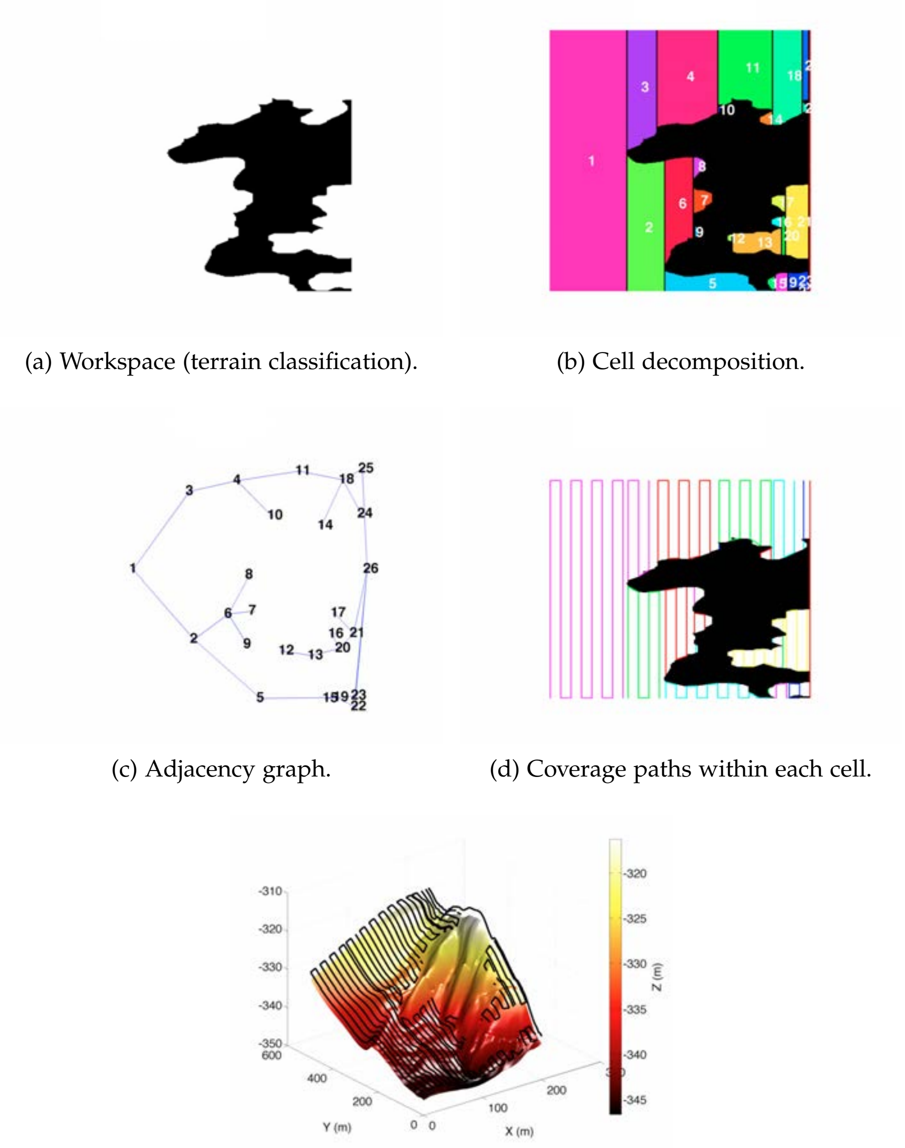
Figure 76: Slope map of the Caldera 2012 bathymetric dataset (a) and its terrain classification showing a single high-slope region (b).

The slicing and boustrophedon decomposition algorithms are then applied to the corresponding regions. Figure 77 shows the boustrophedon decomposition process on the effectively planar areas of the Caldera 2012 scenario, while Figure 78 illustrates the slicing algorithm process.

The resulting coverage path after combining the outcome from the boustrophedon decomposition algorithm with that of the slicing algorithm is shown in Figure 79, while a standard lawnmower-type path is shown in Figure 80. The algorithms are implemented in unoptimized MATLAB and generate the full coverage path in less than 5 seconds on a standard PC.

To obtain a quantitative evaluation of the coverage path, we compare the path obtained using our bathymetric coverage path planning technique and a standard constant-altitude lawnmower-type survey path in Table 3, where the path lengths and average viewing angles for each region, for the full coverage path and for the lawnmower survey path are reported. The viewing angles are calculated as shown in Figure 56, assuming a down-looking sensor is used to cover the effectively planar areas and that a forward-looking sensor is used to cover the high-slope areas (a down-looking sensor is assumed for the standard lawnmower-type path). Recall that the ideal viewing angle is 0°.

Note that, although the standard lawnmower-type survey path is shorter, it incurs a significantly higher average viewing angle. This is mainly due to



(a) Workspace (terrain classification).

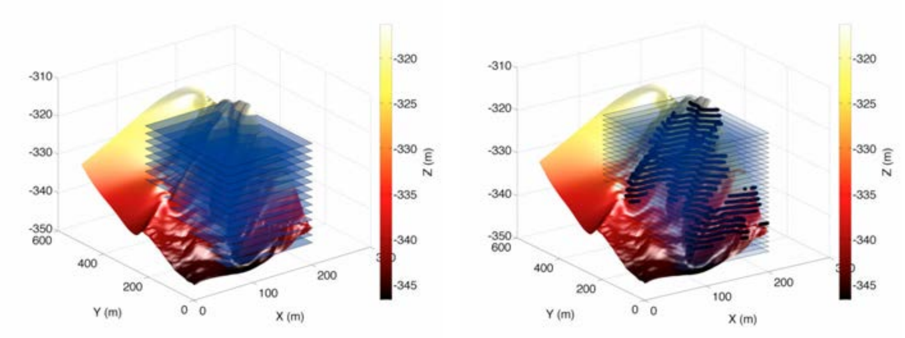
(b) Cell decomposition.

(c) Adjacency graph.

(d) Coverage paths within each cell.

(e) Coverage path for effectively planar terrain on the bathymetric map.

Figure 77: Application of the Morse-based boustrophedon decomposition algorithm for coverage of effectively planar areas on the Caldera 2012 scenario.



(a) Slicing planes.

(b) Intersection edges.

Figure 78: Application of the slicing algorithm for coverage of high-slope areas on the Caldera 2012 scenario.

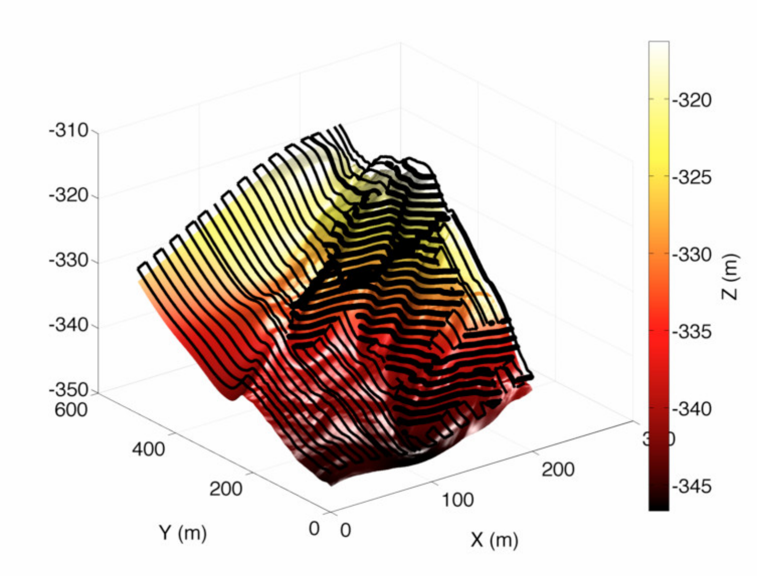


Figure 79: In-detail 3D coverage path for the selected ROI of the Caldera 2012 dataset. The coverage path for the high-slope region is indicated with thicker lines.

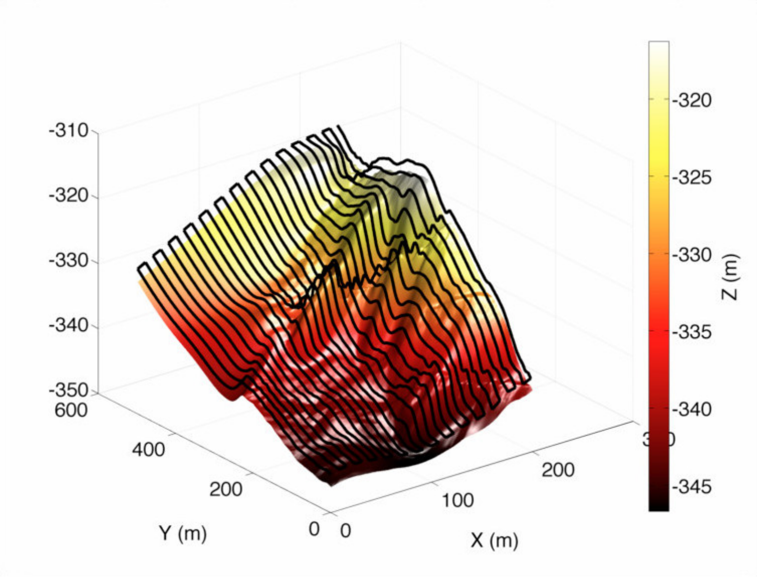


Figure 80: Standard lawnmower-type coverage path for the selected ROI of the Caldera 2012 dataset.

the sudden depth changes the vehicle needs to tackle when covering high-slope terrain following this path. By contrast, our method generates a path with a lower viewing angle, suiting better imaging applications.

PATH	PATH LENGTH	AVERAGE VIEWING ANGLE
High-slope region	5864.9 m	9.3° $(\sigma = 4.5°)$
Planar region	4973.6 m	12.3° $(\sigma = 6.6°)$
Total (both regions)	10838.5 m	10.8° $(\sigma = 5.6°)$
Standard lawnmower path	9834.2 m	23.2° $(\sigma = 7.1°)$

Table 3: Coverage path length and viewing angles comparing our coverage method and a standard lawnmower-type survey path.

5.5.2 *3D Coverage with Sensor-based Profile Following Outcomes*

Next, we show results of our profile following method obtained both in simulation and in pool trials with the GIRONA 500 AUV. (Recall the reader can refer to Appendix A for an introduction to the GIRONA 500 AUV.) First, we show results obtained in simulation of our vertical profile following scheme on a man-made object model. We test our horizontal profile following scheme in a simulated 3D coverage task of an underwater boulder. We use a model of a real-world underwater boulder built from a bathymetric dataset autonomously recorded by GIRONA 500 in the target area.

We finally report on two in-water trials conducted in our lab's pool, where GIRONA 500 follows the vertical and horizontal profiles of the pool.

For the simulation experiments we use the UWSim (Prats et al., 2012a) underwater robotics simulation package, which provides a high-fidelity simulation environment. We import into UWSim a dynamic model of GIRONA 500 and a rotating pencil-beam sonar sensor model after the Tritech's SeaKing profiler GIRONA 500 is equipped with (see Appendix A). The very same software architecture which runs on GIRONA 500 during sea trials is used in conjunction with UWSim to carry out the simulation, thus allowing for seamless transition from simulation to real-world missions.

5.5.2.1 *Simulated Vertical Profile Following*

Figure 81 shows the trajectory traced by GIRONA 500 when using the proposed method to follow the vertical profile of a man-made object model. The desired offset distance is 2 m. As can be observed in Figure 81, by exploiting the hovering capability of GIRONA 500 our profile following method is able to successfully maneuver the vehicle to completely follow the "C" profile of the object. Due to the presence of a totally vertical profile traditional bottom following techniques would not be able to handle it, leading to a collision threat for the vehicle. Our method favors hovering-capable AUVs, being able survey structures in closer proximity than traditional bottom following surveys typically used in torpedo-shaped vehicles. Note that the robot makes a 180° turn to avoid navigating backwards when the ceiling following behavior is engaged.

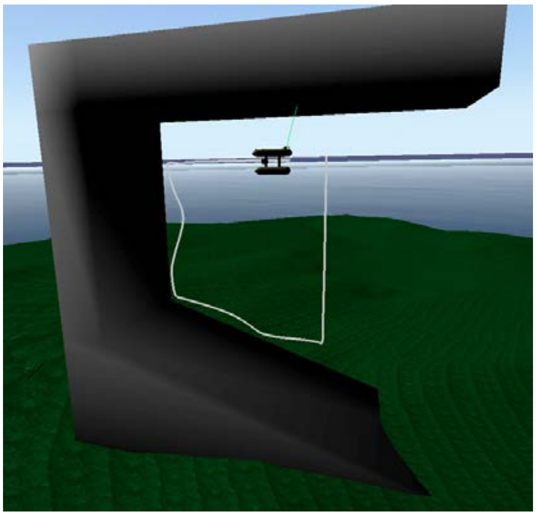
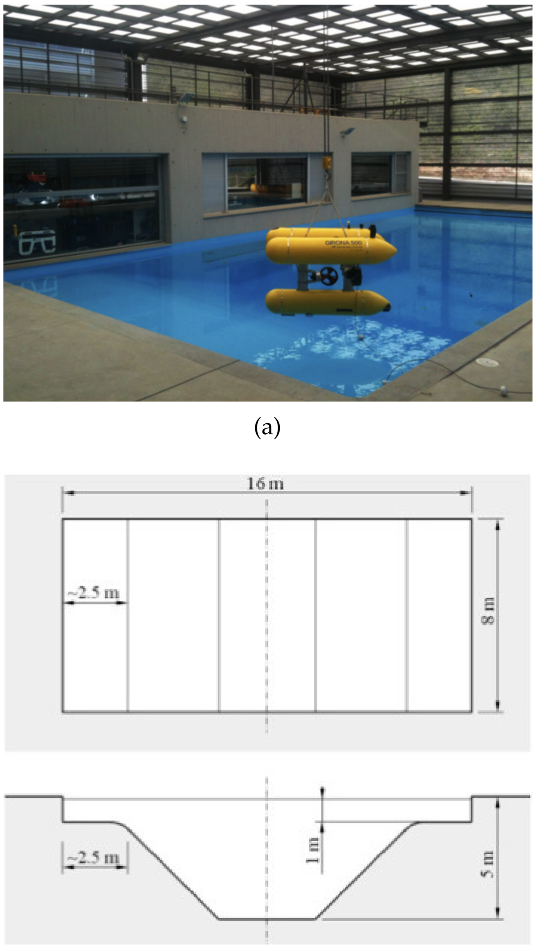


Figure 81: Trajectory traced by GIRONA 500 while following the vertical profile of a man-made object in simulation.

5.5.2.2 *In-Water Profile Following Trials*

We tested our method in a real-world setting with the GIRONA 500 AUV following the vertical and horizontal profiles of our pool at CIRS. As shown in Figure 82, the pool has a "V" shaped bottom. A supervision room with a window under the water level allows to keep track of the AUV operation.



(a)

(b)

Figure 82: GIRONA 500 at the CIRS facility of the University of Girona. (a) and a blueprint of the testing pool therein (b).

The horizontal profile following module was tested at a constant 4 m depth. Figure 83 shows a sequence of photos of GIRONA 500 following the horizontal profile of the pool, seen from the supervision room. The frontal distance and slope errors of two complete circumnavigations of the pool's horizontal profile are shown in Figure 84. Despite the sharp 90° corners in the pool's profile, our profile following method is able to keep the estimated front distance within 1 m of the desired nominal distance.

The vertical profile following module was tested along the profile of the pool. Figure 85 shows the estimated frontal distance to the profile (in the cliff region) during the vertical profile following in the pool. The AUV started in the shallow region of the pool (left of Figure 82b, moving to the right), where the pencil-beam sonar faced both the opposite ramp and the farthest wall of the pool. As a result, sonar energy returns from said ramp and wall lead to a spread occupied area on the local map and, consequently, to a poor estimation of the actual profile. Effectively, returns from the wall and from the

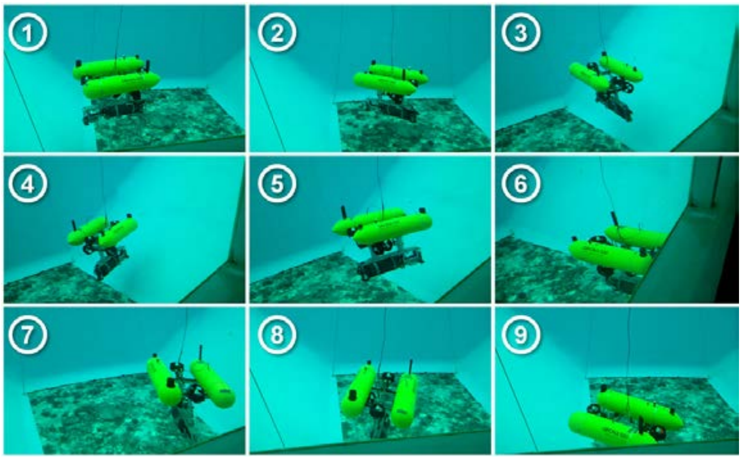
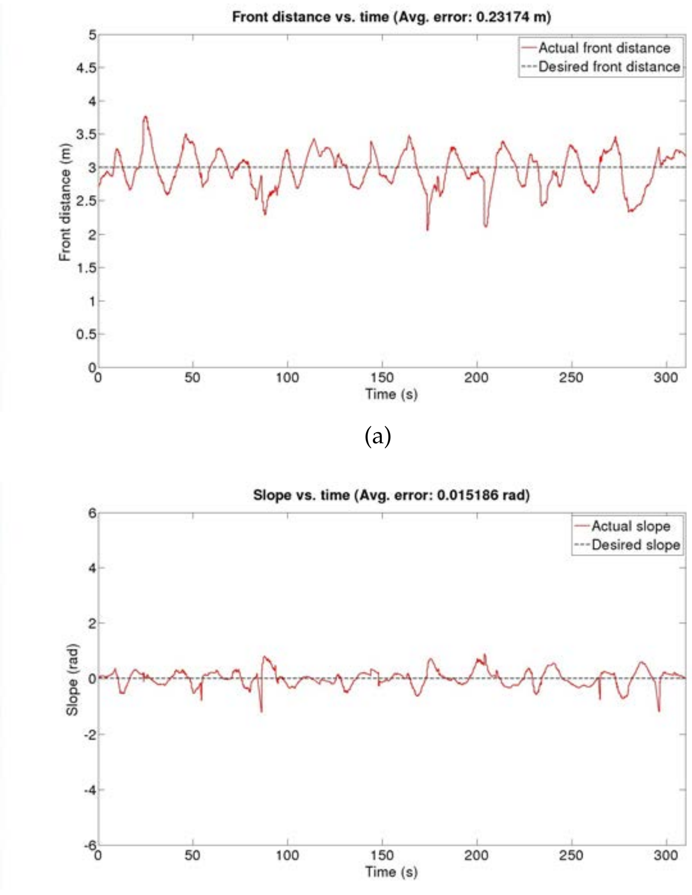


Figure 83: Photo sequence (left to right, top to bottom) of GIRONA 500 following the horizontal profile of our lab's pool.



(a)

(b)

Figure 84: In-pool horizontal profile following: frontal distance (a) and slope (b) vs. time.

ramp were being confused in the vehicle's perception. Due to this situation, a region of high error appears in Figure 85 during the first 20 seconds of the experiment. Once the vehicle reaches the first ramp on the pool's profile and the vehicle dives the sonar measurements start to match the actual environment and the vehicle can adjust its distance to the estimated profile. For the same reason, at second 40 another error peak arises when the vehicle reaches the end of the second ramp. The error in this experiment could have been reduced by using a pencil-beam sonar with a narrower beam, providing a more accurate mapping of the profile. We note, however, that due to the relatively big size of GIRONA 500 in comparison with the profile followed and the non-convex and confined environment this was a challenging profile following task.

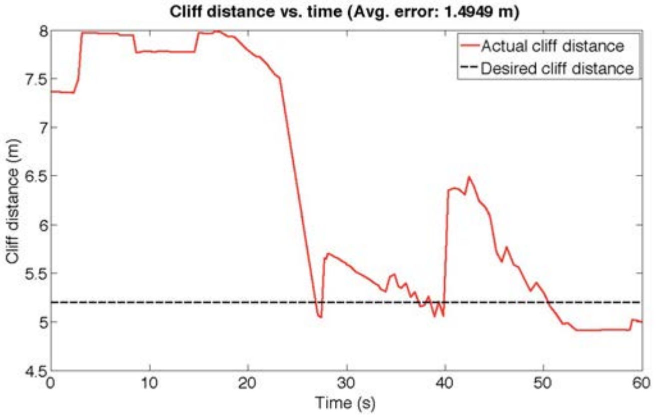


Figure 85: In-pool vertical profile following: distance to cliff (frontal wall) vs. time.

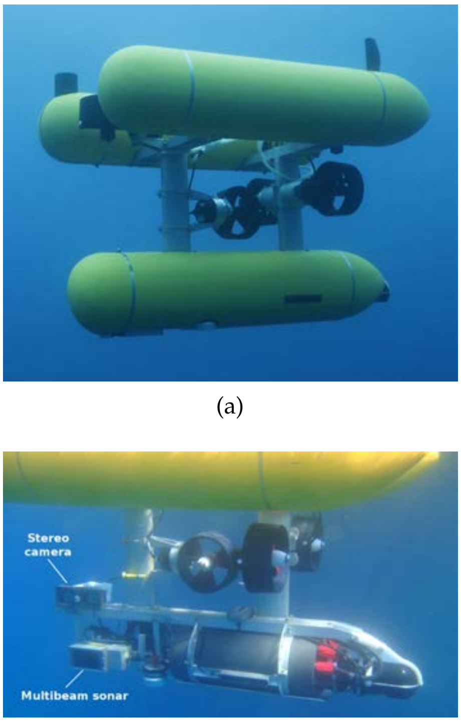
5.5.3 *3D Coverage with Realtime Replanning Outcomes*

Our replanning algorithm has been implemented in Python and integrated with the rest of GIRONA 500's software architecture using the Robot Operating System (ROS) framework (ROS, 2013). The implementation produces an optimized path at each replanning step in less than a second. To present our results, we replay the mission logs using Rviz, the visualization package provided in the ROS framework. We have tested our method by performing two coverage tasks with the GIRONA 500 AUV at sea[†]. In the first task we cover a concrete block of a breakwater structure in a harbor. In the second task, we cover "l'Amarrador" underwater boulder, already targeted by the experiments in Chapter 4. Next we present the minimum validation results obtained in the concrete block scenario, while the results involving the underwater boulder scenario are presented in the next section below.

The *a priori* bathymetric charts used to plan the nominal coverage paths were created by members of CIRS using a vessel equipped with GIRONA 500's Delta T multibeam sonar. GIRONA 500's pencil-beam and multibeam sonars were used to perceive the target structure with the maximum range

[†] A video showcasing these experiments can be found at http://www.youtube.com/watch?v=2REWf6jbdZ0

set at 20 m. (The reader can refer to Appendix B for more details on the sonar technology used in these experiments.) The replanning range is set at $R = 5$ m. The stereo camera and multibeam sonar of GIRONA 500, which were mounted in a side-looking manner, were used as payload sensors to gather both optical and range data from the structures. The GIRONA 500 AUV is shown in Figure 86 equipped with the sensors used in these sea trials. In both experiments, we estimate to be dealing with a maximum GPS error smaller than 5 m, whereas the estimation for the total maximum error ϵ_{max} varies depending on each particular setup, as we detail below.



(a)

(b)

Figure 86: The GIRONA 500 AUV during the 3D coverage with replanning sea trials. (a) Standard configuration with the pencil-beam sonar protruding on the top-left. (b) Payload configuration with side-looking multibeam sonar and stereo camera.

We validate the benefits of our method for 3D mapping using both bathymetric and optical techniques. Our 3D mapping results show how the paths planned with our method are useful in mapping complex 3D structures, not amenable for traditional 2.5D mapping of marine environments. On one hand, the unorganized range data collected by the multibeam sonar promotes the use of 3D surface reconstruction techniques. More precisely, we apply the screened Poisson method (Kazhdan and Hoppe, 2013) to recover a triangle mesh resembling the surface described by the range data. On the other hand, we apply another mapping technique based on optical data only. Using solely images, we follow a sequential pipeline composed by a camera trajectory estimation via structure from motion (Nicosevici et al., 2009), followed by a dense point set sampling through multiple-view stereo (Yang and Pollefeys, 2003), the surface reconstruction (Kazhdan and Hoppe, 2013) and a final per-vertex texture mapping. All the data products we show are the direct result of the automatic 3D mapping techniques we use, without any manual tuning or refinement. The 3D maps here presented were generated by colleague Ricard Campos from the Underwater Vision Lab of the Computer Vision and Robotics Institute at the University of Girona, using the data collected with our coverage path planning technique in the sea trials. They are included and discussed in this thesis for the purpose of demonstrating the effectiveness of our 3D coverage path planning method in obtaining rich data for 3D mapping applications.

As previously mentioned, the first coverage task in which we test our method serves as a minimal test of our implementation. The target structure is a concrete block of a breakwater structure composed of twenty of such blocks. Each block's footprint is approximately 5 m × 5 m, spanning from 2 m above the surface down to the sea bottom at 10 m depth. This structure is in the harbor of Sant Feliu de Guíxols, in the Costa Brava in Catalonia (Spain). It is located next to its main pier and provides protection from the effects of bad weather and longshore drift. Figure 87 shows the *a priori* bathymetric chart (overlapped on satellite imaging) we use to plan a nominal coverage path (also shown in Figure 87) for this task using our off-line planning method. In this minimal validation experiment, we target the right-most block of the structure and we plan a coverage path of a single contour at 5 m depth, which will allow the multibeam sonar to image most of the in-water part of the block. Aiming to capture optical data of the structure, and since we deal with a somewhat controlled environment in this experiment, we use a relatively short offset distance $\Omega = 6.0$ m to plan the nominal path (hence assuming $\epsilon_{max} < 6.0$ m). The nominal path resulting from the off-line planning phase is also shown in Figure 88. Note that the path is not closed (it resembles a semi-circle) since there is not enough clearance between the concrete blocks for the vehicle to go through. Therefore, the path provides coverage on only three of the four vertical faces of the block.

The trajectory followed by the robot during the realtime replanning phase is shown in Figure 88 with the on-line map and the depth-colored raw range data acquired by the side-looking multibeam sonar. It can be observed that the map includes many outliers, mainly due to surface reflections of the pencil-beam sonar beams. Nonetheless, the resulting trajectory provides full sensor coverage of the targeted in-water part of the structure. Along this coverage trajectory, and according to the on-line map, the AUV kept a mean distance to the target structure of 7.71 m, with a standard deviation of 2.03 m.

Figure 89 shows the surface reconstructed from the raw range data. Note how the point cloud depicted in Figure 88 is far from ideal, as it contains high levels of noise and outliers coupled with registration errors. Prior to reconstruction, and since the method used (Kazhdan and Hoppe, 2013) requires oriented point sets, we computed per-point normals with the method of Hoppe (Hoppe et al., 1992), using a neighborhood of $k = 200$ points. De-

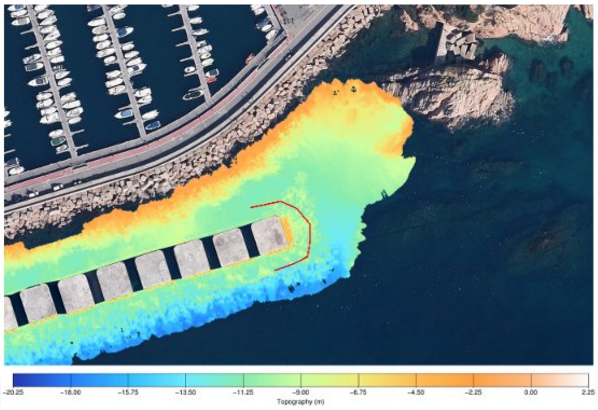


Figure 87: Bathymetric map of the area surrounding Sant Feliu harbor's breakwater structure overlapped on satellite imagery (provided by Google Earth). The nominal coverage path, targeting the right-most block of the breakwater structure, is shown in red.

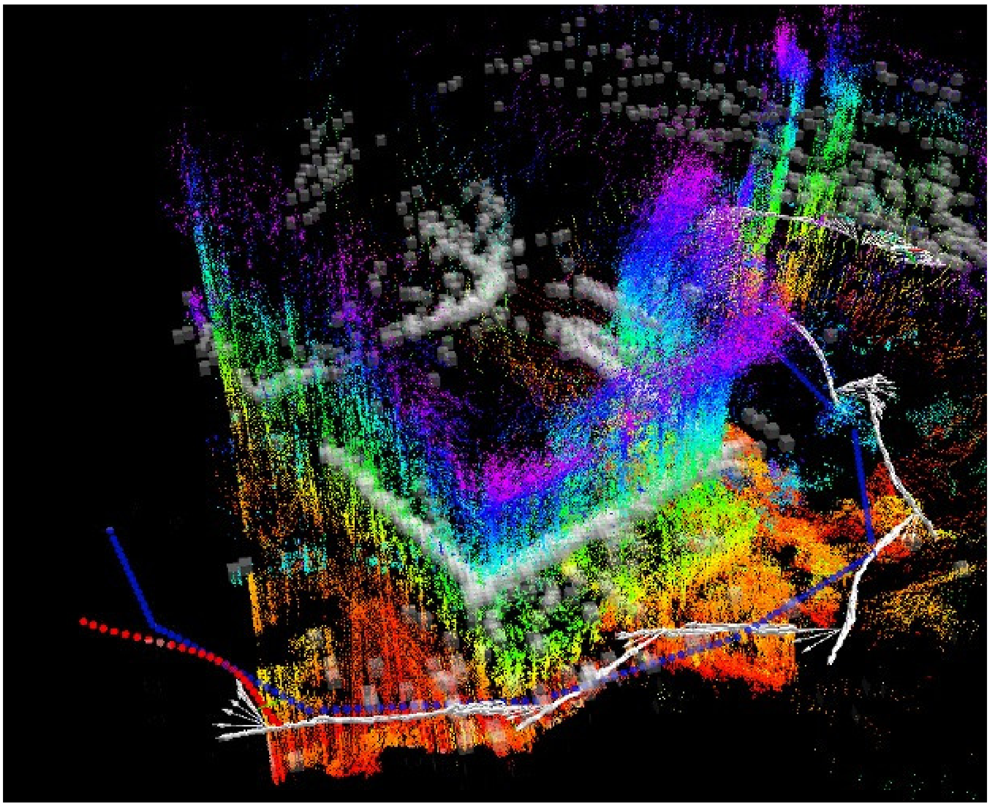
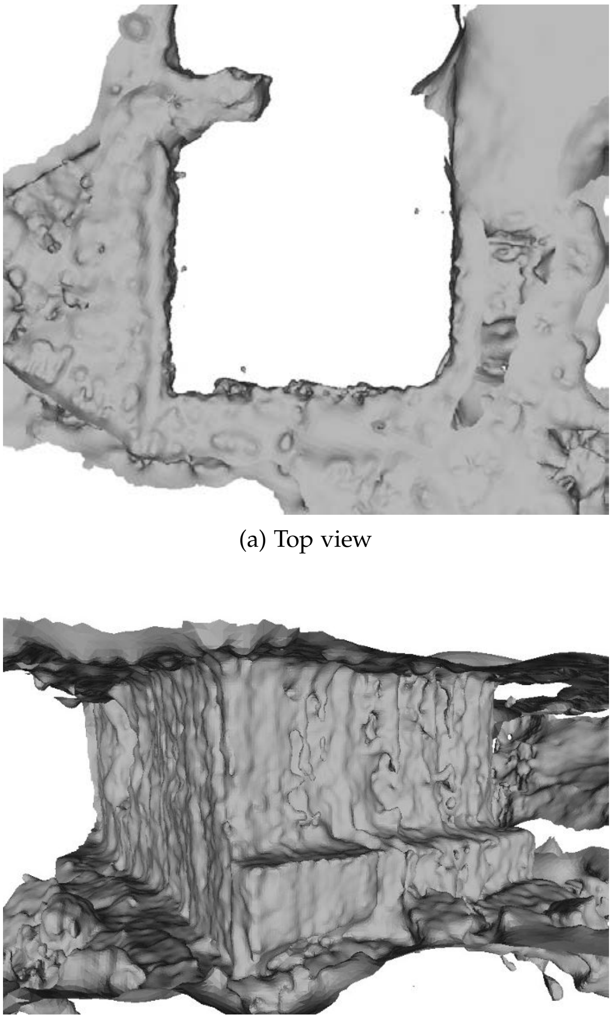


Figure 88: Realtime replanning on the concrete block coverage experiment at the last replanning step of the task: nominal coverage path (blue-dotted line); optimized trajectory which the robot is executing at that particular instant (red-dotted line); overall trajectory (white arrows); occupied cells in the on-line map (white cubes). The depth-colored range data acquired by the multibeam sonar is also displayed.

spite the defect-laden nature of the input data, the screened Poisson method is able to recover the surface with reasonable accuracy. However, data defects cause some non-existent artifacts to show up in the top- and bottom-most parts of the model and some undesirable roughness in its front wall.

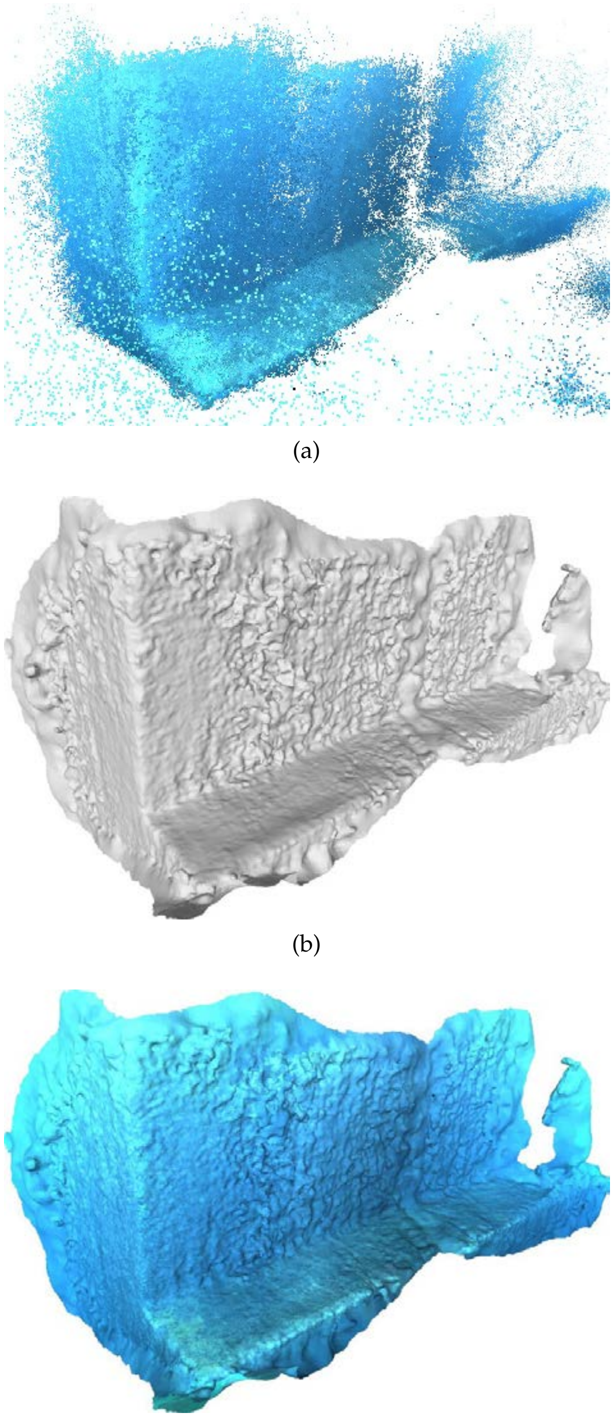


(a) Top view

(b) Slanted view

Figure 89: Surface reconstruction of the concrete block from range data. Image credit: Ricard Campos.

Finally, we used the on-board stereo camera to reconstruct a 3D model of the site using just optical data. Due to low visibility conditions brought about by water turbidity, we are only able to obtain a partial reconstruction of the structure. This produces a less complete model than its bathymetric counterpart. Additionally, for the same reason, the model suffers from small defects caused by accumulating cascading errors in the processing pipeline stages. Nevertheless, Figure 90 shows intermediate results in the pipeline stages and the final recovered model, where we can see how the front face of the block is reconstructed with more detail and less error than in the model obtained using range data.



(a)

(b)

(c)

Figure 90: Slanted views of the optical reconstruction on the concrete block dataset. (a) Dense point set with per-vertex texture mapping. (b) Surface. (c) Textured surface. Image credit: Ricard Campos.

5.5.4 *All Together: Testing at "l'Amarrador" Site*

We now show results obtained in simulation and in real-world experiments at l'Amarrador underwater boulder, a diving site off the Costa Brava in Sant Feliu de Guíxols, Girona, Catalonia (Spain). It is the same site targeted by the experiments reported in Chapter 4. Recall that the underwater boulder rises from 40 m depth up to 28 m, being approximately 12 m high. We will apply to this scenario, subsequently, our off-line 3D coverage path planning algorithm (Section 5.5.4.1), our profile following scheme (Section 5.5.4.2) and our realtime replanning algorithm (Section 5.5.4.3). The particular dataset used in these experiments for off-line planning and in simulation was collected by GIRONA 500 following a pre-planned standard survey path with a multibeam sonar at 5 m depth in April 2013. The bathymetric chart of the site obtained from the data gathered during the survey mission is shown in Figure 91. Each cell in the uniform grid composing the bathymetric model is 40-by-40 cm.

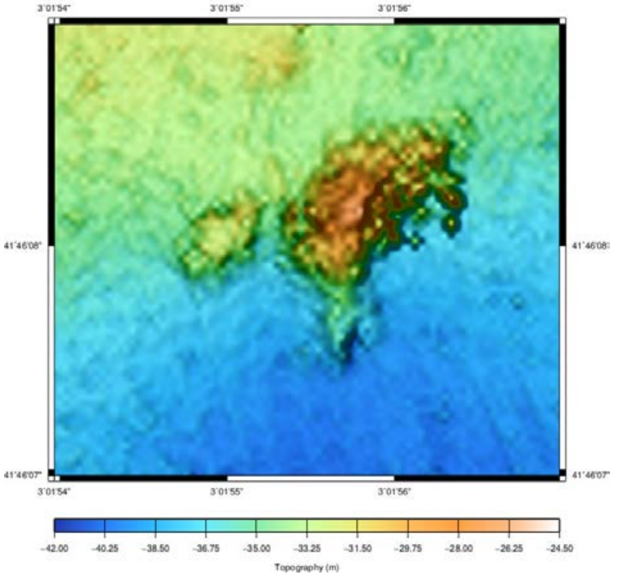
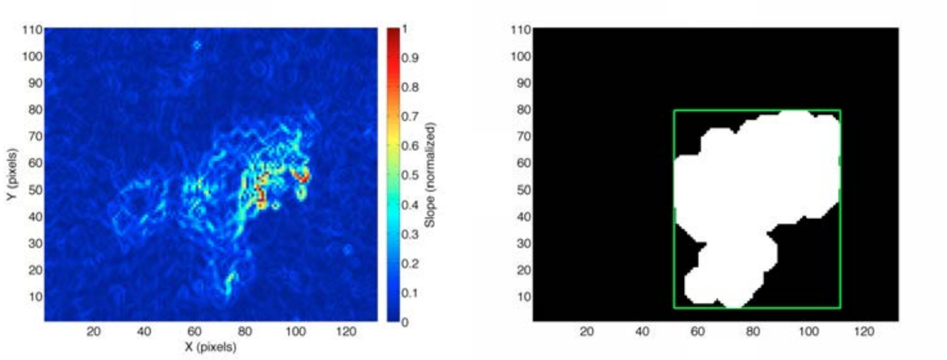


Figure 91: Bathymetric map of l'Amarrador site. Data were collected by GIRONA 500.

5.5.4.1 *Off-line 3D Coverage Path Planning at "l'Amarrador"*

We start by generating a coverage path for the entire site the technique introduced in Section 5.2. We use a desired offset distance $\Omega = 10$ m. The terrain classification for "l'Amarrador" site is shown in Figure 92. The application of the boustrophedon algorithm for coverage of effectively planar areas is illustrated in Figure 93, while Figure 94 shows the application of the slicing algorithm for coverage of the high-slope areas. The final off-line, nominal coverage path for the site is shown in Figure 95. The plan consists of 2 contours spaced 2 m apart in the vertical axis. This spacing provides redundant coverage, which is of interest for testing SLAM and 3D reconstruction algorithms.



(a) Slope map

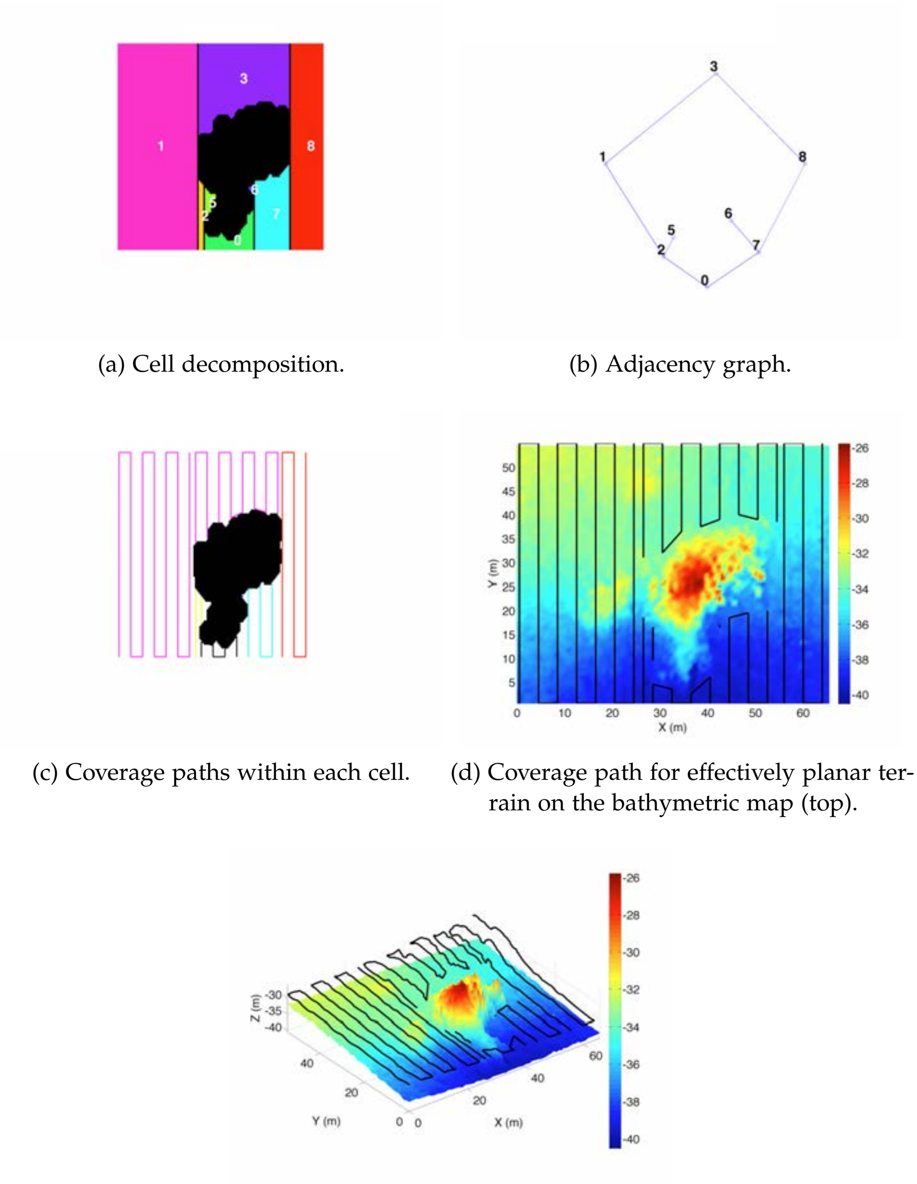
(b) Terrain classification

Figure 92: Slope map and terrain classification for “l’Amarrador” site.

5.5.4.2 *Simulated 3D Coverage using Profile Following and “l’Amarrador” Bathymetric Dataset*

Next, we demonstrate our profile following strategy at “l’Amarrador” in simulation using a model of the site and UWSim. Since this is a simulation in idealized sea state conditions, we create a coverage plan at a less conservative distance of $\Omega = 5$ m. To generate the plan, we use the slicing algorithm (Algorithm 3) to yield two cross-section profiles, the first at a constant 5 m depth below the summit of the underwater boulder and the second at 3 m below the summit. This off-line coverage plan is illustrated in Figure 96 with the triangle mesh model of the site used in simulation. The desired offset distance is $\Omega = 5$ m, and it is assumed that the vehicle is using a payload sensor with a 60° FOV, such as a typical camera.

The trajectory traced by GIRONA 500 during the profile-following-based coverage experiment is shown Figure 97 and Figure 98 shows the distance and slope errors during profile following in the diving site. The trajectory presents some sharp features induced by the low resolution and roughness of the 3D model of the underwater boulder used in the simulation. For this very same reason, the method eventually needs a few seconds to deal with a highly non-convex region of the profile. This situation arose around second 450 in our experiment (see Figure 98), where a high-frequency period of approximately 20 seconds takes place until the method is able to successfully detect a profile to the right of the vehicle and perform the 90° turn maneuver described in Section 5.3.2.1. We hypothesize that in the real-world site this situation is less likely to happen since the actual profile of the underwater boulder is smoother than that of the 3D model used in the simulation. Overall, this experiment shows how our method can successfully cover the target structure by following its complex profiles within 1.5 m of the nominal desired distance.



(a) Cell decomposition.

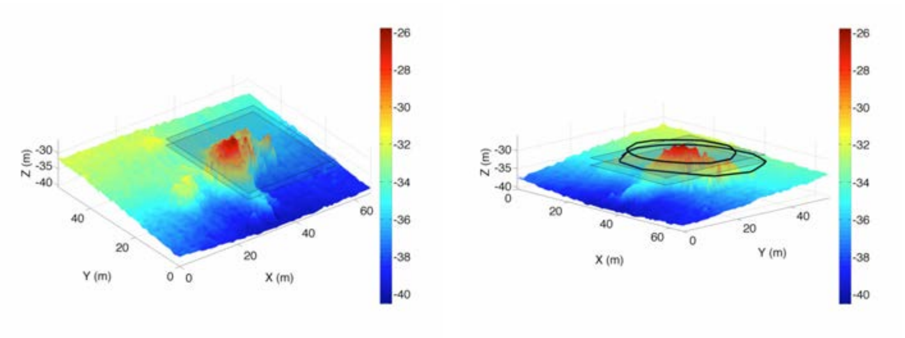
(b) Adjacency graph.

(c) Coverage paths within each cell.

(d) Coverage path for effectively planar terrain on the bathymetric map (top).

(e) Coverage path for effectively planar terrain on the bathymetric map (slanted).

Figure 93: Application of the Morse-based boustrophedon decomposition algorithm for coverage of effectively planar areas on the "l'Amarrador" scenario.



(a) Slice planes.

(b) Offset coverage edges.

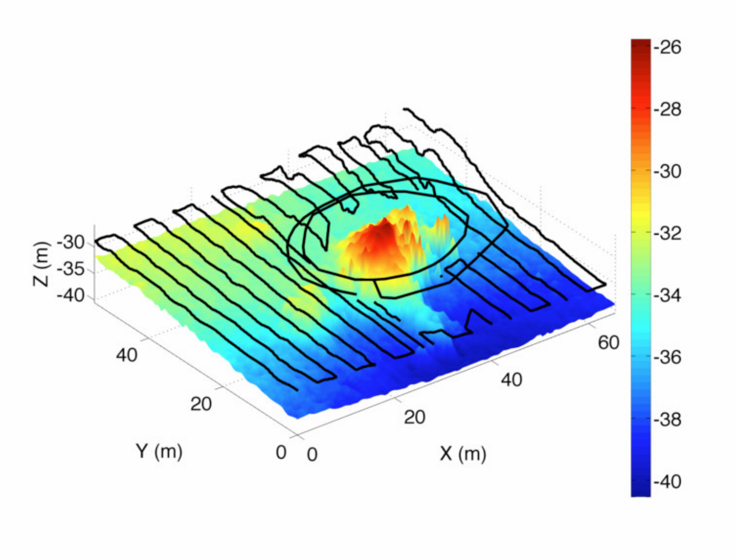
Figure 94: Application of the slicing algorithm for coverage of high-slope areas on the "l'Amarrador" scenario.

This experiment serves as a proof of concept of coverage of a protruding, rugged region of the seabed using our profile following method at incremental depths. Indeed, by following the trajectory obtained with our method the AUV is able to image the target surface using a forward-looking sensor. Since the profile following method orientates the vehicle along the profile normal, an appropriate viewpoint for imaging purposes is obtained. This contrasts with the askew angle of incidence obtained from an overhead viewpoint using traditional bottom profile following.

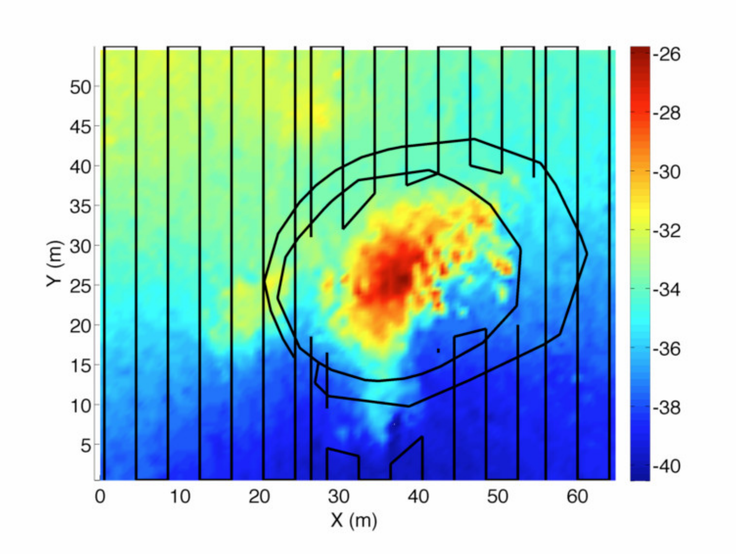
5.5.4.3 *Coverage of "l'Amarrador" Underwater Boulder with Realtime Replanning*

Next, we report on our second coverage with realtime replanning task targeting "l'Amarrador" underwater boulder. The nominal coverage path for the site is shown in Figure 95. There are two important factors to take into account when choosing an offset distance to plan this task. On one hand, this site is in an open sea environment and there exist a threat of strong currents. On the other hand, the mission is significantly longer, incurring a potentially bigger error due to dead-reckoning drift. For these reasons, we use a more conservative offset distance than in the previous task: $\Omega = 10$ m. Unfortunately, at this offset distance, the water turbidity conditions did not allow for optical imaging of the underwater boulder. Therefore, only the sonar range data is of interest in this experiment.

Figure 99 shows two instants of the realtime replanning phase together with the nominal path, which GIRONA 500 is reshaping so it agrees with the perceived sonar range data of the underwater boulder. As can be observed in the figure, the vehicle starts at the surface, dives down to the depth of the first coverage edge of the plan in a safe area, and starts the coverage task. Due to resource constraints, only the deepest coverage edge of the plan was executed in this experiment. Along the overall coverage trajectory in this experiment, and according to the on-line map, the AUV kept a mean distance to the target structure of 9.41 m, with a standard deviation of 0.93 m. The trajectory executed to cover the deepest coverage edge of the plan is



(a) Slanted view



(b) Top view

Figure 95: Off-line coverage plan for "l'Amarrador" site.

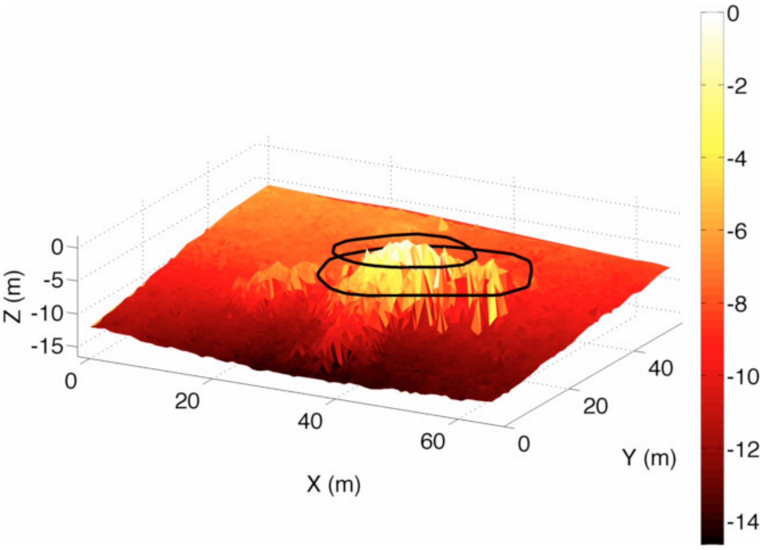
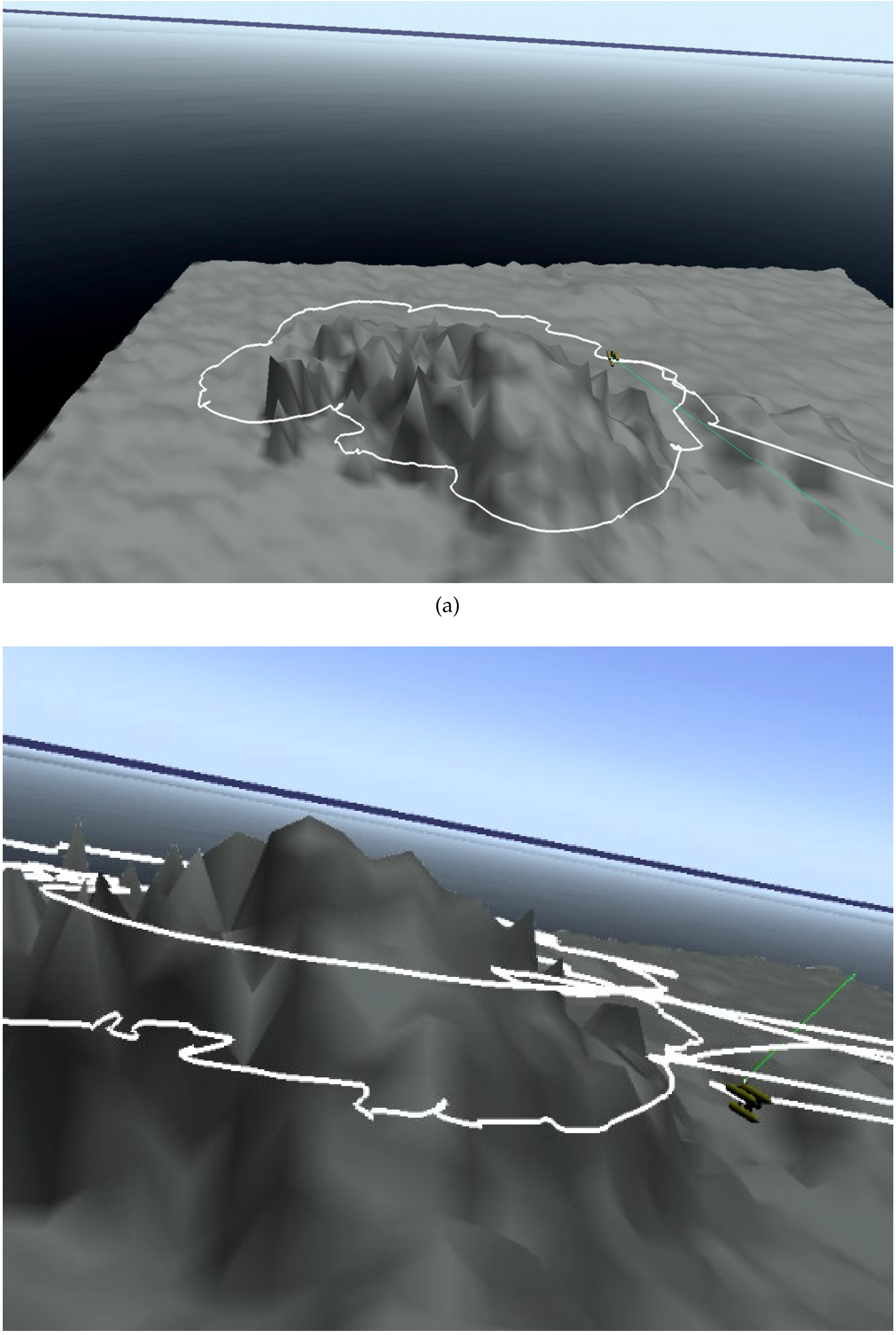


Figure 96: 3D triangle mesh model of l'Amarrador site with the off-line coverage plan for profile-following-based coverage consisting of 2 target cross-section profiles: one at 5 m below the summit and another at 3 m below the summit. The desired offset distance is $\Omega = 5$ m.

shown in comparison with the off-line plan in a top view in Figure 100, and in slanted views in Figure 101. Note how the coverage trajectory, by contrast with the off-line plan, adapts to the actual shape of the boulder perceived on site. The trajectory provides successful coverage of the underwater boulder, allowing a full 3D perception of the target structure as demonstrated in the resulting 3D maps below.

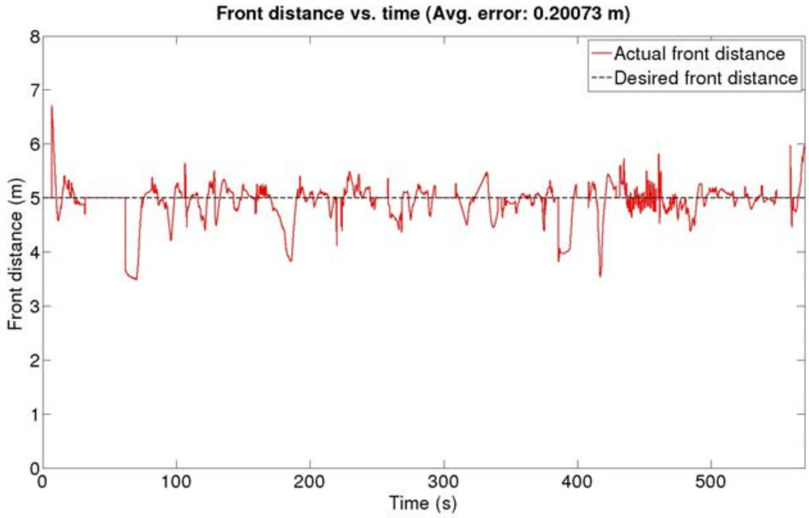
Indeed, Figure 102 shows the reconstructed surface from the raw range data in Figure 99, with normals computed with a neighborhood of $k = 100$ points. The overall surface faithfully represents the shape of the surveyed underwater boulder, increasing by far the resolution from the off-line model in Figure 95. However, we note that undersampled parts are overly extrapolated (in particular at the top of the mount) and outliers and registration errors create some small artifacts.



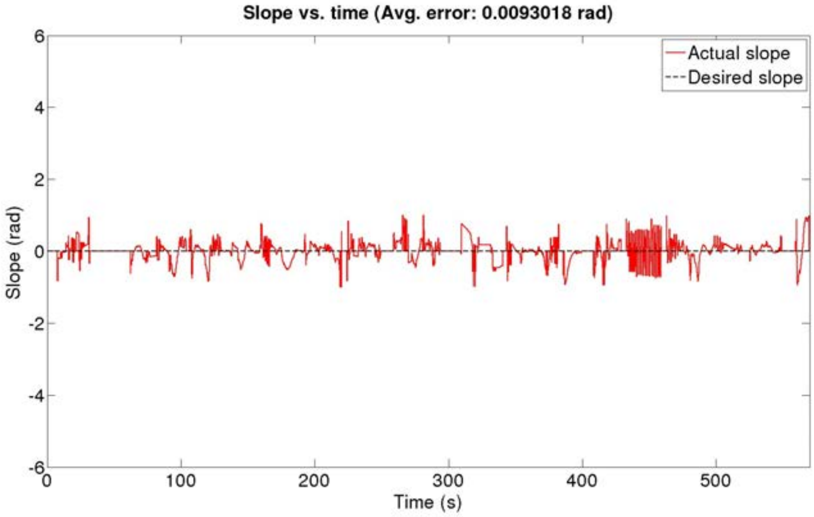
(a)

(b)

Figure 97: Path traced by GIRONA 500 when performing coverage of l'Amarrador site using horizontal profile following in simulation: path after following the deepest profile (a) and path after finishing following the second profile (b).

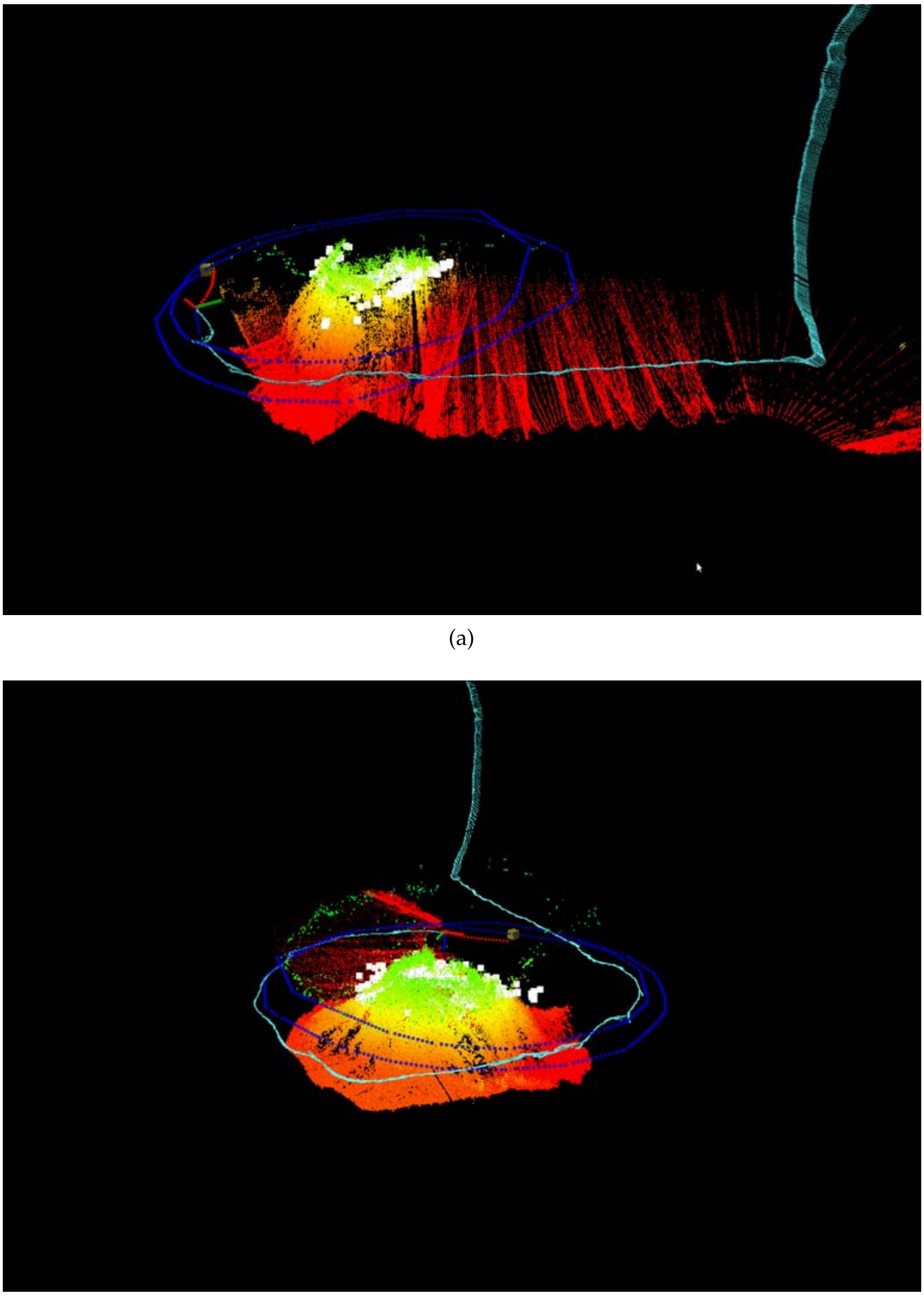


(a)



(b)

Figure 98: Simulated coverage using horizontal profile following at l'Amarrador site: frontal distance (a) and slope (b) given by our profile estimation process vs. time.



(a)

(b)

Figure 99: Realtime replanning on “l’Amarrador” underwater boulder at the beginning (a) and at the end (b) of the deepest coverage edge: nominal coverage path (blue-dotted line); optimized trajectory which the robot is executing at that particular instant (red-dotted line); overall trajectory (cyan arrows); occupied cells in the current on-line map (white cubes); and last processed waypoint of the off-line plan (yellow cube). The current pose of the vehicle is represented by the red-green-blue 3D axis. The depth-colored range data acquired by the multibeam sonar is also displayed.

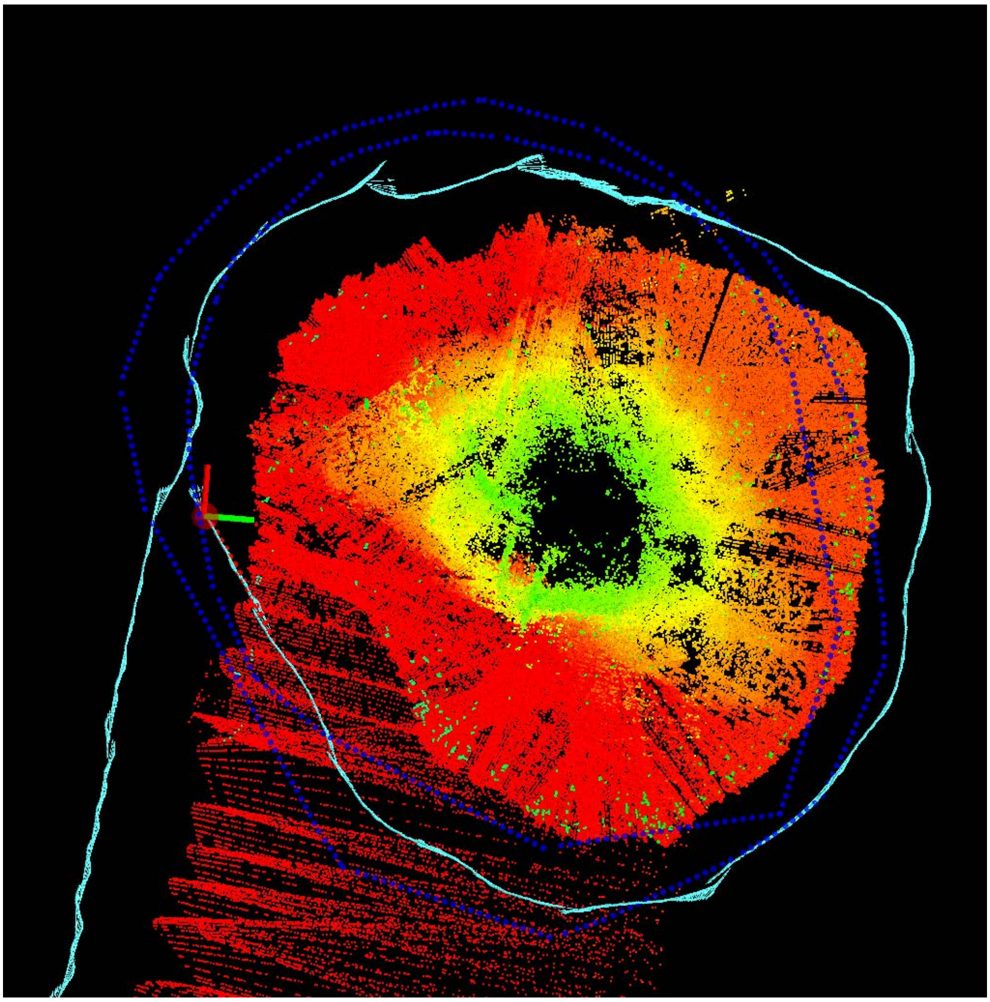
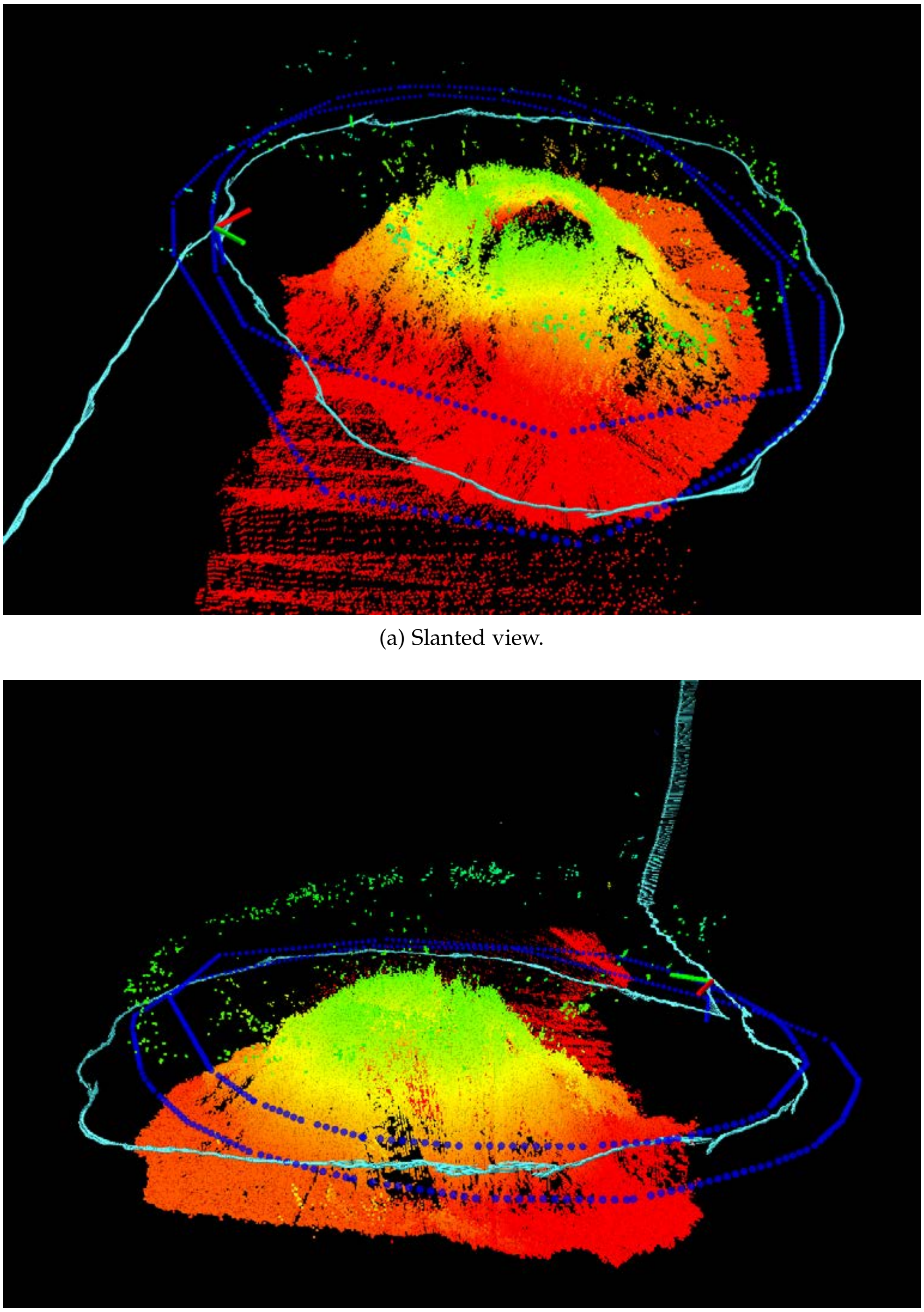


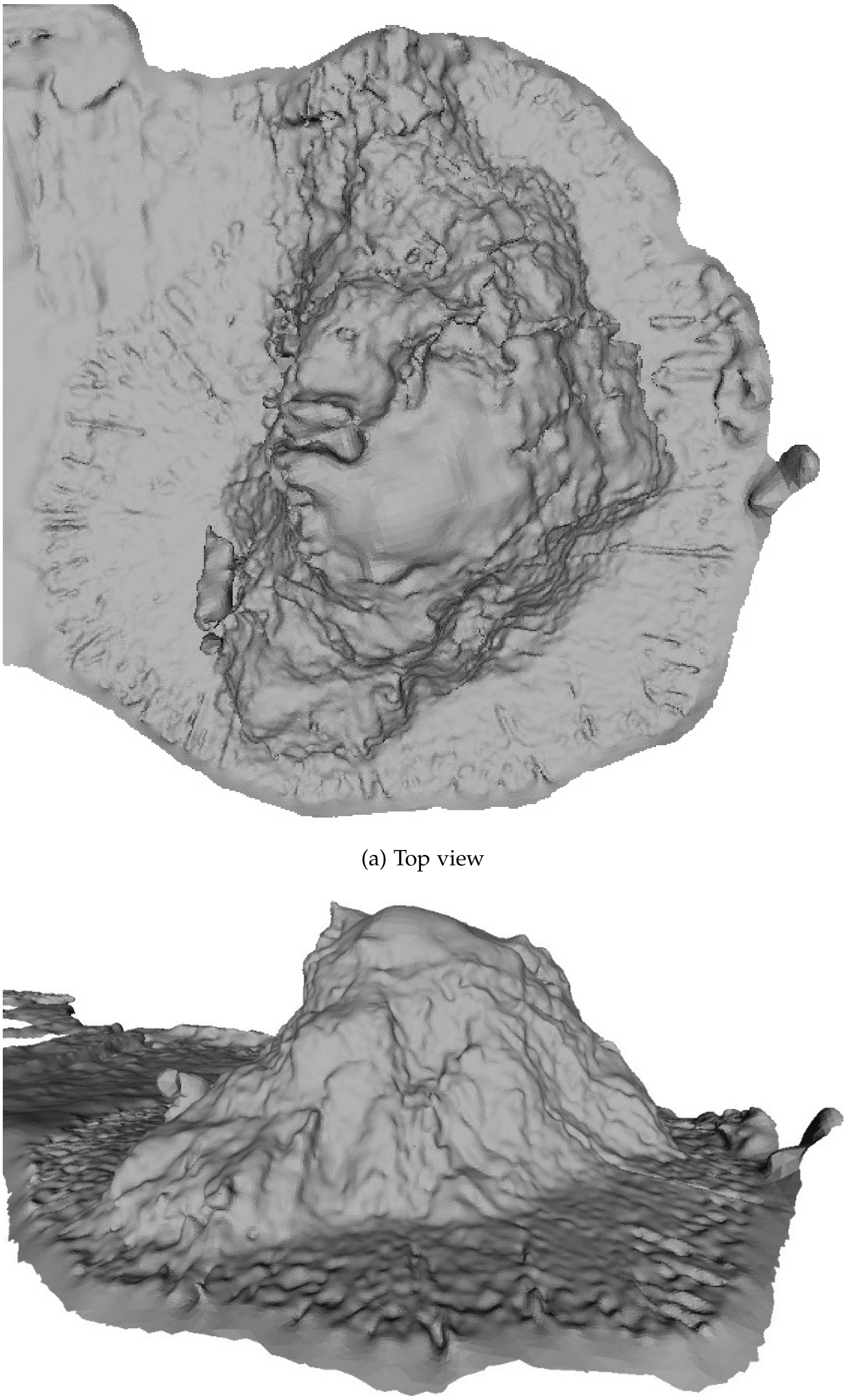
Figure 100: Coverage trajectory on "l'Amarrador" underwater boulder (top view): nominal coverage path (blue-dotted line) and overall trajectory (cyan arrows). The current pose of the vehicle is represented by the red-green-blue 3D axis. The depth-colored range data acquired by the multibeam sonar is also displayed.



(a) Slanted view.

(b) Opposite angle.

Figure 101: Coverage trajectory on "l'Amarrador" underwater boulder (slanted views): nominal coverage path (blue-dotted line) and overall trajectory (cyan arrows). The current pose of the vehicle is represented by the red-green-blue 3D axis. The depth-colored range data acquired by the multibeam sonar is also displayed.



(a) Top view

(b) Slanted view

Figure 102: Surface of “l’Amarrador” underwater boulder, recovered from the raw range data. Image credit: Ricard Campos.

6 CONCLUSION

Real stupidity beats artificial intelligence every time.
— Terry Pratchett, "Hogfather" (1996)

This chapter concludes this thesis by presenting a summary of work completed in Section 6.1, reviewing the contributions of the thesis in Section 6.2 and the publications that have resulted from it in Section 6.3. Finally, compelling areas for future work are outlined in Section 6.4.

6.1 SUMMARY OF WORK COMPLETED

This thesis has addressed the coverage path planning problem in the underwater domain, working in 2D and 3D environments with different requirements. In Chapter 2 we reviewed the state of the art of coverage path planning, providing a classification of the most popular and successful coverage path planning algorithms and applications, providing a novel classification and pointing out directions for further research in the problem. The review puts an emphasis on the underwater domain and represents the first comprehensive review of the coverage path planning literature presented in more than ten years, reflecting the recent breakthroughs in the field.

In Chapter 3 we proposed a novel algorithm for 2D coverage of a target region on the ocean floor using an AUV that provides AUV survey planning with a principled way to account for obstacles. The main advantage of this method is that it minimizes the redundant coverage arising when the vehicle surveys the area at constant depth. By segmenting the target surface in regions of similar depth and addressing them as individual coverage path planning problems, the method is able to adjust the inter-lap spacing and orientation of the lawnmower-type paths used to cover the area, leading to a more efficient, shorter path. The proposed method has been validated in simulation using a real-world bathymetric dataset, showing a significant increase in path efficiency in comparison to a standard lawnmower-type path.

In Chapter 4 we have introduced a new survey path planning method for area coverage aimed at minimizing the uncertainty in the vehicle position estimates which takes into account the application constraints of bathymetric mapping. We compared the proposed method performance to standard lawnmower-type paths in terms of position uncertainty along the path in simulation with different real-world datasets and in sea trials with the GIRONA 500 AUV. Additionally, we compared the mapping performance associated to our method to a standard survey path using a bathymetric mapping algorithm. Results showed the benefit of incorporating uncertainty in the survey path planning phase both in terms of position uncertainty and mapping quality enhancement. Although our algorithm focuses on bathymetric mapping, we believe that many underwater robotic applications can benefit from it, especially those relying on the AUV trajectory estimate such as optical and sonar mosaicing. Likewise, data products obtained with different sensors, such as Forward-Looking Sonar (FLS) and Side-Scan Sonar (SSS), can benefit from the improved trajectory estimates enabled by our planning method.

In Chapter 5 we presented an off-line 3D coverage path planning method including two algorithms for coverage of bathymetric surfaces. The method takes into account the slope of the areas on the bathymetric surface and generates paths suiting the characteristics of effectively planar regions and high-slope regions. In high-slope regions, the planned coverage path follows cross-sections of the target structure at a given offset distance, being able to provide a clear and continuous data product. We demonstrated the feasibility of the method by planning a coverage path on a real world bathymetric dataset from the Caldera 2012 trials. Given that a perfect execution of the coverage path is not a realistic assumption in a real-world scenario due to uncertainty, we proposed two different methods to adapt a nominal coverage path in realtime using range sensor measurements perceived *in situ* by the AUV. On the one hand, we have presented a seabed profile estimation and following technique using a hovering-capable AUV. The proposed profile estimation technique uses a local map and simple linear regression to obtain an estimate of the profile in certain regions of interest around the robot. The profile estimation information is then used by profile following modules which reactively guide the vehicle to follow the estimated profile. Our method is suited for both vertical and horizontal profile following, enabling inspection of seabed structures in close proximity. The method has been tested both in simulation and in-water trials using the GIRONA 500 AUV, which have demonstrated the effectiveness of this approach and its applicability to 3D coverage tasks. The method performed well both in structured (swimming pool) and unstructured (diving site) scenarios. On the other hand, we proposed a 3D coverage path replanning method for inspection of complex underwater structures. The method takes a nominal 3D coverage path as input and uses stochastic trajectory optimization to reshape it in realtime during the mission. Provided a bound for the vehicle's position error, the replanning algorithm is able to adapt the nominal path according to range sensor measurements onboard the vehicle. Our method has proven successful in two coverage experiments at sea with the GIRONA 500 AUV, involving coverage of a part of a breakwater structure and of an underwater boulder rising from 40 m up to 28 m depth. Moreover, we have presented 3D models of the inspected sites that show the benefits of our coverage path planning method for 3D mapping of complex structures, not amenable for traditional bathymetric mapping.

6.2 review of contributions

This thesis has contributed to coverage path planning and to the field of underwater robotics in five main respects.

literature review This thesis has presented a thorough, comprehensive review of the coverage path planning literature. The review classifies the methods according to their target workspace and approach and reports on application experiments. Remarkably, the review presented in this thesis represents the first survey on coverage path planning in more than a decade.

efficient 2d coverage with obstacles This thesis has proposed 2D survey path planning methods for maritime vehicles, providing a principled way to account for obstacles and ensure complete, efficient coverage by reducing coverage overlapping.

uncertainty-driven auv survey planning A 2D off-line survey planning method has been proposed that minimizes the uncertainty in the vehicle position estimates induced by the path, leading to more accurate data products. The proposed method goes beyond traditional survey planning, where uncertainty is ignored in the planning phase.

coverage of 3d structures This thesis has proposed a set of methods for planning and executing coverage paths in realtime for 3D structures on the ocean floor. The resulting 3D coverage plans are adapted on site using the AUV onboard sensors and provide close proximity inspections with appropriate viewing angles for imaging applications.

experimental evaluation This thesis has provided experimental evaluation of the proposed methods with a physical AUV, demonstrating their field applicability in real-world environments.

6.3 review of publications

Next, we review the list of publications that have resulted from this thesis and connect them to the content of this document. The list of publications below is first organized by theme and, therein, chronologically from newest to oldest.

Literature Review

- [ras'13] Enric Galceran and M. Carreras. *A Survey on Coverage Path Planning for Robotics.* Robotics and Autonomous Systems. Volume 61, Issue 12, December 2013, Pages 1258-1276.

The literature review presented in this thesis in Chapter 2 has been published in the *Robotics and Autonomous Systems* top robotics journal, especially reflecting the advances in coverage path planning occurred in the last decade and serving as a reference for researchers in the field.

2D Coverage Path Planning

- [OCEANS'12] Enric Galceran and Marc Carreras *Coverage Path Planning for Marine Habitat Mapping.* OCEANS'12. Hampton Roads (VA), USA. October 2012.
- [IROS'12] Enric Galceran and Marc Carreras. *Efficient Seabed Coverage Path Planning for ASVs and AUVs.* Intelligent Robots and Systems (IROS). Vilamoura, Portugal. October 2012.
- [OCEANS'11] Emili Hernández, Marc Carreras, Enric Galceran and Pere Ridao. *Path Planning with Homotopy Class Constraints on Bathymetric Maps.* OCEANS'11. Santander, Spain. June 2011.

The [IROS'12] paper presented the efficient 2D coverage path planning method introduced in Chapter 3, while the [OCEANS'12] paper suggested an extension of this method for covering regions of interest in marine habitats for monitoring purposes. The [OCEANS'11] paper included a collaboration of the author in generating 2D workspaces from bathymetric maps.

Uncertainty-driven Coverage

- [IROS'13] Enric Galceran, Sharad Nagappa, Marc Carreras, Pere Ridao and Albert Palomer. *Uncertainty-driven Survey Path Planning for Bathymetric Mapping.* Intelligent Robots and Systems (IROS). Tokyo, Japan. November 2013.

This [IROS'13] paper presented the uncertainty-driven AUV survey planning algorithm introduced in Chapter 4 to the robotics community.

3D Coverage

- [ICRA'14] Enric Galceran, Ricard Campos, Narcís Palomeras, Marc Carreras and Pere Ridao. *Coverage Path Planning with Realtime Replanning for Inspection of 3D Underwater Structures.* International Conference on Robotics and Automation (ICRA). Hong Kong, China. June 2014 (to appear).
- [JBR'13] Enric Galceran, Narcís Palomeras and M. Carreras. *Profile Following for Inspection of Underwater Structures.* Paladyn Journal of Behavioral Robotics. Volume 4, Issue 4, December 2013, Pages 209-220.
- [MARTECH'13] Enric Galceran, Ricard Campos and Marc Carreras. *Automating Seafloor Inspection using Autonomous Underwater Vehicles.* Fifth

International Workshop in Marine Technology (MARTECH). Girona, Catalonia (Spain). October 2013.

- [OCEANS'13] Enric Galceran, Marc Carreras, Narcís Palomeras and Pere Ridao. *Complex Structure Profile Estimation and Following with the GIRONA500 AUV.* OCEANS'13. Bergen, Norway. June 2013.

- [ICRA'13A] Enric Galceran and Marc Carreras. *Planning Coverage Paths on Bathymetric Maps for In-Detail Inspection of the Ocean Floor.* International Conference on Robotics and Automation (ICRA). Karlsruhe, Germany. May 2013.

In regard to 3D coverage, the [ICRA'13A] paper presented the 3D off-line coverage path planning method introduced in Section 5.2. The work in profile following for coverage described in Section 5.3 was first presented in the [OCEANS'13] paper, and later extended and published as a journal paper in [JBR'13]. The coverage path planning with realtime replanning approach introduced in Section 5.4 has been published in the [MARTECH'13] and [ICRA'14] papers.

Other Publications

- [ICRA'13B] Ross Hatton, Ross A Knepper, Howie Choset, David Rollinson, Chaohui Gong and Enric Galceran. *Snakes on a Plan: Toward Combining Planning and Control.* International Conference on Robotics and Automation (ICRA). Karlsruhe, Germany. May 2013.

- [AUTOMAR'12] Enric Galceran. *Coverage Path Planning for In-Detail Seafloor Inspection.* V Jornadas Automar. Girona, Catalonia (Spain). December 2012.

- [NGCUV'12] Enric Galceran, Vladimir Djapic, Marc Carreras and David P. Williams. *A Real-time Underwater Object Detection Algorithm for Multibeam Forward Looking Sonar.* IFAC's workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV). Porto, Portugal. April 2012.

As a result of his research stay at Carnegie Mellon University (CMU) in 2012, the author collaborated in the [ICRA'13B] paper about integrated path planning and control for snake robots. The [AUTOMAR'12] paper presented a recap of the work completed as part of this thesis as of late 2012, including mainly efficient 2D coverage and off-line 3D coverage ([IROS'12] and [ICRA'13A] papers). The [NGCUV'12] paper resulted from a stay at NATO Undersea Research Centre (NURC) in 2011, where the author worked in using sonar technology for detection of objects on the sea floor. The contents of this later paper and a report of the work completed during the stay at NURC are available in Chapter B in the Appendix. The author was hosted by Howie Choset at CMU and by Vladimir Djapic at NURC.

6.4 COMPELLING AREAS FOR FUTURE WORK

The work completed in this thesis opens the door to an array of compelling areas for future work.

FUTURE WORK IN 2D COVERAGE WITH OBSTACLES The 2D coverage path planning method we presented in Chapter 3 showed success in simulation, providing a path that successfully avoids obstacles and minimizes redundant coverage. The feasibility of the coverage path planning also presented therein was demonstrated in simulation as well. Evaluating the performance of these methods in real-world experiments is definitely an interesting subject for further research.

FUTURE WORK IN UNCERTAINTY-DRIVEN COVERAGE The method for generating survey paths for coverage of marine environments accounting for sensing uncertainty we introduced in Chapter 4 can easily be integrated into common survey planning tools for marine robotics, such as MB-System (MBARI, 2013), which is freely available to the scientific community. The integration of this method can endow the users with a tool that capitalizes on the benefits of incorporating uncertainty in maritime survey planning. Therefore, short-term efforts will explore this possibility. Further work will consider incorporating *a priori* map errors into the robot's belief estimation, thereby accounting for the uncertainty in the environment's map. Exploring the theoretical uncertainty and optimality bounds of the proposed method is also a topic for future work. Finally, we would like to study the possibility of using multi-objective optimization techniques to balance the trade off between uncertainty and path length in our planning method.

FUTURE WORK IN 3D COVERAGE In regard to the methods for realtime replanning of nominal 3D coverage paths we presented in Chapter 5, on one hand, we will focus our immediate efforts on testing the profile estimation and following techniques proposed therein at sea, inspecting natural structures of interest. At longer term, we are interested in exploring control techniques to increase the profile following accuracy of our method. On the other hand, we are currently testing further the realtime coverage replanning method in several challenging sites of interest and using different sensor configurations. In particular, we are interested in incorporating the multibeam sonar data in the realtime map construction process. This would yield a more dense map of the 3D structure and help filtering outliers that prevent a more accurate replanning. In the future, we plan to incorporate map uncertainty in the cost function to obtain safer trajectories with smaller probability of collision. A theoretical analysis of the presented replanning method is also a subject for future research.

FUTURE WORK IN AUV MOTION PLANNING IN GENERAL The path planning algorithms introduced in this thesis generate geometric paths in a workspace that do not take into account the controller used during the execution of the path nor potential kinodynamic constraints brought about by the vehicle at use. That is, these issues are left for the lower-level path following controllers. As a result, the generated paths might not be readily feasible for a particular AUV. On the other hand, environmental disturbances such as currents might severely affect the vehicle's control. Therefore, incorporating kinodynamic constraints of the vehicle and information about the control algorithms that will be used to execute the path in the planning phase, and using robust control algorithms for execution can lead to a more reliable transition from planning to execution. This issue remains as a subject for further research.

APPENDIX

A

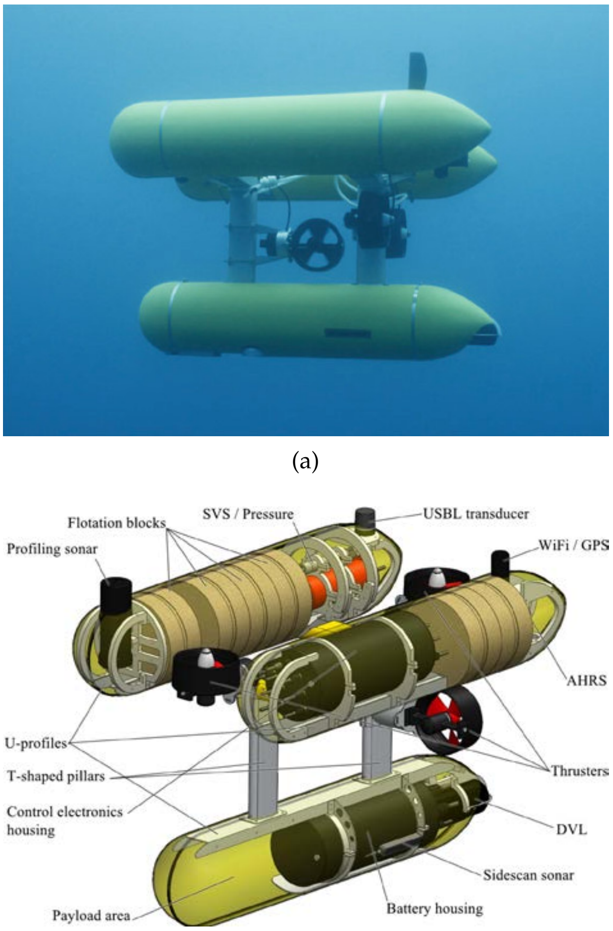
EXPERIMENTAL PLATFORM: THE GIRONA 500 AUV

The GIRONA 500 AUV has been the main platform for the development and experimental evaluation of the algorithms proposed in this thesis. Next, we describe the design and capabilities of GIRONA 500 (Section A.1), its control architecture (Section A.2) and the UWSim underwater simulation package (Section A.3), which provides high-fidelity simulation of the GIRONA 500 AUV and its environment.

A.1 THE GIRONA 500 AUV

GIRONA 500 (Ribas et al., 2012), shown in Figure 103, is a reconfigurable AUV developed at CIRS which is designed to operate at depths up to 500 m. The vehicle is composed of an aluminum chassis supporting three torpedo-shaped hulls (0.3 m in diameter and 1.5 m in length) and the thrusters. This design offers good hydrodynamic performance and room for housing equipment while keeping the vehicle compact, allowing deployment from small vessels. The overall dimensions of the AUV are 1 m height, 1 m width, 1.5 m length and a weighs under 200 kg, depending on configuration. The two upper hulls, which contain the flotation foam and the electronics, are positively buoyant, whereas the lower one contains the heavier elements such as batteries and payload. This particular arrangement of the components separates the center of gravity from the centre of buoyancy by about 11 cm, which is significantly more separation than that found in a typical torpedo shape design. This provides the vehicle with passive stability in pitch and roll, making it suitable for tasks that will benefit from a steady platform such as underwater manipulation.

The most remarkable characteristic of the GIRONA 500 is its capability to be easily reconfigured for different tasks. In its standard configuration, the vehicle is equipped with a typical navigation sensor suite (DVL from Teledyne, AHRS from Tritech, pressure gauge and sound velocity sensor from Valeport and USBL from EvoLogics) and basic survey equipment (Delta T multibeam sonar and side-scan sonar from Imagenex, video camera from Tritech and a Bumblebee 2 stereo camera from Point Grey). In addition to these sensors, almost half the volume of the lower hull is reserved for payload equipment that can be added to the vehicle to meet the requirements of a particular mission. For instance, in can be equipped with a robotic arm for intervention tasks, as shown in Figure 104. The same philosophy has been



(a)

(b)

Figure 103: The GIRONA 500 AUV. (a) The vehicle in its first sea trial. (b) CAD model showing the structure and components of the vehicle. Image credit: Ribas et al. (2012).

applied to the propulsion system, which is also reconfigurable. The basic layout is composed of four thrusters, two vertically oriented thrusters actuating in the heave and pitch DOFs and two horizontally oriented thrusters for the heading and surge DOFs. However, it is possible to reconfigure the vehicle to operate with only three thrusters (one vertical and two horizontal) and with up to eight thrusters to control all six DOFs.

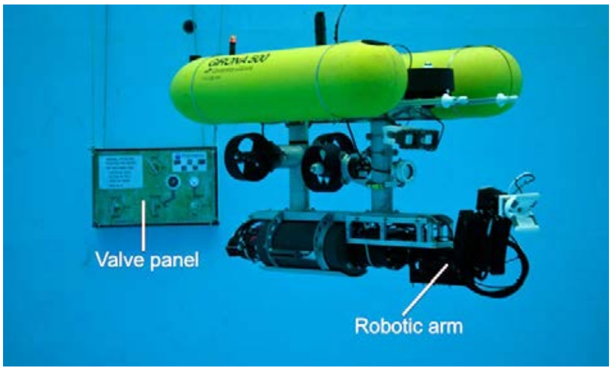


Figure 104: The GIRONA 500 AUV performing an intervention task on a valve panel equipped with a robotic arm.

A.2 THE COLA2 CONTROL ARCHITECTURE

The Component-Oriented Layer-based Architecture for Autonomy (COLA2) (Palomeras et al., 2012), GIRONA 500's control architecture, drives the AUV during survey and intervention missions. A block diagram describing the architecture is shown in Figure 105. The architecture consists of a Perceive-Plan-Act module connected to the device drivers for sensors an actuators. The Perceive-Plan-Act module constructs a representation of the world using sensor measurements including images, point clouds and ranges, estimates the vehicle trajectory, plans actions according to them and executes said actions. It maintains an estimate of the robot's trajectory via the Navigation component, and this estimate is used by the Object Recognition and Mapping components. The Object Recognition component uses *a priori* knowledge to seek for matchings between sensor measurements and object models. The Mapping component maintains a multi-modal 3D representation of the world and uses it to provide feedback to the Navigation component via SLAM. The Mission Planning component receives and monitors the execution of a mission plan consisting of a sequence of tasks from the user via the Human-Machine Interface and Communications components. The Task Execution component executes each task during a mission making use of the Path Planning and Coverage Path Planning components (this later one implementing the coverage path planning methods proposed in this thesis), which generate collision-free Cartesian paths. The Task Execution component also receives input from the Learning component when performing manipulation tasks. The tasks are executed by sending position setpoints to the Guidance and Low-level Control components, which generate velocity setpoints. In particular, the Guidance component includes the implementation of the profile following strategy presented in Section 5.3. Finally, the Velocity Controller component transforms the desired velocity setpoints into commands that are sent to the actuators. The architecture is implemented in C++ and Python using the ROS framework.

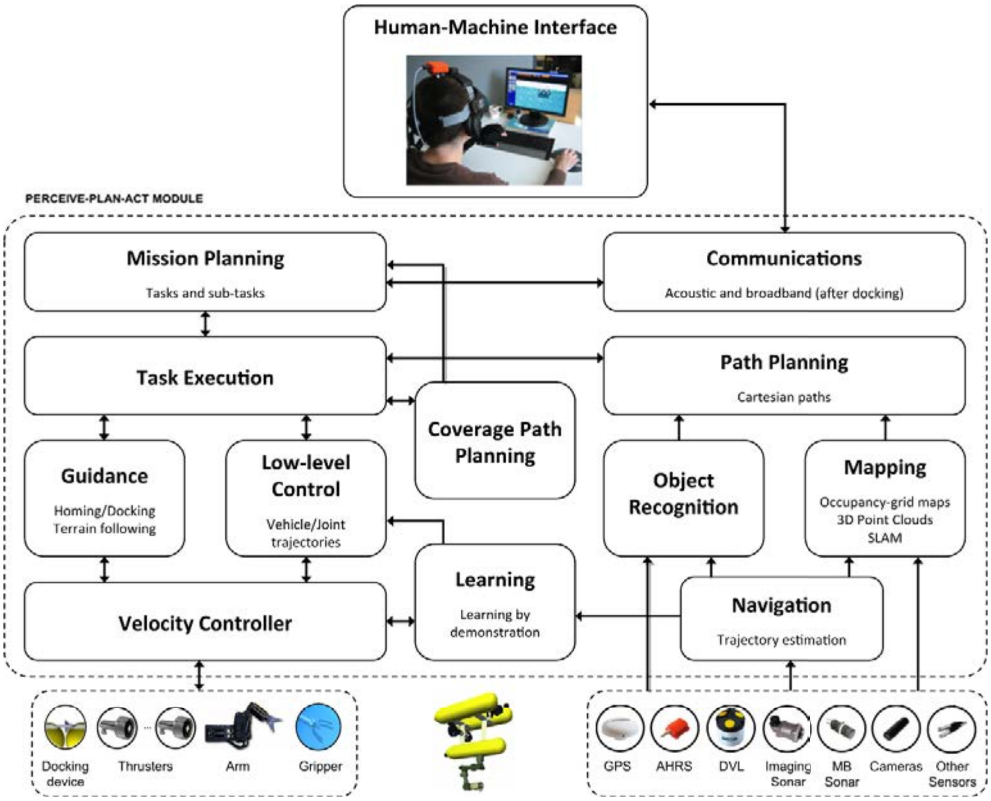


Figure 105: COLA2: GIRONA 500's control architecture.

A.3 SIMULATION WITH UWSIM

UWSim is a tool for testing and integrating perception and control algorithms for marine vehicles before running them in the real world. UWSim visualizes an underwater virtual scenario that can be configured using standard 3D modeling software. Support for realistic effects such as the ocean surface, water turbidity, water color, silt particles and crepuscular rays and physics simulation (e.g. collision, force feedback) is provided. Controllable underwater vehicles, surface vessels and robotic manipulators, as well as simulated sensors, can be added to the scene and accessed externally through network interfaces, including support for missions with multiple vehicles. This allows to easy integration with existing control architectures. An example environment with a model of the GIRONA 500 AUV controlled by the COLA2 architecture is shown in Figure 106.

UWSim provides support and includes off-the-shelf models for 4- and 7-DOF arms, positioning sensors, cameras, range-sensing sonars (such as multibeams and mechanically-scanning profilers) and DVL devices, which can be

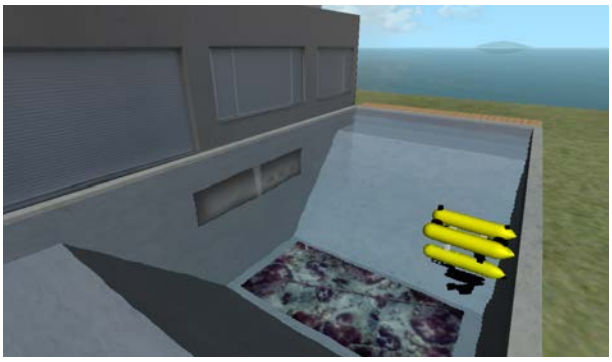


Figure 106: The GIRONA 500 AUV executing a mission in UWSim with a model of the CIRS facilities.

integrated with the vehicles and interfaced by external applications. In fact, all the different robots, sensors and actuators can be interfaced with external software through the network. ROS support is provided, and through the network interfaces, it is possible to access/update any vehicle position or velocity, move arm joints, access the images generated by virtual cameras and ultimately, interface all the components involved in the simulation.

B

UNDERWATER OBSTACLE DETECTION

Detection of obstacles in the environment is an integral component of any robotic system that is to perform a coverage path planning task. In this thesis we have mainly used range sensing sonars (multibeam and profilers) to perceive the environment. However, having dramatically evolved in the last two decades, there exist nowadays a wide range of sensors enabling autonomous maritime vehicles to perceive underwater environments. This appendix presents a concise review of sonar technology suitable for detecting obstacles in underwater environments. Furthermore, as an illustrative application of sonar technology to underwater obstacle detection, we present a realtime underwater man-made object detection algorithm developed in the framework of this thesis. The bulk of the work here presented was carried out during 2011 at NURC, now known as Centre for Maritime Research and Experimentation (CMRE), in La Spezia, Italy, where acoustic real-world datasets from state-of-the-art sonars could be collected and analyzed and testing could be carried out on physical vehicles at sea. The sonar technology review is presented in Section B.1 and the realtime object detection algorithm is presented in Section B.2.

B.1 REVIEW OF ACOUSTIC SENSING FOR UNDERWATER OBSTACLE DETECTION

Most coverage path planning methods reviewed in Chapter 2 rely on range detection sensors for perceiving the environment and drive the robot accordingly. Usually, these sensors are assumed to be perfect by coverage algorithms, that is, providing no noisy measurements and arbitrary precision. This is a fair assumption in many ground and aerial robotics applications, where laser-based range sensors and cameras provide a precise perception of the robot's environment. However, the applicability of such technology is very limited in the underwater domain, where the lack of visibility and the particularities of the environment restrict the accuracy of the sensorial information. By contrast, this very same lack of visibility due to light attenuation in underwater environments makes acoustic perception the most reliable and widely used technology for sensing such environments. Active sonars work by transmitting an acoustic signal. Objects within the path of the acoustic beam then reflect some of the energy back toward the sonar transducer (this effect is known as backscatter). The fast propagation of sound through water makes possible for an acoustic wave to travel hundreds of meters without the signal losing significant energy, allowing to perceive the environment at long ranges even in turbid and poorly illuminated water conditions.

In regard to the backscattered signals, two different kinds of information can be obtained from them. On one hand, range-sensing record the two-way travel time of the reflected acoustic pulse and convert it to distance assuming a particular sound velocity in water. On the other hand, imaging sonars can also measure the intensity of the returning pulses which and obtain an "acoustic image" describing the composition of the scanned terrain. It is worth to underline that, despite the increased range with respect to "light-based" sensors in underwater environments, sound transmission underwater is also attenuated, especially when using high frequencies. Therefore, a tradeoff exists between a longer range provided by a low frequency sonar and the higher resolution provided by a high frequency sonar.

Next, we concisely review the principal types of sonar sensors suitable for underwater obstacle detection aiming to introduce their main characteristics, configuration and applications. We discuss range-sensing sonars in Section B.1.1 and imaging sonars in Section B.1.2. For a more thorough review on acoustic sensing the reader can refer to Lurton (2002).

B.1.1 *Range-sensing Sonars*

B.1.1.1 *Single-beam Sounders*

Single-beam sounders, used since the 1920's, are the most basic and common underwater acoustic systems. They are usually mounted on a vessel's hull facing downwards in order to measure the ocean depth. However, they can be used to detect approaching obstacles from the front by mounting it in a horizontal plane parallel to the vessel's motion. For instance, an approaching wall is an obstacle feasible to be detected by a single-beam sounder.

Single-beam sounders use a single transducer that emits a short sound pulse (typically between 10^{-4} s and 10^{-3} s) a beam of narrow aperture (typically between 5° and 15°), as illustrated in Figure 107. This transducer receives the echo signal and a corresponding range is computed. The operating frequencies of single-beam sounders depend on the working ranges. Usually they range from 12 kHz for deep waters (up to 11000 meters) to 400 kHz or even more for shallower waters (from 1 to 50 meters). There are also dual-frequency transducers available in the market. The range resolution depends on the emitted pulse duration but typically ranges between 0.075 and 0.75 meters.

With only a single beam, these devices are not a precise obstacle detection instrument as they can easily miss obstacles approaching from nonorthogonal directions to the beam direction. However, they can be integrated in small vehicles and their data can be processed quickly. As a result, single-beam sounders are popular in a great number of underwater robotics applications.

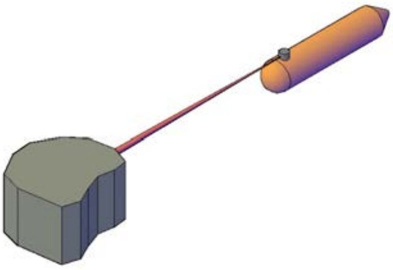


Figure 107: Obstacle perception with a single-beam sounder. Image credit: Miquel Rusca.

B.1.1.2 *Mechanically-scanning Profilers*

Mechanically-scanning profilers are composed of a mechanically actuated transducer which can be sequentially oriented at different angles and produces a series of range measurements. They can be thought as a rotary single-beam sounder. Usually, the size of the scan sector can be selected from a few degrees to a complete 360° scan around the transducer, which is particularly interesting for obstacle detection. When mounted in a down-looking position, they can also be utilized to collect bathymetric data. With a mechanically scanned profiler a point cloud corresponding to the incidence of the beams along the profile of an obstacle can be obtained. An important issue that has to be taken into account when using these devices is that some time is necessary for a scan to complete. Hence, if the vehicle is moving at high speeds, the range measurements get distorted as a consequence of the vehicle's motion.

B.1.1.3 *Multibeam Sonars*

Motivated by the limitations of single beam sounders, multibeam sonars appeared in the 1970's and since then have greatly evolved and are nowadays a widespread system for seafloor mapping tasks. However, their application to obstacle detection is somewhat limited.

As mechanically scanned profilers, multibeam sonars produce a series of range measurements along a scan sector. By contrast, however, they provide multiple readings at a single time step rather than using a rotary beam. That is, multibeam sonars transmit a swath of individual acoustic beams (usually 200-400 beams, between 1° and 3° each), obtaining an array of range information around the transducer, as illustrated in Figure 108). The swath's aperture angle typically ranges from 90° to 180°. Usually a multibeam sonar is composed of two transducer arrays, one for transmission and one for reception. In this way, the information provided by a single beam sounder is obtained at multiple angles at once. A multibeam sounder provides a point cloud corresponding to the incidence of each beam along the profile of the obstacle. As an advantage over mechanically-scanning profilers, multibeam sonars are much less affected by motion-induced distortion.

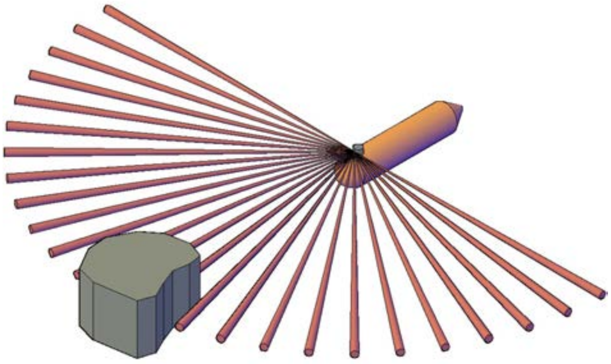


Figure 108: Obstacle perception with a multibeam sonar. Image credit: Miquel Rusca.

B.1.2 *Imaging Sonars*

B.1.2.1 *Forward-looking Sonars*

FLSs are similar to single beam and multibeam technologies but they usually provide imagery data rather than range information (see Figures 109 and 110). FLSs have been used for many years in ROVs, obstacle avoidance and naval mine detection. The major advantage of this type of sonar is its capability of detecting objects or seabed features, such as protruding rocks on the ocean floor, at large distances so they can be observed in subsequent scans and tracked. Unfortunately, an important limitation for FLS range is depth; due to surface and bottom echo interference, FLSs can generally see ahead only about six times the depth of the water column. Two main types of FLS are available: mechanically-scanning imaging sonars and multibeam FLS.

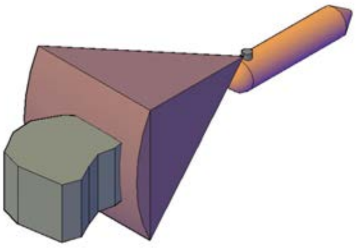
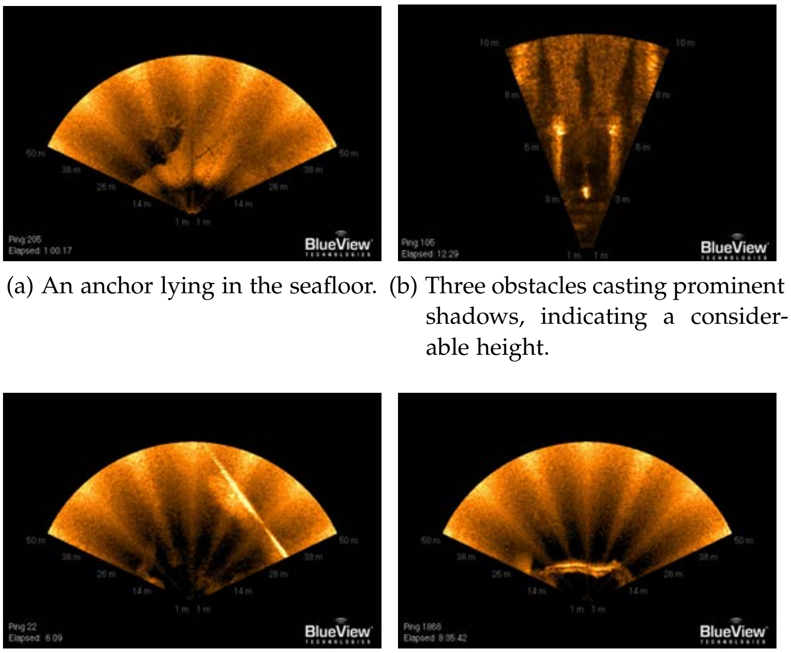


Figure 109: Obstacle perception with a multibeam FLS. Image credit: Miquel Rusca.

Mechanically-scanning imaging sonars consist of a single transducer which is mechanically rotated along an axis. The returns are then used to create an image. Most systems provide the user with the option of choosing the size of the sector to scan and with some degree of control on the resolution. However, as in mechanically-scanning profilers, mechanically-scanning sonars suffer from motion-induced distortion.

Multibeam FLS use a fixed array of transducers, processed electronically, which allows much update rates (e.g., the Seabat 6012 multibeam FLS provides up to 30 times a second). These sonars are more costly than mechanical systems; nevertheless their popularity in the underwater community has been growing in the recent years. Automatic methods for obstacle avoidance, motion estimation, image recognition and sonar mosaicing using multibeam FLS imagery have already appeared in the literature (Hurtos et al., 2014). Fast update rates make distortion due to motion practically negligible in these devices.

Some high-frequency multibeam FLSs have recently appeared on the market, termed "acoustic cameras" in regard to their capability to provide acoustic video imagery (high refresh rates and high resolution at short ranges). The cost of this sort of sensors is still quite high (almost three times the cost of a standard multibeam FLS), but as their size and price decreased they become an increasingly interesting option for AUV applications.



(a) An anchor lying in the seafloor.

(b) Three obstacles casting prominent shadows, indicating a considerable height.

(c) A pier.

(d) A pipe-like obstacle.

Figure 110: Several sonar images captured with BlueView FLS devices.

B.1.2.2 *Side Scan Sonars*

SSS is a category of imaging sonar systems used to efficiently create an image of large areas of the sea floor. This contrasts with FLS, which provides coverage at a smaller scale. SSS is a helpful tool to detect debris items and other obstructions on the seafloor that may be hazardous to shipping or to seafloor installations. SSSs use a transducer that emits conical or fan-shaped pulses down toward the seafloor across a wide angle perpendicular to the path of the sensor through the water (track), as shown in Figure 111. The intensity of the acoustic reflections from the seafloor of these fan-shaped beams are recorded in a series of cross-track slices. When stitched together along the "along track" direction, these slices form an image of the sea bottom within the swath (coverage width) of the beams. The sound frequencies used in SSS usually range from 100 to 500 kHz.

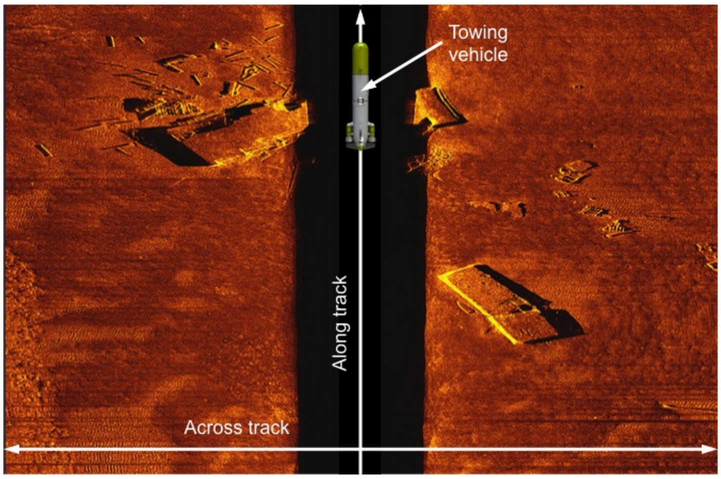


Figure 111: Sea floor perception using SSS imagery. Sonar imagery credit: EdgeTech.

B.1.2.3 *Synthetic Aperture Sonars*

Synthetic Aperture Sonars (SASs) build upon the same working principle as SSSs but use sophisticated post-processing techniques to obtain higher resolution imagery. SASs combine a number of acoustic pings to form an image with much higher resolution than conventional SSSs, typically 10 times higher. To achieve increased resolution, SAS illuminates the same spot on the sea floor with several pings. By coherent reorganization of the data from all the pings, a synthetic image is produced with improved along-track resolution. In contrast to conventional SSS, SAS processing provides range-independent along-track resolution.

B.2 REALTIME UNDERWATER OBJECT DETECTION USING FLS

The capability of perceiving underwater environments even in very poor visibility conditions has proven particularly useful for the detection of underwater man-made objects. With recent advances in maritime technology, the sonar data used to address this task can be collected by an ASV or an AUV with no human intervention. Detection and intervention on man-made objects are costly tasks and imply inherent danger and time constraints. Thus, to equip the autonomous vehicle with intelligence so that it can immediately react to the data it collects is a priority. Before this goal can be realized, however, an algorithm is needed that can perform object detection in realtime onboard.

Two sorts of sonar technology in particular have proved useful for automatic man-made object detection. On one hand, long range, high resolution imagery provided by SSS and especially by SAS allows for performing detection in vast survey areas (i.e. hundreds of meters long survey tracks). On the other hand, multibeam FLS allows for a closer, more detailed inspection of possible man-made object locations. A typical approach to the problem consists in first detecting possible object locations in SSS or SAS imagery, and then performing reacquisition of these locations by means of FLS to assess that there are in fact objects of interest in such locations, and perhaps carrying out some intervention task on them.

Hayes and Gough (1992) showed that the high-resolution imagery provided by SAS is suitable for the detection of man-made objects on the seabed. In the past few years, several man-made object detection methods have been proposed for SSS and SAS imagery (Dobeck et al., 1997; Reed et al., 2003; Fawcett et al., 2006; Maussang et al., 2007; Groen et al., 2009; Williams and Groen, 2011). These methods are suitable for detecting possible locations of man-made objects on vast surveyed areas of the sea floor. Nonetheless, after this large-scale detection one might need to reacquire the detected targets for a closer and more in-detail inspection task, for which FLS is a more suitable option than SSS or SAS. Therefore, there is a need to perform automatic detection also with FLS in order to conduct fully autonomous detection missions with an ASV or an AUV.

Some generic obstacle detection methods for FLS have been proposed as well. Lu and Sang (1998) make use of image processing techniques and the near field acoustic scattering principles of underwater targets to estimate the two-dimensional position and size/shape of nearby targets. Guo et al. (1998) use the continuous image sequences generated by an electronic scanning FLS to achieve the aim of obstacle avoidance and visual navigation for an AUV. Here, they use a track-before-detect strategy to extract information contained in image sequences to estimate the dynamics of the AUV, then they apply a dynamic programming algorithm to solve the problem of detection. This method aims to reduce the computational cost to meet the realtime demand on obstacle avoidance and navigation of an AUV system.

However, the critical demand for realtime signal processing and the uncertainties of the AUV's dynamics make online detection of obstacles a challenging task. Martin et al. (2000) propose an obstacle detection method where the FOV of the FLS is sub-divided into various cells. The cells are filled with the raw intensity data collected from the FLS sensor. For each filled cell, a cell signature is computed. The maximum signature cell is extracted from the grid. This cell contains transformed target information such as range, bearing to target, and cell signature. Petillot et al. (2001) describe a framework for segmentation of sonar images, tracking of underwater objects and motion estimation. The realtime data flow (acoustic images) acquired with multibeam FLS is first segmented and relevant features are extracted. This framework is applied to the design of an obstacle avoidance and path planning system for underwater vehicles.

However, little attention has been paid to the specific problem of man-made object detection with FLS. Moreover, the aforementioned methods suffer from several limitations that preclude them from being a suitable option for seabed man-made object detection. In particular:

- Most detection algorithms rely on training data from a different area and hence do not take into account the particular environmental conditions where the surveying takes place. Thus, such methods are not able to dynamically deal with different seabed compositions in different regions.

- Generic obstacle detection algorithms do not consider domain-specific geometrical and physical knowledge about the man-made object to be detected.

Aiming to address and overcome these limitations, we propose a novel object detection algorithm for reacquisition of man-made objects on the seabed using multibeam FLS. To achieve this objective, we borrow ideas from the detection algorithm for SAS imagery proposed by Williams and Groen (2011). Our proposed method explicitly takes into account environmental characteristics and other geometrical knowledge about the problem domain.

Next, the proposed detection algorithm is described in Section B.2.1. Experimental results on real autonomous missions conducted at sea with the Gemellina ASV are presented in Section B.2.2. Lastly, directions for possible extensions of the proposed algorithm are given in Section B.2.3.

B.2.1 *Proposed Detection Algorithm*

We propose a detection algorithm whose overall objective is to detect underwater man-made objects of interest on FLS imagery. The algorithm needs to be made fast as it is intended to run in realtime onboard an autonomous vehicle with (possibly) limited computational resources. Since no human intervention is allowed, the algorithm is designed to deal with changing environmental conditions that have a direct effect on the collected sonar data. At the same time, the proposed method addresses the limitations stated in the previous section.

The main idea of the algorithm is to locate echo highlights that are locally higher than their background. As in the object detection algorithm intended for SAS imagery presented by Williams and Groen (2011), the successive phases of the algorithm are concatenated in such a way that computational costs are minimized by operating on smaller portions of the image at each phase. Also as in the aforementioned work, the integral-image representation is used to speed up the algorithm. We take advantage of the *a priori* knowledge about the object (i.e., shape, size, etc.) by applying filtering steps in order to consider only highlights corresponding to an actual object of interest. The following sections will describe each step of the detection algorithm that resulted from these considerations. Each step will be demonstrated on an example FLS image (see Figure 112).

B.2.1.1 *Region of Interest of the Sonar Image*

The proposed algorithm is designed to work on a rectangular ROI of the sonar images. This design criteria is based on two reasons. First, it allows avoidance of noisy or poor-quality areas of the sonar image produced by some sonars at certain ranges. Second, it aids speeding up the detection by not processing the entire image, but only a subregion of it. Hence, from the very beginning, every step discussed hereafter will be applied only inside the predetermined ROI.

The rectangular ROI is specified by the position of its top-left corner in a Cartesian coordinate system with origin at the sonar head. This coordinate system will be used by all the subsequent phases of the algorithm. In our particular case, using a maximum range of 25 m the top-left corner (x_r, y_r) of the rectangle was located at $x_r = -11.5$ m, $y_r = 23$ m from the sonar head, with a width, $w_r = 23$ m and a height, $h_r = 11$ m (see Figure 112a).

Naturally, the ROI can be enlarged to the entirety of the image if the particular sonar images used are of good quality all along the sonar range and if the computational resources of the vehicle allow for realtime processing of entire images.

B.2.1.2 *Integral Image*

An integral image (Viola and Jones, 2004), also called a summed area table, is a representation of an image that allows for fast computation of the sum of pixel values in a given rectangular area of the image. In subsequent stages of the algorithm, we will exploit this image representation for quickly assessing certain distinguishing characteristics of objects, such as background and echo levels. The fast calculations allowed by the use of the integral-image representation also make realtime detection possible. Thus, rather than operating on the pixel-based system of the sonar image, we immediately transform to an integral-image system, which contains equivalent information.

The corresponding integral image, I, of an original sonar image, A, is constructed as follows. The value at a location (x, y) in the integral image corresponds to the sum of pixel values above and to the left of (x, y), inclusive, in the original image, A. That is,

$$I(x, y) = \sum_{x' \leqslant x, y' \leqslant y} A(x', y'). \tag{26}$$

The integral image is quickly generated by applying the following recursive relation:

$$I(x, y) = I(x-1, y) + z(x, y), \tag{27}$$

where $z(x, y)$ is the cumulative sum of pixels in a row of the original image,

$$z(x, y) = z(x, y-1) + A(x, y). \tag{28}$$

As can be noticed, the integral image is computed in only one pass over the original image. Using the integral image, the sum of pixel values in any arbitrary rectangle in the image can be computed with only four array references to the integral image (without the need for referencing all the involved pixels in the original image). We will take advantage of this fact in our algorithm. The integral-image representation corresponding to the ROI of the sonar image in Figure 112a is shown in Figure 112b overlapped on the original image. It is worth noticing that once the integral image is calulated, the algorithm uses this representation in the subsequent stages rather than the original image.

B.2.1.3 *Background Estimation*

The first use of the integral image, I, is in the estimation of the sonar-image background map, B. The purpose of the background map is to establish the seabed reverberation level. Once established, the reverberation level will be used subsequently to determine locations of echo highlights in the image.

The reverberation level strongly depends on the seabed composition. Thus, using a predefined threshold for all possible seabed types to determine which pixels correspond to the seabed is not a reliable option. For example, the reverberation level of a soft muddy bottom will be lower than that of a bottom of hard-packed sand (Williams and Groen, 2011).

For this reason, an approach that defines the background level according to some global intensity average over the sonar image could fail catastrophically. Rather than using some global threshold, we argue that the background estimation should be performed locally in the image.

We do this local estimation by using two different-sized, concentric sliding windows: a bigger, outer window with an inner, smaller window laying inside the bigger one. For each pixel in the ROI of the sonar image, we compute the mean pixel value of the neighbor pixels laying in the bigger window, but we ignore the pixels laying in the inner window. Pixels in the inner window are ignored because, if an object were present, they would correspond to a high-intensity echo return related to the object.

Window sizes can be adjusted according to the object we want to search for. In our particular case, the bigger, outer window has width, $b_{ox} = 4$ m in the sonar's X axis and height, $b_{oy} = 4$ m in the sonar's Y axis; the smaller, inner window has width, $b_{ix} = 1$ m and height, $b_{iy} = 1$ m. The background score at location (x, y), $B(x, y)$, is then the mean pixel value in the bigger window centered around (x, y), ignoring pixels laying inside the smaller window.

The calculation of the two rectangles involved in the background value at a given location can be computed quickly using the integral-image representation. A total of only eight array references to the integral image are necessary to compute the values for the two rectangles (four references per rectangle). Specifically, the background score at pixel location (x, y) is calculated as

$$\begin{aligned}
B(x,y) = (n_o - n_i)^{-1} \times \\
[I(x - \frac{\delta_{ox}}{2}, y - \frac{\delta_{oy}}{2}) - I(x - \frac{\delta_{ox}}{2}, y + \frac{\delta_{oy}}{2}) \\
+I(x + \frac{\delta_{ox}}{2}, y + \frac{\delta_{oy}}{2}) - I(x + \frac{\delta_{ox}}{2}, y - \frac{\delta_{oy}}{2})] \\
-[I(x - \frac{\delta_{ix}}{2}, y - \frac{\delta_{iy}}{2}) - I(x - \frac{\delta_{ix}}{2}, y + \frac{\delta_{iy}}{2}) \\
+I(x + \frac{\delta_{ix}}{2}, y + \frac{\delta_{iy}}{2}) - I(x + \frac{\delta_{ix}}{2}, y - \frac{\delta_{iy}}{2})],
\end{aligned} \tag{29}$$

where $\delta_{ox}, \delta_{oy}, \delta_{ix}$ and δ_{iy} are the number of pixels that correspond to b_{ox}, b_{oy}, b_{ix} and b_{iy}, respectively, and n_o and n_i are the total number of pixels involved in the sums of the outer window rectangle and the inner window rectangle, respectively, so that the result is the mean pixel value of the pixels laying inside the outer window but not in the inner window*. The background map corresponding to the ROI of the sonar image in Figure 112a is shown in Figure 112c.

B.2.1.4 *Echo Estimation*

Once the background estimation is ready, the integral image is used again to construct an echo map. The purpose of the echo map is to help determine locations of high-intensity echo returns in the image that might have been produced by objects of interest.

The echo map is constructed using a single sliding window. The size of the sliding window used to construct the echo map is related to the size of the object we want to detect. Here, we use a window with width, $e_x = 1.5$ m and height, $e_y = 1.5$ m. For each pixel in the ROI of the sonar image, we compute the mean pixel value of the neighboring pixels laying in the window. The echo map value at location (x, y), $E(x, y)$, is then the mean pixel value in the window. Again thanks to the integral-image representation, each value can be computed quickly with only four array references to the integral image. Specifically, the echo map value at location (x, y) is calculated as

$$\begin{aligned}
E(x,y) = (n_e)^{-1} \times \\
[I(x - \frac{\delta_{ex}}{2}, y - \frac{\delta_{ey}}{2}) - I(x - \frac{\delta_{ex}}{2}, y + \frac{\delta_{ey}}{2}) \\
+I(x + \frac{\delta_{ex}}{2}, y + \frac{\delta_{ey}}{2}) - I(x + \frac{\delta_{ex}}{2}, y - \frac{\delta_{ey}}{2})],
\end{aligned} \tag{30}$$

where δ_{ex} and δ_{ey} are the number of pixels that correspond to e_x and e_y, respectively, and n_e is the total number of pixels involved in the sum of

* After the sea trials discussed below, we figured out that the background score calculation in Equation 29 does not produce the expected result. Instead, the calculation should be either performed on the original image or rather on the integral image but using the four rectangles that actually contribute to the background scoring.

the rectangle so that the final echo value is the mean pixel value inside the rectangle.

The echo map corresponding to the ROI of the sonar image in Figure 112a is shown in Figure 112d.

B.2.1.5 *Potential Alarms Determination*

After a background map, B, and an echo map, E, are constructed, the regions of the image that may possibly contain targets of interest are determined. Any pixel for which the echo map value is sufficiently higher than the corresponding background map value is declared to be a ROI that will receive further investigation. Specifically, if $E(x, y) > \beta B(x, y)$, then the pixel (x, y) is considered to be part of an echo highlight and therefore a potential alarm that will receive further examination. The scaling factor β adjusts the severity of the requirement for echo highlights. (Essentially, this defines that an echo highlight is when the pixel value is a certain amount more than the average value of the surrounding background.) We use $\beta = 1.2$ in our particular application.

This test is the first data-reduction stage of the detection process. It achieves a considerable reduction in pixels that must be examined further. In practice, more than 80% of the pixels are usually removed in this step. Thus, computational costs on all the subsequent stages are greatly reduced.

The binary result of the background and echo map comparison is shown in Figure 112e, where white pixels correspond to regions of potential alarms that will be investigated further. Also in this very same phase, a standard labeling algorithm is applied to the resulting binary image. Therefore, on the subsequent stages we will only work with the potential alarm "blobs" determined here. As can be noticed, there are three such blobs in Figure 112e in our case.

B.2.1.6 *Geometrical and Morphological Filtering*

Next, the determined potential alarm regions are filtered according to their geometrical and morphological properties. The purpose of this step is to filter out potential alarm regions with a geometry and/or morphology that do not correspond to the object we want to detect.

We use the major axis length and the minor axis length of the potential alarm region to determine if it must be discarded. Specifically, a region will be discarded if any of the following conditions is met:

- Region's major axis length is longer than a certain maximum major axis length, M_{max}.

- Region's major axis length is shorter than a certain minimum major axis length, M_{min}.

- Region's minor axis length is longer than a certain maximum minor axis length, m_{max}.

- Region's minor axis length is shorter than a certain minimum minor axis length, m_{min}.

Depending on the objects we want to detect, other region properties can be used for filtering, e.g. the circularity coefficient or the area. The map of potential alarms that remain after this stage are shown in Figure 112f.

B.2.1.7 *Echo Scoring and Thresholding*

Next, the echo score is calculated for the remaining potential alarms. For each alarm, i, its echo score, s_i, is the mean pixel value of all the pixels in the region, that is:

$$s_i = \sum_{\forall (x,y) \in A_i} (x,y), \tag{31}$$

where A_i is the set of pixels, (x, y) constituting the potential alarm i.

As the potential alarm blobs are not rectangular regions in general, the integral-image representation can not be used. Here however, the amount of pixels in each blob is typically small (as ensured by the previous geometrical and morphological filtering step), and therefore the computational performance of the algorithm is not compromised.

The echo scores are directly related to the intensity strength of the objects, which means that a detection threshold can be determined rigorously. Specifically, the threshold can be set such that we wish to detect any object for which the intensity strength exceeds a given level.

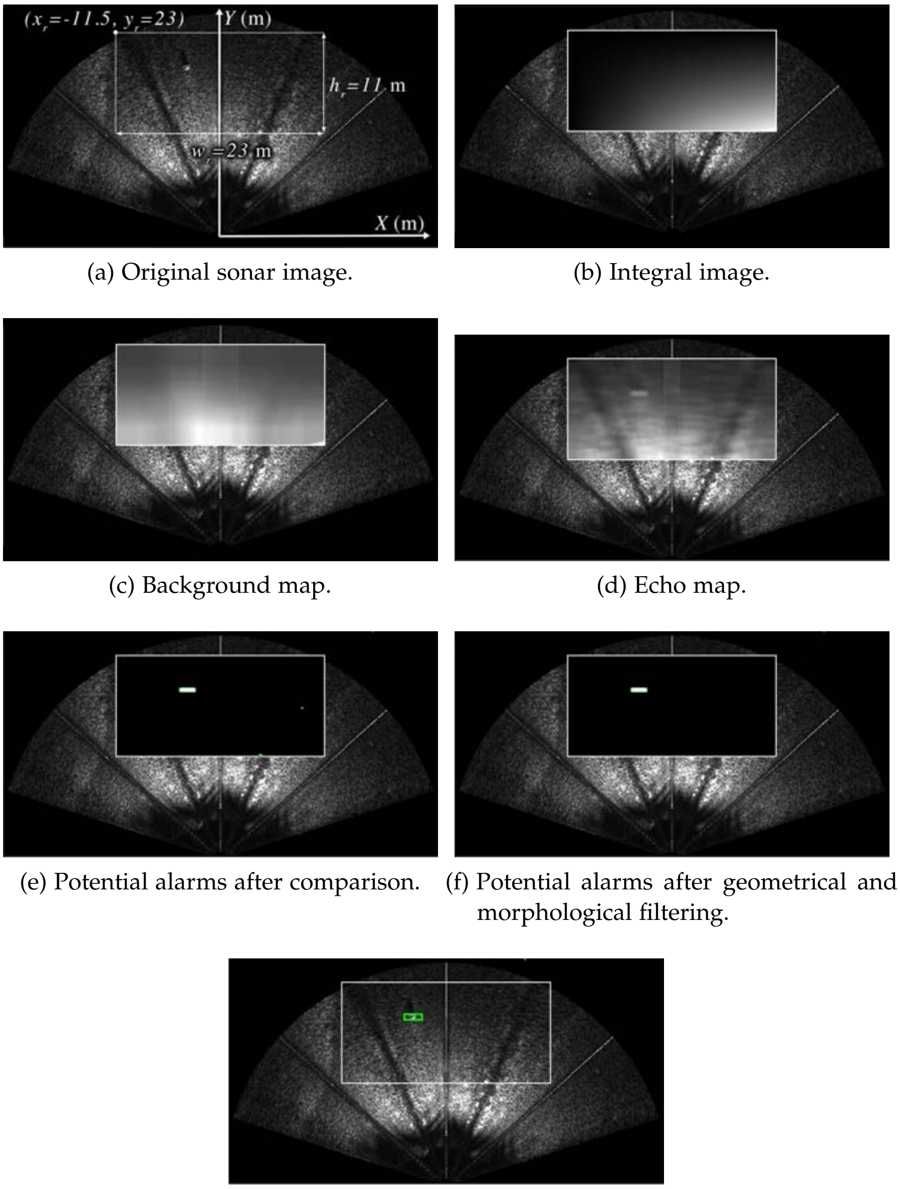
The final stage of the detection process removes those areas for which the echo score is below the desired threshold. The map of potential alarms that remain after this stage are shown in Figure 112g.

B.2.2 *Experimental Results*

In October 2011, NURC conducted the Autonomous Neutralization Trials (ANT'11) off the coast of Isola d'Elba, Italy. During this sea trial, four targets of two different shapes were deployed on the seafloor: cylinder shape and truncated cone shape. Targets were laying at depths ranging from 5 to 12 m. Multi-beam forward-looking sonar data were collected and processed in real time by the Gemellina ASV, which is equipped with a BlueView P900-130 900 kHz sonar. The sonar is mounted on a variable-depth pole (0 to 2 m depth) in the center of the ASV. The sonar can also be oriented by means of a pan and tilt unit (see Figure 113).

The detection system presented here was implemented and fully integrated into Gemellina's software system, hence being able to run in real time. The algorithm was implemented in C++ and integrated in the Gemellina's Mission Oriented Operating Suite (MOOS)-based (Newman, 2007) architecture.

In order to test the proposed detection algorithm, several autonomous detection missions were carried out with Gemellina ASV. Those missions



(a) Original sonar image.

(b) Integral image.

(c) Background map.

(d) Echo map.

(e) Potential alarms after comparison.

(f) Potential alarms after geometrical and morphological filtering.

(g) Final result after echo thresholding.

Figure 112: Object detection algorithm using FLS. A ROI of the original sonar image defined in the sonar coordinates system (a) is converted into its equivalent integral-image representation (b), which is then used to generate a background map (c) and an echo map (d). The comparison of the background map and the echo map generates a map of potential alarms (e). These potential alarms are then filtered according to their geometry and morphology (f). Next, an echo score is given to each remaining potential alarm. Finally, echo scores are thresholded producing discrete detections (g).

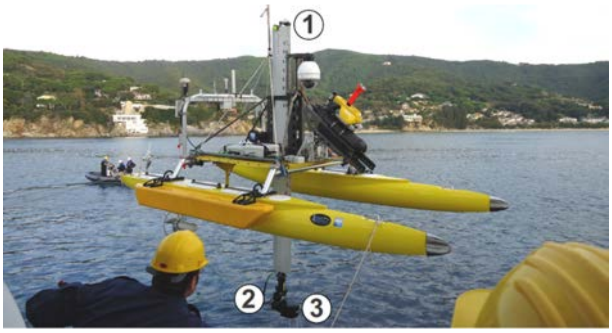
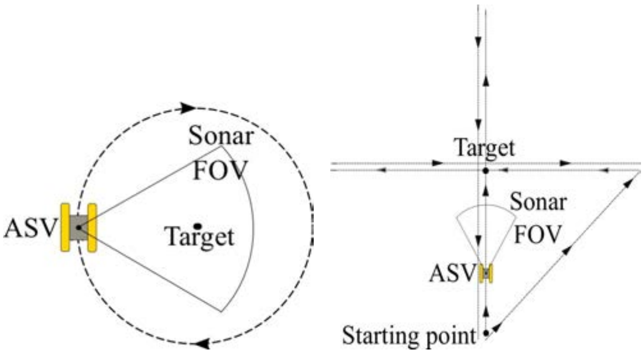


Figure 113: The Gemellina ASV being deployed. 1: variable-depth pole; 2: pan and tilt unit; 3: BlueView P900-130 sonar.

consisted of navigating around an *a priori* known target location tracing one of the two following target reacquisition patterns:

- The ASV circles around the target location, always keeping the target in the field of view of the sonar (see Figure 114a). Circle radius is 15 m.
- The ASV traces a cross pattern centered on the target location. Each *arm* of the cross is 40 m long and the vehicle goes along each arm of the cross twice. The target is not always in the field of view of the sonar. See Figure 114b.

The maximum range of the sonar was set to 25 m in all missions because from the field trials in these environmental conditions we concluded that this was the limit for this type of sonar.



(a) Circle-around mission (b) Cross-pattern mission

Figure 114: Trajectory patterns traced by the ASV around the known target locations.

During every mission, the detection algorithm was run in every sonar ping, at a rate of approximately 2 pings per second. If a positive detection occurred, its location in the world coordinate frame (i.e., UTM coordinates) was stored for further examination. At the end of the mission, the mean position of all the stored detections was computed. Here we assume that most of the detections will be true positives, i.e., its location will correspond to the real target since the targets were deployed on a flat, sandy bottom without clutter around them.

A total of eight missions, four for each type of target (cylinder and truncated cone), were run to test the algorithm. Table 4 shows, for each mission, the pattern traced by the ASV, the target depth and the distance error of the computed location with respect to the *a priori* known location.

MISSION PATTERN	TARGET TYPE	TARGET DEPTH	DISTANCE ERROR
Circle	Trunc. cone	6 m	3.8 m
Circle	Trunc. cone	10 m	1.6 m
Circle	Cylinder	5 m	1.1 m
Circle	Cylinder	12 m	3 m
Cross	Trunc. cone	6 m	2.6 m
Cross	Trunc. cone	10 m	1.4 m
Cross	Cylinder	5 m	0.6 m
Cross	Cylinder	12 m	1.6 m

Table 4: Target detection mission results.

B.2.3 *Possible Extensions*

A novel algorithm for the detection of underwater man-made objects on FLS imagery able to run in real time has been presented. The proposed pioneering and promising approach addresses limitations of existing generic underwater object detection algorithms by considering specific domain knowledge of the problem. By taking advantage of integral-image representations and by progressively reducing the computational burden at every stage, the algorithm allows for realtime detection onboard an ASV or an AUV, as has been demonstrated in real experiments conducted at sea.

If one can assume that the objects to be detected will not be located in cluttered environments, we believe it is interesting to investigate the possibility of adding density filtering to the algorithm, that is, to ignore alarms concentrated on a certain small area.

Considering shadows casted by objects on the seabed can potentially help reduce the number of false alarms. To achieve this, an extra step might be added to the algorithm's cascaded architecture to make only objects that cast a shadow prevail.

Finally, we believe that the presented algorithm can be extended to perform detection and tracking of a mid-water target, such as a moving Unmanned Underwater Vehicle (UUV). Promising preliminary experimental results were already obtained in this regard during the ANT'11 sea trials.

C

START-TO-GOAL PATH PLANNING ALGORITHMS

"Start-to-goal" path planning algorithms that find a collision-free path from a start state to a goal state have been used or mentioned throughout this thesis , by contrast to coverage path planning that seeks to cover an area. For instance, the A* path planner was used in Chapter 3 to connect coverage paths in different cells or regions. Here, we provide an introduction to some of these path planning algorithms.

C.1 SEARCH-BASED PLANNING: THE A* ALGORITHM

A* (Hart et al., 1968; Russell and Norvig, 2003) extends Dijkstra's algorithm by incorporating a heuristic to the cost estimation of paths from each node of a graph to the goal. (Dijkstra's algorithm (Dijkstra, 1959) is an efficient search algorithm for finding the optimal path in a graph when no other information apart from a graph is given.) That is, A* operates in a discrete or discretized environment, such as an occupancy grid. Each node is ordered according to the sum of its current path cost from the start and a heuristic estimation of its path cost to the goal. The node with the minimum value is evaluated first, since it is the most promising to belong to an optimal path from the start node to the goal node. When applied to path planning, the algorithm uses an optimistic heuristic to ensure that the shortest path is found. For instance, the straight line distance from a node to the goal can be used as an optimistic heuristic. A* is complete and optimal, that is, it tells whether or not a path to the goal exists and, if so, find the optimal path.

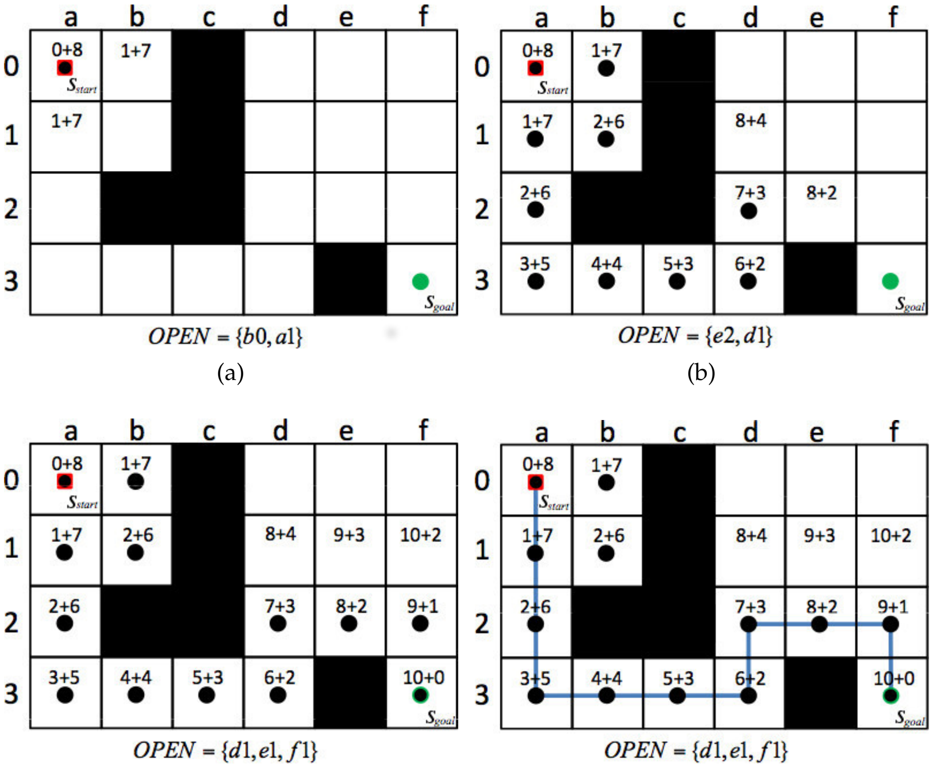
The complete A* algorithm is given in Algorithm 6. The algorithm plans a path from an initial state $s_{start} \in S$ to a goal state $s_{goal} \in S$, where S is the set of states in some finite state space, such as a graph or a grid. To do this, it stores an estimate $g(s)$ of the path cost from the initial state to each state s. Initially, $g(s) = \infty$ for all states $s \in S$. The algorithm begins by updating the path cost of the start state to be zero and then places this state onto a priority queue known as the OPEN list. Each element s in this queue is ordered according to the sum of its current path cost from the start, $g(s)$, and a heuristic estimate of its path cost to the goal, $h(s, s_{goal})$. The state with the minimum such sum is at the front of the priority queue. The heuristic $h(s, s_{goal})$ typically underestimates the cost of the optimal path from s to s_{goal} and is used to focus the search. The algorithm then pops the state s at the front of the queue and updates the cost of all states reachable from this state through a direct edge (its neighbors). If the cost of state s, $g(s)$, plus the cost of the edge between s and a neighboring state s', $c(s, s')$, is less than the current cost of state s', then the cost of s' is set to this new, lower value. If the cost of a neighboring state s' changes, it is inserted into the OPEN list. The algorithm continues popping states off the queue until it pops off the goal state. At this stage, if the heuristic is admissible, i.e. guaranteed to not overestimate the path cost from any state to the goal, then the path cost of s_{goal} is guaranteed to be optimal. The path found by the algorithm can be then retraced from the goal state via backtracking. Figure 115 illustrates the execution of A* on an example environment.

Algorithm 6: A*

```
Input: $s_{start}$ and $s_{goal}$ states.
foreach $s \in S$ do
    $g(s) \leftarrow \infty$
$g(s_{start}) = 0$
$\mathrm{OPEN} = \emptyset$
Insert $s_{start}$ into OPEN with value $g(s_{start}) + h(s_{start}, s_{goal})$
while $\arg\min_{s \in \mathrm{OPEN}}(g(s) + h(s, s_{goal})) \neq s_{goal}$ do
    Remove state $s$ from the front of OPEN
    foreach $s' \in \mathrm{Neighbors}(s)$ do
        if $g(s') > g(s) + c(s, s')$ then
            $g(s') = g(s) + c(s, s')$
            Insert $s'$ into OPEN with value $g(s') + h(s', s_{goal})$
return
```

C.2 SAMPLING-BASED PLANNING: PRM AND RRT

Deterministic algorithms such as A*, described above in Section C.1, rely on an explicit representation of the geometry of the space free of obstacles. Because of this, as the dimension of the configuration space grows, these planners become impractical. Therefore, these planners fail to find a solution in reasonable time in high-dimensional state spaces or for systems with differential constraints, such as a 7-DOF robotic arm or a second-order car model. Sampling-based path planning algorithms were developed to address these problems. These algorithms explore the space free of obstacles by taking collision-free samples at random and seeking to locally connect them with simple paths in order to obtain a global path from a start state to a goal state. This strategy capitalizes on the fact that it is much faster to check a given state for collision than explicitly constructing a path and has proven tremendously successful in the last two decades. However, these algorithms are not complete, that is, they do not report whether a solution exists or not. They can only be proven probabilistically complete, meaning that they will find a solution as the number of samples approaches infinity.



(a) (b) (c) (d)

Figure 115: Execution of the A* algorithm on an example environment at successive steps of the exploration process (a)-(c). (d) The path recovered via backtracking from the goal. Image credit: Hernandez (2012).

We next describe two of the most popular sampling-based path planning algorithms: Probabilistic Roadmap (PRM) (Kavraki et al., 1996) and Rapidly-exploring Random Tree (RRT) (LaValle and Kuffner, 2000). PRM constructs a graph representation of the free space by taking collision-free random samples of the workspace and connecting them with simple paths. Then, given a planning query with a start and a goal, the start and the goal are connected to the graph and the graph is searched for a solution using a standard graph search algorithm. By contrast, RRT incrementally grows a tree structure guided by random sampling and stops when the tree reaches the goal.

C.2.1 *The PRM Algorithm*

The PRM algorithm is given in Algorithm 7. It constructs a graph representation of the free space (called a roadmap) by taking a set of collision-free random samples. Then these samples are connected to their nearest neighbors by a local path planning generating simple paths that observe any dynamic constraints on the robot, e.g. straight line paths. Only the collision-free local paths are added to the roadmap. Once the roadmap is constructed, it can be searched using standard graph search algorithms (e.g. Dijkstra's algorithm) to find a solution to a given planning query consisting of a start state and a goal state. Note that the same roadmap can be reutilized for multiple planning queries, with no need to building it from scratch for each query. The PRM path planning procedure is illustrated in Figure 116.

C.2.2 *The RRT Algorithm*

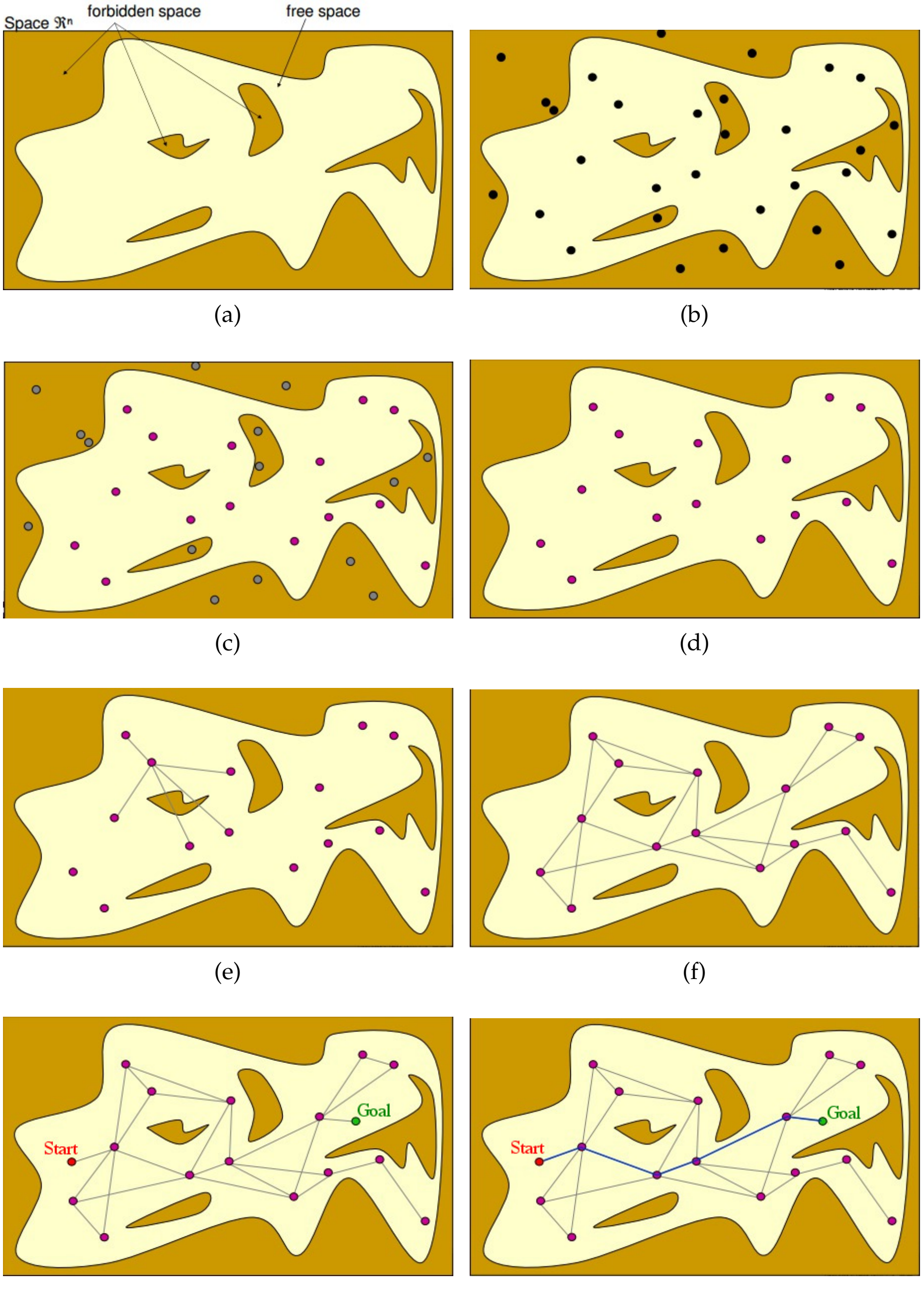
The RRT algorithm, listed in Algorithm 8, is a random sampling-based algorithm useful for exploring large states spaces that cannot be searched exhaustively. It iteratively chooses a random point p in the state space (by means of the RANDOM_STATE function) and attempts to extend the current search tree toward that point, as depicted in Figure 117a. The extension is performed by considering the random point p, and its nearest neighbor q, within the tree T. The NEW_STATE function determines a control input u_{new} that observes any dynamic constraints on the robot, and which, when applied to the robot at state q, results in some new state x_{new}. If NEW_STATE finds a new state and control input without violating dynamic constraints or colliding with an obstacle, the new state and input are added to the tree. This process of selecting a node and extending the tree from there repeats until a state within some tolerance of the goal, x_{goal}, is added to the tree. An example execution of the algorithm is shown in Figure 117b.

Algorithm 7: PRM

```
Input:
    • $n$: number of nodes to put in the roadmap
    • $k$: number of closest neighbors to examine for each configuration
Output: A roadmap $G = (V, E)$
V ← ∅
E ← ∅
while —V— ¡ n do
    repeat
        $q \leftarrow$ a random configuration
    until $q$ is collision-free
    $V \leftarrow V \cup \{q\}$
foreach $q \in V$ do
    $N_q \leftarrow$ the $k$ closest neighbors of $q$ chosen from $V$ according to some distance measure
    foreach do
        if $(q, q') \notin E$ and CONNECT$(q, q') \neq \emptyset$ then
            $E \leftarrow E \cup \{(q, q')\}$
return $G$
```

Algorithm 8: RRT

```
T.init($x_{start}$)
while $x_{goal} \notin T$ do
    $p \leftarrow$ RANDOM_STATE()
    $q \leftarrow$ NEAREST_NEIGHBOR$(p, T)$
    if NEW_STATE$(p, q, x_{new}, u_{new})$ then
        T.add_vertex($x_{new}$)
        T.add_edge$(q, x_{new}, u_{new})$
    return T
```



(a)

(b)

(c)

(d)

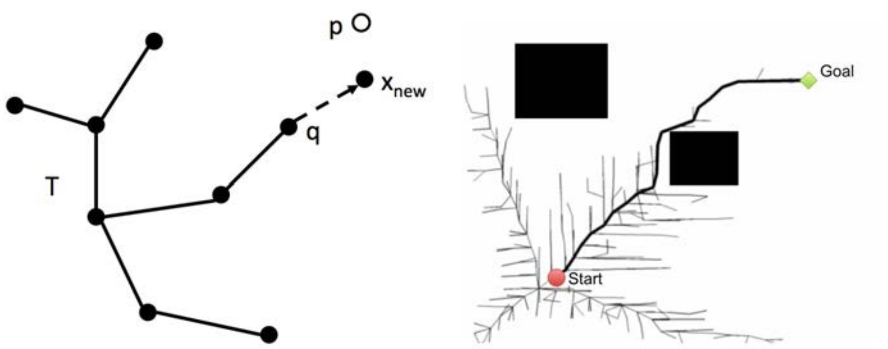
(e)

(f)

(g)

(h)

Figure 116: Execution of the PRM algorithm. Starting with a map of the environment (a), a set of random samples throughout the workspace is obtained (b). After checking them for collisions (c), only the samples in the free space are considered (d). Each sample (node of the PRM) is connected to its nearest neighbors by simple paths (edges) (e), only keeping the collision-free edges. As a result, a graph representing the free space of the environment is obtained (f). To solve a planning query, the start and goal states are connected to their nearest node in the graph, respectively, (g) and the graph is searched to obtain a solution (h). Image credit: Department of Electrical Engineering, University of Linköping (http://www.matlabinuse.com/11519).



(a)

(b)

Figure 117: Execution of the RRT algorithm. (a) An RRT extension. (b) Example path found with the RRT algorithm.

BIBLIOGRAPHY

Acar, E. and Choset, H. (2002a). Sensor-based coverage of unknown environments: Incremental construction of morse decompositions. *International Journal of Robotics Research*, 21(4):345–366. 15, 18

Acar, E., Choset, H., and Atkar, P. (2001). Complete sensor-based coverage with extended-range detectors: a hierarchical decomposition in terms of critical points and voronoi diagrams. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 3, pages 1305 –1311 vol.3. 22

Acar, E. U. and Choset, H. (2000). Critical point sensing in unknown environments. In *Proceedings of the 2000 IEEE International Conference on Robotics & Automation*. 18

Acar, E. U. and Choset, H. (2001). Robust sensor-based coverage of unstructured environments. In *Proc. IEEE/RSJ International Intelligent Robots and Systems Conference*, volume 1, pages 61–68. 15

Acar, E. U. and Choset, H. (2002b). Exploiting critical points to reduce positioning error for sensor-based navigation. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA 2002*, volume 4, pages 3831–3837. 40, 50

Acar, E. U., Choset, H., and Lee, J. Y. (2006). Sensor-based coverage with extended range detectors. *IEEE Transactions on Robotics*, 22(1):189–198. 22, 47

Acar, E. U., Choset, H., Rizzi, A. A., Atkar, P. N., and Hull, D. (2002). Morse decompositions for coverage tasks. *International Journal of Robotics Research*, 21(4):331–344. 12, 15, 17, 19, 22, 47, 53, 54, 81

Acar, E. U., Choset, H., Zhang, Y., and Schervish, M. (2003). Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods. *International Journal of Robotics Research*, 22(7-8):441–466. 9, 21

Adhami-Mihosseini, A., Aguiar, A., and Yazdanpanah, M. (2011). Seabed tracking of an autonomous underwater vehicle with nonlinear output regulation. In *Decision and Control and European Control Conference (CDC-ECC)*, pages 3928 – 3933. 106

Agmon, N., Hazon, N., and Kaminka, G. (2006). Constructing spanning trees for efficient multi-robot coverage. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1698 –1703. 42

Ahmadzadeh, A., Keller, J., Jadbabaie, A., and Kumar, V. (2006). An optimization-based approach to time critical cooperative surveillance and coverage with unmanned aerial vehicles. In *International Symposium on Experimental Robotics*. 43, 51

Alterovitz, R., Simeon, T., and Goldberg, K. (2007). The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty. In *Robotics: Science and Systems III (Proc. RSS 2007)*. 69

Amat, J., Monferrer, A., Batlle, J., and Cufi, X. (1999). Garbi: a low-cost underwater vehicle. *Microprocessors and Microsystems*, 23(2):61–67. 6

Applegate, D. L., Bixby, R. E., Chvatal, V., and Cook, W. J. (2006). *The Traveling Salesman Problem: A Computational Study*. Princeton University Press.

Arkin, E. M., Fekete, S. P., and Mitchell, J. S. (2000). Approximation algorithms for lawn mowing and milling. *Computational Geometry*, 17(1-2):25 – 50. 10

Arkin, E. M. and Hassin, R. (1994). Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Mathematics*, 55(3):197–218. 10

Atkar, P., Choset, H., and Rizzi, A. (2003). Towards optimal coverage of 2-dimensional surfaces embedded in ir3: choice of start curve. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 4, pages 3581 – 3587 vol.3. 36

Atkar, P., Conner, D., Greenfield, A., Choset, H., and Rizzi, A. (2009). Hierarchical segmentation of piecewise pseudoextruded surfaces for uniform coverage. *Automation Science and Engineering, IEEE Transactions on*, 6(1):107 –120. 49

Atkar, P., Greenfield, A., Conner, D., Choset, H., and Rizzi, A. (2005a). Hierarchical segmentation of surfaces embedded in r3 for auto-body painting. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 572 – 577. 36

Atkar, P., Greenfield, A. L., Conner, D. C., Choset, H., and Rizzi, A. (2005b). Uniform coverage of automotive surface patches. *The International Journal of Robotics Research*, 24(11):883 – 898. 9, 33, 36

Atkar, P. N., Choset, H., Rizzi, A. A., and Acar, E. U. (2001). Exact cellular decomposition of closed orientable surfaces embedded in r3. In *Proc. Int. Conf. Robotics and Automation*, volume 1, pages 699–704. 34, 49, 99

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., and Protasi, M. (1999). *Complexity and Approximation. Combinatorial optimization problems and their approximability properties*. Springer Verlag. 71

Bang, F. (1956). A bacterial disease of limulus polyphemus. *Bulletin of the John Hopkins Hospital*, 98(5):325–351. 2

Barkby, S., Williams, S., Pizarro, O., and Jakuba, M. (2012). Bathymetric particle filter slam using trajectory maps. *The International Journal of Robotics Research*. 80

Barrientos, A., Colorado, J., del Cerro, J., Martinez, A., Rossi, C., Sanz, D., and Valente, J. (2011). Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots. *Journal of Field Robotics*, 28(5):667–689. 44

Bartels, R. H., Beatty, J. C., and Barsky, B. A. (1987). *An Introduction to Splines for Use in Computer Graphics &Amp; Geometric Modeling*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 109

Batalin, M. A. and Sukhatme, G. S. (2002). Spreading out: A local approach to multi-robot coverage. In *Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems*, pages 373–382. 43

Batlle, J., Ridao, P., Garcia, R., Carreras, M., Cufí, X., El-Fakdi, A., Ribas, D., Nicosevici, T., Batlle, E., Oliver, G., Ortiz, A., and Antich., J. (2004). *Automation for the Maritime Industries*, chapter URIS: Underwater Robotic Intelligent System, pages 177–203. Instituto de Automática Industrial, Consejo Superior de Investigaciones Científicas. 6

Bennet, A., Leonard, J., and Bellingham, J. (1995). Bottom following for survey-class autonomous underwater vehicles. In *International Symposium on Unmanned Untethered Submersible Technology*, pages 327–336. 106

Bingham, B., Foley, B., Singh, H., Camilli, R., Delaporta, K., Eustice, R., Mallios, A., Mindell, D., Roman, C. N., and Sakellariou, D. (2010). Robotic tools for deep water archaeology: Surveying an ancient shipwreck with an autonomous underwater vehicle. *J. Field Robotics*, 27(6):702–717. 3, 68

Bosse, M., Nourani-Vatani, N., and Roberts, J. (2007). Coverage algorithms for an under-actuated car-like vehicle in an uncertain environment. In *Proc. IEEE Int Robotics and Automation Conf*, pages 698–703. 9, 17

Bretl, T. and Hutchinson, S. (2013). Robust coverage by a mobile robot of a planar workspace. In *Proc. International Conference on Robotics and Automation*. 40, 50

Brignone, L., Munaro, M., Allais, A., and Opderbecke, J. (2011). First sea trials of a laser aided three dimensional underwater image mosaicing technique. In *OCEANS, 2011 IEEE - Spain*, pages 1 –7.

Brunel, J. M., Salmi, C., Loncle, C., Vidal, N., and Letourneux, Y. (2005). Squalamine: a polyvalent drug of the future? *Current Cancer Drug Targets*, 5(4):267–72. 2

Burgard, W., Fox, D., and Thrun, S. (1997). Active mobile robot localization by entropy minimization. In *Advanced Mobile Robots, 1997. Proceedings., Second EUROMICRO workshop on*, pages 155–162. 69

Burns, B. and Brock, O. (2006). Sampling-based motion planning using uncertain knowledge. Technical report, University of Massachusetts at Amherst. 69

Butler, Z. J., Rizzi, A. A., and Hollis, R. L. (1999). Contact sensor-based coverage of rectilinear environments. In *Proc. IEEE Int Intelligent Control/Intelligent Systems and Semiotics Symposium*, pages 266–271. 25, 48

Butler, Z. J., Rizzi, A. A., and Hollis, R. L. (2000). Complete distributed coverage of rectilinear environments. In *Proc. of the Workshop on the Algorithmic Foundations of Robotics*. 42, 51

Caccia, M., Bono, R., Bruzzone, G., and Veruggio, G. (2003). Bottom-following for remotely operated vehicles. *Control Engineering Practice*, 11(4):461 – 470. 106

Caccia, M., Bruzzone, G., and Veruggio, G. (1999). Active sonar-based bottom-following for unmanned underwater vehicles. *Control Engineering Practice*, 7(4):459 – 468. 106

Candido, S. and Hutchinson, S. (2010). Minimum uncertainty robot path planning using a pomdp approach. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1408 –1413. 69, 70

Canny, J. F. (1988). Constructing roadmaps of semi-algebraic sets i: Completeness. *Artificial Intelligence*, 37(1-3):203–222. 15

Canny, J. F. (1993). An opportunistic global path planner. *Algorithmica*, 10:102–120. 15

Cao, Z. L., Huang, Y., and Hall, E. L. (1988). Region filling operations with random obstacle avoidance for mobile robotics. *Journal of Robotic Systems*, 5(2):87–102. 9, 10

Carreras, M., Batlle, J., and Ridao, P. (2001). Hybrid coordination of reinforcement learning-based behaviors for auv control. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 3, pages 1410–1415 vol.3. 6

Carreras, M., Candela, C., Ribas, D., Mallios, A., Magi, L., Vidal, E., Palomeras, N., and Ridao, P. (2013). Sparus ii, design of a lightweight hovering auv. In *5th International Workshop on Marine Technology*. 6

Carrillo, H., Latif, Y., Neira, J., and Castellanos, J. (2012). Fast minimum uncertainty search on a graph map representation. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2504–2511. 70

Castellanos, J., Tardos, J., and Schmidt, G. (1997). Building a global map of the environment of a mobile robot: the importance of correlations. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 2, pages 1053 –1059 vol.2. 26

Chakravorty, S. and Kumar, S. (2011). Generalized sampling-based motion planners. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(3):855 –866. 69

Cheng, P., Keller, J., and Kumar, V. (2008). Time-optimal uav trajectory planning for 3d urban structure coverage. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2750 –2757. 36, 49

Chin, W.-P. and Ntafos, S. (1991). Shortest watchman routes in simple polygons. *Discrete & Computational Geometry*, 6(1):9–31. 10

Choi, Y.-H., Lee, T.-K., Baek, S.-H., and Oh, S.-Y. (2009). Online complete coverage path planning for mobile robots based on linked spiral paths using constrained inverse distance transform. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems IROS 2009*, pages 5788–5793. 29

Choset, H. (2000). Coverage of known spaces: the boustrophedon cellular decomposition. *Autonomous Robots*, 9(3):247–253. 16

Choset, H. (2001). Coverage for robotics–a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31:113–126. 11, 12, 26, 33, 39

Choset, H., Acar, E., Rizzi, A. A., and Luntz, J. (2000a). Exact cellular decompositions in terms of critical points of morse functions. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA '00*, volume 3, pages 2270–2277. 14

Choset, H. and Burdick, J. (2000). Sensor-based exploration: The hierarchical generalized voronoi graph. *The International Journal of Robotics Research*, 19(2):96–125. 22

Choset, H. and Kortenkamp, D. (1999). Path planning and control for aercam, a free-flying inspection robot in space. *ASCE Journal of Aerospace Engineering*, 12(2):74–81. 22

Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., and Thrun, S. (2005). *Principles of Robot Motion: Theory, Algorithms, and Implementation*. The MIT Press. 13, 39, 47

Choset, H. and Pignon, P. (1997). Coverage path planning: The boustrophedon cellular decomposition. In *Proceedings of International Conference on Field and Service Robotics*. 12, 14, 47

Choset, H., Walker, S., Eiamsa-Ard, K., and Burdick, J. (2000b). Sensor-based exploration: Incremental construction of the hierarchical generalized voronoi graph. *The International Journal of Robotics Research*, 19(2):126–148. 22

Creuze, V., Jouvencel, B., and Baccou, P. (2001). Seabed following for small autonomous underwater vehicles. In *OCEANS, 2001. MTS/IEEE Conference and Exhibition*, volume 1, pages 369 –374 vol.1. 106

Danner, T. and Kavraki, L. (2000). Randomized planning for short inspection paths. In *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, volume 2, pages 971 –976 vol.2. 37

de Berg, M., Cheong, O., v. Kreveldand, M., and Overmars, M. (2008). *Computational Geometry*. Springer, 3rd edition edition. 22

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271. 181

Dobeck, G., Hyland, J., , and Smedley, L. (1997). Automated detection/classification of seamines in sonar imagery. In *Proc. SPIE International Society of Optics*, volume 3079, pages 90–110. 171

Easton, K. and Burdick, J. (2005). A coverage algorithm for multi-robot boundary inspection. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 727–734. 43, 51

El-Fakdi, A., Carreras, M., and Galceran, E. (2010). Two steps natural actor critic learning for underwater cable tracking. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2267–2272. 6

El-Fakdi, A., Tiano, A., Ridao, P., and Batlle, J. (2003). Identification of nonlinear models of unmanned underwater vehicles: Comparison between two identification methods. In *6th International Conference on Maneuvering and Control of Marine Craft*. 6

Elfes, A. (1987). Sonar-based real-world mapping and navigation. *Robotics and Automation, IEEE Journal of*, 3(3):249 –265. 26

Elibol, A., Gracias, N., and Garcia, R. (2012). Efficient image mosaicing for optical underwater mapping. In *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, pages 1 –4.

Englot, B. and Hover, F. (2012). Sampling-based coverage path planning for inspection of complex structures. In *International Conference on Automated Planning and Scheduling (ICAPS)*. xvii, 9, 37, 38, 49

Escartin, J., Garcia, R., Delaunoy, O., Ferrer, J., Gracias, N., Elibol, A., Cufi, X., Neumann, L., Fornari, D. J., Humphris, S. E., and Renard, J. (2008). Globally aligned photomosaic of the lucky strike hydrothermal vent field (mid-atlantic ridge, 37 deg 18.5 min n): Release of georeferenced data, mosaic construction, and viewing software. *Geochemistry, Geophysics, Geosystems*, 9(12):n/a–n/a. 3, 68

Eustice, R., Singh, H., and Leonard, J. (2006). Exactly sparse delayed-state filters for view-based slam. *Robotics, IEEE Transactions on*, 22(6):1100–1114. 3

Fairfield, N. and Wettergreen, D. (2008). Active localization on the ocean floor with multibeam sonar. In *OCEANS 2008*, pages 1–10. 69

Farsi, M., Ratcliff, K., Johnson, J. P., Allen, C. R., Karam, K. Z., and Pawson, R. (1994). Robot control system for window cleaning. In *Proc. American Control Conf*, volume 1, pages 994–995. 9

Fawcett, J., Crawford, A., Hopkin, D., Myers, V., and Zerr, B. (2006). Computer-aided detection of targets from the citadel trial klein sonar data. Technical report, Defence R & D Canada - Atlantic, Canada. 171

Fazli, P., Davoodi, A., Pasquier, P., and Mackworth, A. (2010). Complete and robust cooperative robot area coverage with limited range. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5577–5582. 42

Ferrer, J., Elibol, A., Delaunoy, O., Gracias, N., and Garcia, R. (2007). Large-area photo-mosaics using global alignment and navigation data. In *OCEANS 2007*, pages 1 –9.

Gabriely, Y. and Rimon, E. (2002). Spiral-stc: an on-line coverage algorithm of grid environments by a mobile robot. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA '02*, volume 1, pages 954–960. 27, 48

Gage, D. W. (1994). Randomized search strategies with imperfect sensors. In *Proc. SPIE, Mobile Robots VIII–Int. Soc. Optical Engineering*, pages 270–279, Boston, MA. 9

Garcia, E. and de Santos, P. G. (2004). Mobile-robot navigation with complete coverage of unstructured environments. *Robotics and Autonomous Systems*, 46:195–204. 21

Gonzalez, E., Alvarez, O., Diaz, Y., Parra, C., and Bustacara, C. (2005). Bsa: A complete coverage algorithm. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA 2005*, pages 2040–2044. 29

Gonzalez, J. and Stentz, A. (2009). Using linear landmarks for path planning with uncertainty in outdoor environments. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 1203 –1210. 70

Groen, J., Coiras, E., and Williams, D. (2009). Detection rate statistics in synthetic aperture sonar images. In *Proc. Intl. Conf. & Exh. Underwater Acoustic Measurements*, pages 367–374. 171

Guibas, L., Hsu, D., Kurniawati, H., and Rehman, E. (2008). Bounded uncertainty roadmaps for path planning. In *Proceedings of the Workshop on Algorithmic Foundations of Robotics*. 69

Guo, J., Cheng, S.-W., and Liu, T.-C. (1998). Auv obstacle avoidance and navigation using image sequences of a sector-scanning sonar. In *Proc. of the 1998 International Symposium on Underwater Technology*. 171

Guo, Y. and Balakrishnan, M. (2006). Complete coverage control for nonholonomic mobile robots in dynamic environments. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1704 –1709. 32

Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science, and Cybernetics*, SSC-4(2):100–107. 181

Hayes, M. and Gough, P. (1992). Broad-band synthetic aperture sonar. *IEEE Journal of Oceanic Engineering*, 17(1):80–94. 171

Hazon, N. and Kaminka, G. (2005). Redundancy, efficiency and robustness in multi-robot coverage. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 735 – 741. 42

Hazon, N., Mieli, F., and Kaminka, G. (2006). Towards robust on-line multi-robot coverage. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1710 –1715. 42, 51

Hernandez, E. (2012). *Path Planning With Homotopic Constraints For Autonomous Underwater Vehicles*. PhD thesis, University of Girona. 183

Hernandez, E., Carreras, M., Antich, J., Ridao, P., and Ortiz, A. (2011). A topologically guided path planner for an auv using homotopy classes. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2337–2343. 6

Hert, S., Tiwari, S., and Lumelsky, V. (1996). A terrain-covering algorithm for an auv. *Autonomous Robots*, 3:91–119. xvii, 9, 33, 34, 35, 49

Hodgkin, L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and exitation in nerve. *Journal of Physics London*, 117:500–544. 31

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. (1992). Surface reconstruction from unorganized points. *SIGGRAPH Comput. Graph.*, 26(2):71–78. 132

Hornung, A., Wurm, K., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). Octomap: an efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206. 107, 116

Houts, S., Rock, S., and McEwen, R. (2012). Aggressive terrain following for motion-constrained auvs. In *Autonomous Underwater Vehicles (AUV), 2012 IEEE/OES*, pages 1–7. 106

Huang, W. H. (2001). Optimal line-sweep-based decompositions for coverage algorithms. In *Proc. ICRA Robotics and Automation IEEE Int. Conf*, volume 1, pages 27–32. 39, 50

Huang, Y. and Gupta, K. (2008). Rrt-slam for motion planning with motion and map uncertainty for robot exploration. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1077 –1082. 69

Huang, Y. and Gupta, K. (2009). Collision-probability constrained prm for a manipulator with base pose uncertainty. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 1426 –1432. 69

Hudson, I., Jones, D., and Wigham, D. (2005). A review of the uses of work-class rovs for the benefits of science: Lessons learned from the serpent project. *The International Journal of the Society for Underwater Technology*, 26(3):83–88. 2

Hurtos, N. (2009). Integration of optical and acoustic sensor data for 3d underwater scene reconstruction. Master's thesis, University of Girona. 56

Hurtos, N., Ribas, D., Cufi, X., Petillot, Y., and Salvi., J. (2014). Fourier-based registration for robust fls mosaicing in low visibility underwater environments. *Journal of Field Robotics*. In press. 169

Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(11):1254–1259. 70, 72

IUCN (2013). Iucn red list of threatened species (ver. 2013.2). http://www.iucnredlist.org (Accessed: 10 December 2013).

Jimenez, P. A., Shirinzadeh, B., Nicholson, A., and Alici, G. (2007). Optimal area covering using genetic algorithms. In *Proc. IEEE/ASME international conference Advanced intelligent mechatronics*, pages 1–5. 39, 50

Jin, J. and Tang, L. (2011). Coverage path planning on three-dimensional terrain for arable farming. *Journal of Field Robotics*, 28(3):424–440. 36, 49

Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P., and Schaal, S. (2011). Stomp: Stochastic trajectory optimization for motion planning. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4569–4574. 116, 118

Karras, G., Bechlioulis, C., Abdella, H., Lakworthy, T., Kyriakopoulos, K., and Lane, D. (2013). A robust sonar servo control scheme for wall-following using an autonomous underwater vehicle. In *Intelligent Robots and Systems (IROS)*. 106

Kavraki, L., Svestka, P., Latombe, J.-C., and Overmars, M. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Robotics and Automation, IEEE Transactions on*, 12(4):566 –580. 69, 184

Kazhdan, M. and Hoppe, H. (2013). Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3):29:1–29:13. 131, 132

Kenny, A., Cato, I., Desprez, M., Fader, G., Schüttenhelm, R., and Side, J. (2003). An overview of seabed-mapping technologies in the context of marine habitat classification. *ICES Journal of Marine Science: Journal du Conseil*, 60(2):411–418.

Kewlani, G., Ishigami, G., and Iagnemma, K. (2009). Stochastic mobility-based path planning in uncertain environments. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 1183 –1189. 69

Kim, A. (2012). *Active Visual SLAM with Exploration for Autonomous Underwater Navigation*. PhD thesis, The University of Michigan. 40, 50, 69

Kurniawati, H., Hsu, D., and Lee, W. (2008). Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *Proceedings of the Robotics: Science and Systems*. 69

Larkum, A. W., Orth, R. J., Duarte, C. M., Walker, D. I., Kendrick, G. A., and McComb, A. J. (2006). Decline and recovery of seagrass ecosystems – the dynamics of change. In *SEAGRASSES: BIOLOGY, ECOLOGYAND CONSERVATION*, pages 551–565. Springer Netherlands.

Latombe, J. C. (1991). *Robot motion planning*. Kluwer Academic Publishers. 13, 39

LaValle, S. M. (2006). *Planning Algorithms*, chapter Chapter 4: The Configuration Sace, pages 130–131. Cambridge University Press. 10

LaValle, S. M. and Kuffner, J. J. (2000). Rapidly-exploring random trees: Progress and prospects. In *Proceedings Workshop on the Algorithmic Foundations of Robotics*. 69, 184

Lee, T.-S., Choi, J.-S., Lee, J.-H., and Lee, B.-H. (2009). 3-d terrain covering and map building algorithm for an auv. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems IROS 2009*, pages 4420–4425. 34

Li, F. and Klette, R. (2008). An approximate algorithm for solving the watchman route problem. In Sommer, G. and Klette, R., editors, *Robot Vision*, volume 4931 of *Lecture Notes in Computer Science*, pages 189–206. Springer Berlin Heidelberg. 10

Lin, S. and Kernighan, B. W. (1973). An effective heuristic algorithm for the travelling-salesman problem. *Operations Research*, 21:498–516.

Lloyd, S. (1982). Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129 – 137. 58

Loeb, J. (1914). Activation of the unfertilized egg by ultraviolet rays. *Science*, 40(1036):pp. 680–681. 2

Lu, Y. and Sang, E. (1998). Underwater target's size/shape dynamic analysis for fast target recognition using sonar images. In *Proceedings of the 1998 International Symposium on Underwater Technology*. 171

Lumelsky, V. J., Mukhopadhyay, S., and Sun, K. (1990). Dynamic path planning in sensor-based terrain acquisition. *IEEE Transactions on Robotics and Automation*, 6(4):462–472. 12, 33, 81

Luo, C. and Yang, S. (2002). A real-time cooperative sweeping strategy for multiple cleaning robots. In *Intelligent Control, 2002. Proceedings of the 2002 IEEE International Symposium on*, pages 660 – 665. 42

Luo, C. and Yang, S. (2008). A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments. *Neural Networks, IEEE Transactions on*, 19(7):1279 –1298. 31, 48

Luo, C., Yang, S., and Stacey, D. (2003). Real-time path planning with deadlock avoidance of multiple cleaning robots. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 3, pages 4080 – 4085 vol.3. 42, 51

Luo, C., Yang, S. X., Stacey, D. A., and Jofriet, J. C. (2002). A solution to vicinity problem of obstacles in complete coverage path planning. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA '02*, volume 1, pages 612–617. 29, 48

Lurton, X. (2002). *An introduction to underwater acoustics: principles and applications*. Springer. 166

Mannadiar, R. and Rekleitis, I. (2010). Optimal coverage of a known arbitrary environment. In *Proc. IEEE Int Robotics and Automation (ICRA) Conf*, pages 5525–5530. 39, 50

Martin, A., An, E., Nelson, K., and Smith, S. (2000). Obstacle detection by a forward looking sonar integrated in an autonomous underwater vehicle. In *Proc. OCEANS 2000 MTS/IEEE Conf. and Exhibition*, volume 1, pages 337–341. 171

Maussang, F., Chanussot, J., Hétet, A., and Amate, M. (2007). Mean-standard deviation representation of sonar images for echo detection: Application to sas images. *IEEE Journal of Oceanic Engineering*, 32(4):956–970. 171

Maza, I. and Ollero, A. (2007). *Distributed Autonomous Robotic Systems 6*, chapter Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms, pages 221–230. Springer. 44

Mazo, M. and et al. (2004). Robust area coverage using hybrid control. TELEC 2004, Santiago de Cuba. 39

MBARI (2013). Mb-system: Software for the processing and display of swath sonar data. http://www.mbari.org/data/mbsystem/. [Online; accessed 08-March-2013]. 154

Melchior, N. and Simmons, R. (2007). Particle rrt for path planning with uncertainty. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1617 –1624. 69

Melo, J. and Matos, A. (2012). Bottom estimation and following with the mares auv. In *Oceans, 2012*, pages 1 –8. 106

Menezes, R., Martins, F., Vieira, F. E., Silva, R., and Braga, M. (2007). A model for terrain coverage inspired by ant's alarm pheromones. In *Proceedings of* *the 2007 ACM symposium on Applied computing*, SAC '07, pages 728–732, New York, NY, USA. ACM. 43

Milnor, J. (1963). *Morse Theory*. Princeton University Press. 15, 17

Missiuro, P. and Roy, N. (2006). Adapting probabilistic roadmaps to handle uncertain maps. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1261 –1267. 69

Moravec, H. and Elfes, A. (1985). High resolution maps from wide angle sonar. In *Proc. IEEE Int. Conf. Robotics and Automation*, volume 2, pages 116–121. 26

Najjaran, H. and Kircanski, N. (2000). Path planning for a terrain scanner robot. In *Proc. 31st Int. Symp. Robotics*, pages 132–137, Montreal, QC, Canada. 9

Nakhaei, A. and Lamiraux, F. (2008). A framework for planning motions in stochastic maps. In *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on*, pages 1959 –1964. 69

Newman, P. (2007). Introduction to programming with moos. 177

Nicholson, J. and Healey, A. (2008). The present state of autonomous underwater vehicle (auv) applications and technologies. *Marine Technology Society Journal*, 42(1):44–51. 3

Nicosevici, T., Gracias, N., Negahdaripour, S., and Garcia, R. (2009). Efficient three-dimensional scene modeling and mosaicing. *Journal of Field Robotics*, 26:759–788. 6, 131

Oh, J. S., Choi, Y. H., Park, J. B., and Zheng, Y. (2004). Complete coverage navigation of cleaning robots using triangular-cell-based map. *Industrial Electronics, IEEE Transactions on*, 51(3):718–726. 27, 31

Oksanen, T. and Visala, A. (2009). Coverage path planning algorithms for agricultural field machines. *Journal of Field Robotics*, 26(8):651–668. 14

Ollis, M. and Stentz, A. (1997). Vision-based perception for an automated harvester. In *Proc. IEEE/RSJ Int Intelligent Robots and Systems IROS '97. Conf*, volume 3, pages 1838–1844. 9

Paduan, J., Caress, D., Clague, D., Paull, C., and Thomas, H. (2009). High-resolution mapping of mass wasting, tectonic, and volcanic hazards using the mbari mapping auv. In *International Conference on Seafloor Mapping for Geohazard Assessment*. 3

Palacin, J., Palleja, T., Valganon, I., Pernia, R., and Roca, J. (2005). Measuring coverage performances of a floor cleaning mobile robot using a vision system. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 4236–4241. 11

Palomeras, N., Carreras, M., Ridao, P., and Hernandez, E. (2006). Mission control system for dam inspection with an auv. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2551–2556. 6

Palomeras, N., El-Fakdi, A., Carreras, M., and Ridao, P. (2012). Cola2: A control architecture for auvs. *Oceanic Engineering, IEEE Journal of*, 37(4):695–716. 6, 81, 161

Papadopoulos, G., Kurniawati, H., and Patrikalakis, N. M. (2013). Asymptotically optimal inspection planning using systems with differential constraints. In *Proc. International Conference on Robotics and Automation*. 37, 49

Paull, L., Saeedi, S., Li, H., and Myers, V. (2010). An information gain based adaptive path planning method for an autonomous underwater vehicle using sidescan sonar. In *Automation Science and Engineering (CASE), 2010 IEEE Conference on*, pages 835–840. 32

Paull, L., SaeediGharahbolagh, S., Seto, M., and Li, H. (2012). Sensor driven online coverage planning for autonomous underwater vehicles. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2875–2880. 32, 48

Pepy, R. and Lambert, A. (2006). Safe path planning in an uncertain-configuration space using rrt. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5376 –5381. 70

Petillot, Y., Ruiz, I. T., and Lane, D. M. (2001). Underwater vehicle obstacle avoidance and path planning using a muli-beam forward looking sonar. *IEEE Journal of Oceanic Engineering*, 26. 171

Platt, R., Kaelbling, L., Lozano-Perez, T., and Tedrake, R. (2012). Non-gaussian belief space planning: Correctness and complexity. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4711–4717. 70

Platt, R., Tedrake, R., Kaelbling, L., and Lozano-perez, T. (2010). Belief space planning assuming maximum likelihood observations. In *Proceedings of the Robotics: Science and Systems*. 70

Prats, M., Perez, J., Fernandez, J., and Sanz, P. (2012a). An open source tool for simulation and supervision of underwater intervention missions. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2577–2582. 81, 127

Prats, M., Ribas, D., Palomeras, N., García, J. C., Nannen, V., Wirth, S., Fernández, J., Beltran, J. P., Campos, R., Ridao, P., Sanz, P. J., Oliver, G., Carreras, M., Gracias, N., Marin, R., and Ortiz, A. (2012b). Reconfigurable auv for intervention missions: a case study on underwater object recovery. *Intelligent Service Robotics*, 5(1):19–31. 6

Pratt, W. K. (2007). *Digital Image Processing*. John Wiley & Sons, Inc.

Prentice, S. and Roy, N. (November/December 2009). The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research*, 28(11-12):1448–1465. 70

Qiu, X., Song, J., Zhang, X., and Liu, S. (2006). A complete coverage path planning method for mobile robot in uncertain environments. In *Proc. Sixth World Congress Intelligent Control and Automation WCICA 2006*, volume 2, pages 8892–8896. 31

Reeb, G. (1946). Sur les points singuliers d'une forme de Pfaff complètement intégrable ou d'une fonction numérique. *Comptes Rendus Acad. Sciences*, 222:847–849. 20

Reed, S., Petillot, Y., and Bell, J. (2003). An automatic approach to the detection and extraction of mine features in sidescan sonar. *IEEE Journal of Oceanic Engineering*, 28:90–105. 171

Reif, J. H. and Sun, Z. (2001). *Algorithmic and Computational Robotics: New Directions*, chapter An efficient approximation algorithm for weighed region shortest path problem, pages 191–203. A. K. Peters. 10

Rekleitis, I., New, A., Rankin, E., and Choset, H. (2009). Efficient boustrophedon multi-robot coverage: an algorithmic approach. *Annals of Mathematics and Artificial Intelligence*, 52:109–142. 10.1007/s10472-009-9120-2. 41, 51

Ribas, D., Palomeras, N., Ridao, P., Carreras, M., and Hernandez, E. (2007). Ictineuauv wins the first sauc-e competition. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 151–156. 6

Ribas, D., Palomeras, N., Ridao, P., Carreras, M., and Mallios, A. (2012). Girona 500 auv, from survey to intervention. *IEEE/ASME Transactions on Mechatronics*, 17(1):46–53. 6, 159, 160

Ribas, D., Ridao, P., Tardós, J., and Neira, J. (2008). Underwater SLAM in man made structured environments. *Journal of Field Robotics*, 25(11-12):898–921. 6

Ridao, P., Batlle, J., and Carreras, M. (2002). O2ca2, a new object oriented control architecture for autonomy: the reactive layer. *Cont*, 10(8):857–873. 6

Ridao, P., Carreras, M., Ribas, D., and Garcia, R. (2010). Visual inspection of hydroelectric dams using an autonomous underwater vehicle. *Journal of Field Robotics*, 27(6):759–778. 3

Ristic, B., Arulampalam, S., and Gordon, N. (2004). *Beyond the Kalman Filter*. Artech House. 74, 75

ROS (2013). Robot operating system. http://www.ros.org/. [Online; accessed 15-September-2013]. 130

Roy, N., Burgard, W., Fox, D., and Thrun, S. (1999). Coastal navigation-mobile robot navigation with uncertainty in dynamic environments. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 1, pages 35–40 vol.1. 69

Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Pearson Education. 181

Serra, J. (1982). *Image Analysis and Mathematical Morphology*. Academic Press. 58, 99

Shermer, T. (1992). Recent results in art galleries [geometry]. *Proceedings of the IEEE*, 80(9):1384–1399. 10

Shivashankar, V., Jain, R., Kuter, U., and Nau, D. (2011). Real-time planning for covering an initially-unknown spatial environment. In *Proceedings of the Twenty-Fourth International Florida Artificial Intelligence Research Society Conference*. 27

Thrun, S. (1998). Learning metric-topological maps for indoor mobile robot navigation. *Artif. Intell.*, 99(1):21–71. 26

Thrun, S. (2003). Robotic mapping: a survey. In *Exploring Artificial Intelligence in the New Millennium*, pages 1–35. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 26

Tibshirani, R., Walther, G., and Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423. 58

Tully, S., Kantor, G., and Choset, H. (2010). Leap-frog path design for multi-robot cooperative localization. In Howard, A., Iagnemma, K., and Kelly, A., editors, *Field and Service Robotics*, volume 62 of *Springer Tracts in Advanced Robotics*, pages 307–317. Springer Berlin / Heidelberg. 40, 50

Valencia, R., Miro, J., Dissanayake, G., and Andrade-Cetto, J. (2012). Active pose slam. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1885 –1891. 69

van den Berg, J., Abbeel, P., and Goldberg, K. (2011). Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information. *The International Journal of Robotics Research*, 30(7):895–913. 70

van den Berg, J., Patil, S., and Alterovitz, R. (2012). Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278. 69

Viola, P. and Jones, M. (2004). Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154. 173

Wagner, I. A., Altshuler, Y., Yanovski, V., and Bruckstein, A. M. (2008). Cooperative cleaners: A study in ant robotics. *The International Journal of Robotics Research*, 27(1):127–151. 43, 51

Wagner, I. A., Lindenbaum, M., and Bruckstein, A. M. (1999). Distributed covering by ant-robots using evaporating traces. *IEEE Transactions on Robotics and Automation*, 15(5):918–933. 43

Ward, G. (2012). *The Rough Guide to the Titanic*. Rough Guides. 2

Williams, D. P. (2010). On optimal auv track-spacing for underwater mine detection. ICRA. 3, 68

Williams, D. P. and Groen, J. (2011). A fast physics-based, environmentally adaptive underwater object detection algorithm. In *Proc. IEEE - Spain OCEANS*, pages 1–7. 171, 172, 174

Williams, S. B., Pizarro, O., Mahon, I., and Johnson-Roberson, M. (2009). Simultaneous localisation and mapping and dense stereoscopic seafloor reconstruction using an auv. In Khatib, O., Kumar, V., and Pappas, G. J., editors, *Experimental Robotics*, volume 54 of *Springer Tracts in Advanced Robotics*, pages 407–416. Springer Berlin Heidelberg. 80

Wong, S. C. (2006). *Qualitative topological coverage of unknown environments by mobile robots*. PhD thesis, The University of Auckland. 23, 30

Wong, S. C. and MacDonald, B. A. (2003). A topological coverage algorithm for mobile robots. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS 2003)*, volume 2, pages 1685–1690. 23, 47

Xu, A., Virie, P., and Rekleitis, I. (2011). Optimal complete terrain coverage using an unmanned aerial vehicle. In *Proceedings of the 2011 IEEE International Conference on Robotics & Automation*. 39

Xu, L. (2011). *Graph Planning for Environmental Coverage*. PhD thesis, Carnegie Mellon University. 32, 43, 45, 48, 70

Yan, M. and Zhu, D. (2011). An algorithm of complete coverage path planning for autonomous underwater vehicles. *Key Engineering Materials*, 467-469:1377–1385. 31

Yang, R. and Pollefeys, M. (2003). Multi-resolution real-time stereo on commodity graphics hardware. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–211–I–217 vol.1. 131

Yang, S. X. and Luo, C. (2004). A neural network approach to complete coverage path planning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(1):718–724. 29, 48

Yasutomi, F., Yamada, M., and Tsukamoto, K. (1988). Cleaning robot control. In *Proc. Conf. IEEE Int Robotics and Automation*, pages 1839–1841. 9

Yoerger, D. R., Kelley, D. S., and Delaney, J. R. (2000). Fine-scale three-dimensional mapping of a deep-sea hydrothermal vent site using the jason rov system. *The International Journal of Robotics Research*, 19(11):1000–1014. 3, 68

Zandara, S., Ridao, P., Ribas, D., Mallios, A., and Palomer, A. (2013). Probabilistic surface matching for bathymetry based slam. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany. 6, 81

Zelinsky, A., Jarvis, R. A., Byrne, J. C., and Yuta, S. (1993). Planning paths of complete coverage of an unstructured environment by a mobile robot. In *Proceedings of International Conference on Advanced Robotics*, pages 533–538. 27, 48

Zheng, X., Jain, S., Koenig, S., and Kempe, D. (2005). Multi-robot forest coverage. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3852 – 3857. 42, 51

Zheng, X. and Koenig, S. (2007). Robot coverage of terrain with non-uniform traversability. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 3757 –3764. 42

DECLARATION

I hereby declare that his thesis contains no material which has been accepted for the award of any other degree or diploma in any university. To the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference has been made.

Girona, March 2014

Enric Galceran