Universitat de Girona

ADVANCED UNDERWATER VEHICLE MANIPULATION THROUGH REAL-TIME MOTION PLANNING

Dina Nagui Youakim Isaac

Doctoral Thesis

Advanced Underwater Vehicle Manipulation through Real-Time Motion Planning

Dina Nagui Youakim Isaac

2019

Doctoral Thesis

Advanced Underwater Vehicle Manipulation through Real-Time Motion Planning

Dina Nagui Youakim Isaac

2019

Doctoral Program in Technology

Supervised by:

Pere Ridao

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

Doctor of Philosophy

Certificate of Thesis Direction

Dr. Pere Ridao, members of the Departament d'Arquitectura i Tecnologia de Computadors of Universitat de Girona,

Declare:

That the work entitled *Advanced Underwater Vehicle Manipulation through Real-Time Motion Planning* presented by Dina Nagui Youakim Isaac to obtain the degree in Doctor of Philosophy has been developed under my supervision.

Therefore, in order to certify the aforesaid statement, I sign this document.

Girona, May 2019

Dr. Pere Ridao

Acknowledgements

This thesis is not only the result of a scientific journey with plenty of failure and success; but more importantly a personal one that taught me a lot. First, all the praise goes to God, my creator and my eternal companion, without you I would not exist, without your presence I could not have make it that far. After him, comes my parents and my brother whom by example taught me honesty, commitment and perseverance in every life aspect. Credit of this work goes to my life partner, Ferran, I can't thank you enough for believing in me and for your endless support, emotionally, financially, and scientifically. Sharing this adventure with you is just the beginning of many more waiting for us. My friends [Rabha, Mina, Amr, Mirette, Bishbeashy, Dina], the ones who from far away, have always been there showing all kind of support and confidence in me. I would like to dedicate this thesis to the soul of my grandma who, with her smiling face, has always been present with me. As well, to Souza's soul, the one who taught me how to live life to the full.
Special thanks goes to my advisor Dr. Pere Ridao, for the time he invested on this work, his inspiration that was a light and a guide along the way. This work would have not been the same without all CIRS members, specially those who have been closely involved in my research: Patryk, Albert, Roberto; and those who have been around sharing knowledge: Juan David, Klemen, Josep, Guillem, Eric, Eduard, Khadidja, Bruno, Roger. Not forgetting Nuno, Narcis and David for providing help and advice at various moments, as well as Lluis for all the hours he spent preparing the robot for experiments, and always being available to fix unexpected issues. Along this journey, I was blessed enough to witness a small dream coming true, by being part of the Robotics Institute at Carnegie Mellon for a short stay. For that I am thankful to Andrew, for giving me this opportunity; also for the time and knowledge we shared. Another special thanks for Prof. Maxim Likhachev, for all the fruitful meetings we had, without them this work would have not been possible. I am grateful for this chance that widely increased the depth of my knowledge in the challenging field of motion planning.
Lastly, I would like to thank the anonymous reviewers that contributed to improve this thesis with their valuable comments. Also the many reviewers that contributed to the different publications derived from this work.

List of Publications

Publications in the compendium

The presented thesis is a compendium of the following research articles:

- **D. Youakim**, P. Ridao, N. Palomeras, F. Spadafora, D. Ribas, and M. Muzzupappa. "MoveIt!: Autonomous Underwater Free-Floating Manipulation". In: *IEEE Robotics & Automation Magazine* 24.3 (2017), pp. 41–51
  Quality index: [JCR2017 , IEEE Robotics & Automation Magazine, Q1 (4/26)].

- **D. Youakim** and P. Ridao. "Motion planning survey for autonomous mobile manipulators underwater manipulator case study". In: *Robotics and Autonomous Systems* 107 (2018), pp. 20–44
  Quality index: [JCR2017 , Robotics & Autonomous Systems, Q2 (9/26)].

- **D. Youakim**, P. Cieslak, A. Dornbush, A. Palomer, P. Ridao, and M. Likhachev. "Multi-Representation, Multi-Heuristic A* Search-based Motion Planning for a Free-floating Underwater Vehicle Manipulator System in Unknown Environment". submitted on December 2018
  Submitted to Journal of Field Robotics on December 2018
  Quality index: [JCR2017 , Journal of Field Robotics, Q1 (6/26)].

Publications derived from this thesis

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

- **D. Youakim**, A. Dornbush, M. Likhachev, and P. Ridao. "Motion Planning for an Underwater Mobile Manipulator by Exploiting Loose Coupling". In: *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on.* IEEE. 2018, pp. 23–30

- A. Palomer, P. Ridao, **D. Youakim**, D. Ribas, J. Forest, and Y. Petillot. "3D Laser Scanner for Underwater Manipulation". In: *Sensors* 18.4 (2018), p. 1086

- R. Simoni, P. Ridao, P. Cieslak, and **D. Youakim**. "A Novel Obstacle Avoidance Approach for an I-AUV: Preliminary Simulation Results". In: *New Horizon For Underwater Intervention Missions: From Current Technologies to Future Applications IROS Workshop.* IEEE. 2018

- R. Simoni, P. Ridao, P. Cieslak, and **D. Youakim**. "A Novel Obstacle Avoidance Approach for an I-AUV". in: *Autonomous Underwater Vehicle (AUV), 2018 IEEE*. IEEE. 2018

Acronyms

AHRS Attitude and Heading Reference System.

ARA* Anytime Repairing A*.

AUV Autonomous Underwater Vehicle.

BFS Breadth First Search.

CHOMP Covariant Hamiltonian Optimization for Motion Planning.

CIRS Underwater Robotics and Vision Research Center.

cola2 Component Oriented Layer-based Architecture v.2.

CS Configuration Space.

DoF Degree of Freedom.

DVL Doppler Velocity Log.

EUCLID Euclidean.

FCL Flexible Collision Library.

FOV Field of View.

GPS Global Positioning System.

I-AUV Intervention Autonomous Underwater Vehicle.

IK Inverse Kinematics.

IMR Inspection Maintance Repair.

LARA* Lazy Anytime Repairing A*.

MI Manipulability Index.

MP Motion Primitives.

OMPL Open Motion Planning Library.

PRM Probabilistic Roadmap Method.

ROV Remotely Operated Vehicle.

RRT Rapidly Exploring Random Trees.

SBL Single-query Bi-directional Lazy collision checking.

SMPL Search Motion Planning Library.

SRDF Semantic Robot Description File.

STOMP Stochastic Trajectory Optimization for Motion Planning.

STRIDE Search Tree with Resolution Independent Density Estimation.

TP Task Priority.

TPRC Task-Priority Redundancy Control.

UdG Universitat de Girona.

URDF Universal Robot Description File.

UUV Unmanned Underwater Vehicle.

UVMS Underwater Vehicle Manipulation System.

WS Workspace.

List of Figures

List of Tables

Contents

Abstract

A key challenge in autonomous mobile manipulation is the ability to determine in real-time how to safely execute complex tasks when placed in an unknown world. Motion Planning has been widely used in terrestrial and aerial robots to cope with such challenges, while it stayed unexplored for underwater intervention. In the last few years, Intervention Autonomous Underwater Vehicles (I-AUVs) became subject of broad interest in research, with only few real demonstrations mostly relying on variations of the task-priority redundancy control framework.

In response to those arising needs, this thesis focused on advancing the state of the art by investigating the use of Motion Planning to increase the autonomy of I-AUVs in the context of "Inspection, Maintenance, Repair" missions in unknown environments.

Through our work, we initially present a modeling and integration of our I-AUV using common terrestrial mobile manipulation framework *MoveIt!*, showing for the first time motion planning for an I-AUVs in the presence of virtual obstacles. Then, based on our observations of our previous demonstrations, and in order to base our choice of the motion planning technique on a solid scientific foundation, we performed a deep analysis of the state of the art motion planning techniques. We created benchmarks, and compared 17 motion planners in 5 different scenarios and came out with guidelines for choosing the best fitting method for given requirements. Later, we identified planner specifications for underwater intervention: 1) Real-Time response for a high Degree of Freedom (DoF) system, 2) Consistency, 3) Efficient trajectories in terms of traveled distance, safety, and system loose coupling utilization. As a consequence, we propose a new motion planning algorithm under the umbrella of search-based method, that exploits the loose coupling nature of an I-AUV while generating consistent, efficient, and safe trajectories in unknown environments.

Both simulation and experimental (in water tanks) results are presented for various stages of the work, showing the flow and validating the efficiency and potential of the developed algorithm. The proposed method, conveniently integrated within the robot system's architecture, increases the reliability of the I-AUV performing intervention, both safely and robustly when operating in unknown terrains.

Resum

La capacitat de decidir en temps real com executar de manera segura uns tasca complexa en un entorn desconegut és un repte clau en la manipulació mòbil autònoma. Per abordar-ho, s'utilitzen habitualment tècniques de Planificació de Moviment tant en robots terrestres com aeris, mentre la seva aplicació a l'àmbit submarí roman inexplorada. Durant els darrers anys, l'interès de la comunitat científica pels I-AUVs ha crescut significativament, havent fructificat les primeres demostracions experimentals basades en variacions del mètode de control de redundància utilitzant tasques amb prioritat. Aquesta tesi doctoral avança l'estat de l'art investigant l'ús dels mètodes de Planificació de Moviment per augmentar l'autonomia dels I-AUVs per aplicacions d'Inspecció, Manteniment i Reparació executades en entorns desconeguts. A través del nostre treball, presentem la modelització i integració del nostre I-AUV en *MoveIt!*, un entorn per a la programació d'aplicacions de manipulació mòbil comunament utilitzat en robòtica terrestre. Amb l'objectiu de fonamentar científicament l'elecció del mètode de Planificació de Moviment apropiat al nostre problema, s'ha dut a terme un anàlisi comparatiu de l'estat de l'art. S'han definit 5 escenaris de manipulació representatius (*benchmarks*) i s'han comparat 17 planificadors diferents. L'anàlisi dels resultats ens ha permès establir les guies per a l'elecció de la tècnica més apropiada a les nostres necessitats. Posteriorment, s'han identificat les especificacions desitjades pels planificadors en aplicacions d'intervenció submarina: 1) Resposta a temps real d'un sistema amb un elevat nombre de graus de llibertat, 2) Consistència 3) Generació de trajectòries eficients en termes de distància i seguretat, així com l'ús de sistemes dèbilment acoblats. A partir dels resultats de l'anàlisi, es proposa un nou algoritme de planificació de moviment, dintre de la família dels mètodes de cerca, que explota l'acoblament dèbil entre el manipulador i el vehicle, generant, al mateix temps, trajectòries consistents, eficients i segures en entorns desconeguts. Al llarg de la recerca duta a terme s'han utilitzat resultats en simulació i experimentals (en tanc d'aigua), per a validar l'eficiència i el potencial de l'algoritme. El mètode proposat, convenientment integrat en el sistema de control, incrementa la fiabilitat del robot que desenvolupa la intervenció, millorant la seguretat i robustesa operant en un entorn desconegut.

Resumen

La capacidad de decidir en tiempo real cómo ejecutar de manera segura una tarea compleja en un entorno desconocido es un reto clave en la manipulación móvil autónoma. Para abordarlo, se utilizan habitualmente técnicas de Planificación de Movimiento tanto en robots terrestres como aéreos, mientras su aplicación al ámbito submarino permanece inexplorada. Durante los últimos años, el interés de la comunidad científica por los I-AUVs ha crecido significativamente, habiendo fructificado las primeras demostraciones experimentales basadas en variaciones del método de control de redundancia utilizando tareas con prioridad. Esta tesis doctoral avanza el estado del arte investigando el uso de los métodos de Planificación de Movimiento para aumentar la autonomía de los I-AUVs en aplicaciones de Inspección, Mantenimiento y Reparación ejecutadas en entornos desconocidos. A través de nuestro trabajo, presentamos la modelización e integración de nuestro I-AUV en *MoveIt!*, un entorno para la programación de aplicaciones de manipulación móvil comúnmente utilizado en robótica terrestre. Con el objetivo de fundamentar científicamente la elección del método de Planificación de Movimiento apropiado a nuestro problema, se ha llevado a cabo un análisis comparativo del estado del arte. Se han definido 5 escenarios de manipulación representativos (*benchmarks*) y se han comparado 17 planificadores diferentes. El análisis de los resultados nos ha permitido establecer las guías para la elección de la técnica más apropiada a nuestras necesidades. Posteriormente, se han identificado las especificaciones deseadas para los planificadores en aplicaciones de intervención submarina: 1) Respuesta en tiempo real de un sistema con un elevado número de grados de libertad, 2) Consistencia 3) Generación de trayectorias eficientes en términos de distancia y seguridad, así como el uso de sistemas débilmente acoplados. A partir de los resultados del análisis, se propone un nuevo algoritmo de planificación de movimiento, dentro de la familia de los métodos de búsqueda, que explota el acoplamiento débil entre el manipulador y el vehículo, generando, al mismo tiempo, trayectorias consistentes, eficientes y seguras en entornos desconocidos. A lo largo de la investigación llevada a cabo se han utilizado resultados en simulación y experimentales (en tanque de agua), para validar la eficiencia y el potencial del algoritmo. El método propuesto, convenientemente integrado en el sistema de control, incrementa la fiabilidad del robot que desarrolla la intervención, mejorando la seguridad y robustez operando en un entorno desconocido.

1

Introduction

In this chapter we introduce the main problems that have been covered by this thesis: the use of real-time motion planning for underwater intervention in unknown environments. The motive of this work is introduced in Section 1.1, presenting the background of underwater intervention and what has been done so far in the field, as well as the background of using motion planning for autonomous manipulation. Next, we present the objectives of the thesis in Section 1.2 and we briefly describe, in Section 1.3, the context in which this work has been carried out. Lastly, the organization of the thesis document is presented in Section 1.4.

1.1 Motivation

1.1.1 Underwater Intervention

Nowadays, Autonomous Underwater Vehicles (AUVs) are used for survey missions, while other set of arising applications demand intervention capabilities. Such applications include marine rescue, marine science, archaeology and the "Inspection, Maintenance, Repair" (IMR) operations common in the offshore industries. Currently, Remotely Operated Vehicles (ROVs) equipped with tele-operated robotics arm, are used for such interventions. However, their use suffers from its dependence on human involvement to manage the complex cooperation between the vehicle and the arm pilots. In addition, their running cost is very high, making the use of ROVs not practical on the long term.

As a consequence, during the last 20 years researchers were motivated to consider the natural evolution of the intervention ROVs, the I-AUVs, as a practical low cost solution (an extensive list of references and a broader introduction on the IAUVs can be found in [8]).

Pioneering work involving UVMS technology appeared in the early 90s with OTTER (MBARI) [9], ODIN (UH) [10], VORTEX (IFREMER) [11] underwater vehicles. While, field demonstrations were only presented by the 1^{st} decade of the 21^{st} century. Due to the difficulty of demonstrations, most of the trials were carried out in mock-up or constrained sea environments, focusing on two types of applications:

Object Search And Recovery:
The first result based on floating manipulation, was achieved in 2009 in the SAUVIM project [12], [13]. It demonstrated the capability of searching for an object whose position was roughly known *a priori*. The object was endowed with artificial landmarks and the robot autonomously located and hooked it with a recovery device while hovering. A 6-ton vehicle was used and hence the mass of the arm did not cause significant disturbances. In 2012, the same task was approached in the RAUVI project [14] using a lighter vehicle (<200Kg). First, the object was searched using a downward-looking camera and photomosaicing techniques and next, the object was *hooked* autonomously in a water tank. Later on, during the TRIDENT project [15], the experiment was extended in a harbour environment using the 7-DOF arm endowed with a 3 fingered hand, demonstrating the first multipurpose object search and recovery strategy. In contrast with the autonomous trials, innovative recent work [16] has been carried out to bring the ROVs to a new level, where a humanoid diving robot outfitted with human vision, haptic force feedback and an artificial brain, has been successfully used for recovery missions.

Inspection, Maintenance, and Repair (IMR):
Given the importance of IMR tasks for the offshore industry to routinely inspect and maintain sub-merged infrastructures, recent research focused on representative tasks like "Valve Turning" and "Connector Plug/unplug" demonstrated in mock-up environments. Such automation has followed two different patterns:

- **Docking-Base Intervention** The first fully autonomous intervention at sea, was demonstrated by ALIVE [17] 1.5 Ton I-AUV. A mechanical scanning imaging sonar was used to locate and home to a subsea panel, and visual servoing techniques were used for docking the vehicle using 2 hydraulic grasps. Once the vehicle was

docked, a hydraulic 7-DOF manipulator was used to open/close a valve. A similar approach was proposed in the TRITON project [18] to demonstrate docking to a custom sub-sea panel, fixed-based manipulation for both valve turning and hot stab connection. An active localization strategy employing a Sum-of-Gaussian filter and range measurements to the subsea panel was used to discover its position and then, to home onto it. Next, visual servoing methods based on the *a priori* known appearance of the panel were used to autonomously dock the robot into a funnel-shaped docking station.

- **Free-Floating Base Intervention** A more challenging approach is to perform the intervention with the AUV floating. This approach comes with its own issues to solve in terms of AUV and arm coordination to complete the mission. A first autonomous free-floating valve-turning was carried out based on a Learning-by-Demonstration paradigm within the PANDORA project [19], [20]. Later, a kinematic-based task priority [21] approach has been used by the authors to demonstrate the same valve turning intervention. More recently, a compliant motion control of the UVMS end-effector while in contact with an underwater pipe has been demonstrated [22] using force torque sensing.

The previously developed solutions relied on variations of Task-Priority Redundancy Control (TPRC) where a set of control tasks are being executed respecting a pre-defined priority. Even though such tasks might include end-effector guidance or obstacle avoidance, current implementations do not guarantee the success of intervention in proximity of obstacles in challenging, unknown environments as would be expected in a real sea-environment. As a consequence, using a global motion planning approach is the next step to take profit of the planner's knowledge of the environment and move towards more realistic demonstrations of the actual interventions.

1.1.2 Motion Planning

On the other hand, **Motion Planning** has been a fundamental topic in robotics [23, 24, 25], used to describe the process of breaking down a desired motion task into discrete collision-free motions that satisfy one or more constraints and possibly optimizing some aspect of the movement (i.e. length, consumed energy, safety). For decades, it has been a key functionality of autonomous robots, specially mobile manipulators where the high dimensionality of the system introduces more challenges. Other challenging factors are triggered by various sources of uncertainties in perception and robot localization, as well as dealing with unstructured changing environments. As a consequence, the adoption of certain motion planning strategy is critical in order to cope with the given challenges of a particular domain or problem.

Planning for a High Dimensional Space: as discussed in [26] and [27], planning for high dimensionality introduces additional challenges:

1. The increase of the system DoF, resulting into performance degradation of planning techniques due to the computational complexity, even in simple environments.

2. Considering typical human tasks in unstructured environments imposes additional difficulties: For example, a ground mobile manipulator moving a tray would require

the end-effector to move on a constrained trajectory rather than simply reaching a specific location.

3. For successful grasping and manipulation to occur, a minimal to zero tolerance around the goal constraint is needed, suggesting a fine discretization of the high-dimensional state space.

4. While goal constrains are usually defined in the workspace of the end-effector, the feasibility check for a given state is performed in joint space. An effective and fast kinematic conversion between both is thus required.

As mentioned earlier, complex, but common, task requirements are those imposing a real-time response to a rapidly changing world. Often, existing motion planners make assumptions that are too restrictive for unstructured environments and too computationally expensive to satisfy such requirements. Researchers typically make assumptions to relax such constraints. For example, assuming that complete models of objects in the environment are available *a priori* (or may be acquired through sensors), or that the environment remains static during the interaction. All these challenges and assumptions have not yet been investigated in the context of planning for an UVMS. Whereas the focus of this work is on motion planning for mobile manipulation and considering that it is a widely spread topic, we dedicate a full chapter to cover its various aspects with a thorough analysis of the existing approaches, along with their pros and cons under different conditions.

1.2 Objectives

The work included in this thesis has the purpose of *developing a real-time motion planning strategy to perform advanced manipulation for an UVMS in unknown environments, within a framework incorporating system modeling, collision-checking, kinematics, and sensing capabilities.*

This general goal can be broken down into the following more specific objectives:

- **Develop a first solution for an I-AUV using Motion Planning:** To model and integrate the existing Girona500 architecture *cola2* [28] with a motion planning stack using "MoveIt!" framework.

- **Perform an in-depth analysis of state of the art Motion Planning techniques:** to define benchmarks, qualitative and quantitative metrics for comparison, and to come out with a set of guidelines facilitating the choice of a planning strategy for a given system or domain. In addition, use this analysis as a baseline to choose an appropriate motion planning strategy to deploy into our solution.

- **Propose a real-time motion planner for an I-AUV:** To develop a motion planner that satisfies the requirements for underwater intervention, as observed from our previous experiments: consistency, safety, real-time response in unknown environments, and efficiency in terms of trajectories length, and loose coupling between the system components (less accurate, high motion range floating vehicle vs. limited, more accurate manipulator).

- **Validate experimentally the proposed approach:** First, to undertake the systems integration required to validate the proposed motion planning algorithm on

a real I-AUV (GIRONA500 & ECA arm). Then, to demonstrate, through water tank experiments, the viability of the proposed solution to tackle a representative intervention mission.

1.3 Context

The work presented in this thesis has been supported by the FI 2016 grant from the *Secretaria d'Universitats i Recerca del Departament d'Economia i Coneixement de la Generalitat de Catalunya* and has been developed at the Underwater Robotics and Vision Research Center (CIRS) research group of the Universitat de Girona (UdG), which is part of the VICOROB research institute. The group started researching in underwater vision and robotics in 1992 and it is currently formed by pre-doctoral researchers, engineers, technicians, postdoctoral fellows and permanent staff. The group is a leading team in the research and development of AUVs for mapping and intervention. It has participated in several European-funded and National-funded projects (of both basic and applied research) and it has also been involved in technology transfer projects and contracts with companies and institutions worldwide.
The group has developed several AUV prototypes and has currently two fully operative robots: Sparus II [29], a torpedo-shaped vehicle winner of multi-domain robotics competition Eurathlon 2014, 2015 and 2017, and Girona500 [30] a reconfigurable AUV for both survey and intervention. Girona500 along a 4 DoF ECA/CSIP Arm have been used during this thesis for both the simulation and water tank demonstrations.
The line of research in autonomous underwater manipulation started at CIRS since 2009, before this work, it followed two different paradigms: *Learning by Demonstration* [20, 31, 32], and *Control-based (TPRC)* [22, 33]. Simultaneously in CIRS, various path planning techniques for survey missions has been and still being developed: initially a topological path planner using homotopy classes was deployed to plan for AUVs in 2-D [34]. Then, in the work of [35], coverage path planning have been explored, while an online approach for mapping and planning has been developed in [36]. Recently, View Planning is being developed in [37, 38] for mapping and inspection of unexplored structures.
In this context, the work presented in this thesis is a normal progression towards connecting these two lines of research by adopting, for the first time, motion planning for underwater intervention. In addition, this thesis benefited from a research stay at Carnegie Mellon University (USA) under the supervision of Prof. Maxim Likhachev, a research associate professor at the Robotics Institute. His research is focused on search-based motion planning for navigation and manipulation, for both terrestrial and aerial robots [27, 39, 40, 41].

1.4 Document Structure

The rest of the manuscript is organized in a way that presents the incremental and progressive development of the thesis as follow:

Chapter 2: Motion Planning for a Free-Floating I-AUV covers the system modeling and initial implementation of an I-AUV using motion planning in a simplified environment with virtual obstacles.

Chapter 3: Motion Planning Survey - UVMS Case Study is an exhaustive analysis of state of the art motion planning methods. It defines underwater intervention

benchmarks, and compares seventeen algorithm across those benchmarks, using qualitative and quantitative methods. Finally it introduces a set of guidelines for the choice of the best techniques given a set of requirements.

Chapter 4: Multi-Representation, Multi-Heuristic A* Motion Planning for a Free-Floating UVMSs in Unknown Environment is a proposal of a search-based motion planner algorithm based on exploiting system loose coupling. The proposed method has been verified in simulation and in a water tank. The results have been analyzed to verify its efficiency from length, safety and base vs. arm motion optimization perspectives.

Chapter 5: Results and Discussion lays out the results obtained during this thesis, following the objectives previously presented.

Chapter 6. Conclusions, discussion and Future Work it concludes and discusses the work presented in this thesis, and suggests possible extension as future work.

2

Motion Planning for a Free-Floating I-AUV

In this chapter, we demonstrate the use of *MoveIt!* for the first time for underwater manipulation. As a mobile manipulator framework, *MoveIt!* incorporates the latest advances in motion planning, manipulation, 3D perception, kinematics, control and navigation. It has been widely used for terrestrial robots, specially industrial arms but never explored for underwater. We describe the modeling of our I-AUV using Girona500 vehicle and ECA 4 DoF arm. The proposed solution has been validated in water tank, performing two different interventions: valve turning in the presence of virtual obstacles, and connector plug/unplug. The proposed work and the validation results were detailed and published in the following journal paper:

Title: MoveIt!: Autonomous Underwater Free-Floating Manipulation
Authors: **D. Youakim**, P. Ridao, N. Palomeras, F. Spadafora, D. Ribas, and M. Muzzupappa
Journal: IEEE Robotics & Automation Magazine
Volume: 24, Number: 3, Pages: 41–51, Published: 2017
Quality index: JCR2017 , IEEE Robotics & Automation Magazine, Q1 (4/26)

D. Youakim, P. Ridao, N. Palomeras, F. Spadafora, D. Ribas, and M. Muzzupappa. “MoveIt!: Autonomous Underwater Free-Floating Manipulation”. *IEEE Robotics & Automation Magazine*. Vol. 24, issue 3 (2017) : 41–51

https://doi.org/10.1109/MRA.2016.2636369

Abstract

Today, autonomous underwater vehicles (AUVs) are mostly used for survey missions, but many existing applications require manipulation capabilities, such as the maintenance of permanent observatories, submerged oil wells, cabled sensor networks, and pipes; the deployment and recovery of benthic stations; or the search and recovery of black boxes. Currently, these tasks require the use of work-class remotely operated vehicles (ROVs) deployed from vessels equipped with dynamic positioning, leaving such solutions expensive to adopt. To face these challenges during the last 25 years, scientists have researched the idea of increasing the autonomy of underwater intervention systems.

3

Motion Planning Survey - UVMS Case Study

Following our observations from the previously preformed demonstrations, we realized the necessity to explore the various elements affecting the behaviour of a motion planner, in more depth to be able to meet the requirements of an I-AUV. To reach this purpose, in this chapter we present an exhaustive analysis of the state of the art motion planning techniques, we created underwater benchmarks, defined qualitative and quantitative measures; and compared 17 motion planning algorithms along five benchmarks, to come out with a set of guidelines. This extensive theoretical and experimental survey has been published in the following journal paper:

Title: Motion planning survey for autonomous mobile manipulators underwater manipulator case study
Authors: **D. Youakim** and P. Ridao
Journal: Robotics and Autonomous Systems
Volume: 107, Pages: 20–44, Published: 2018
Quality index: JCR2017 , Robotics & Autonomous Systems, Q2 (9/26)

Robotics and Autonomous Systems 107 (2018) 20–44

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Motion planning survey for autonomous mobile manipulators underwater manipulator case study

Dina Youakim *, Pere Ridao

Computer Vision & Robotics Research Institute, University of Girona, Spain

ARTICLE INFO

Article history:
Received 17 January 2018
Received in revised form 14 April 2018
Accepted 7 May 2018
Available online 1 June 2018

Keywords:
Motion planning
Survey
MoveIt!
Manipulation
Underwater
Autonomous
Obstacle avoidance

ABSTRACT

Sampling-based, search-based, and optimization-based motion planners are just some of the different approaches developed for motion planning problems. Given the wide variety of application tackled by autonomous mobile manipulators, the question "which planner to choose" may be tough. In this paper, we review the state of the art of the most common approaches, and present a set of benchmarks with the aim to provide not only a theoretical review but also a qualitative/quantitative comparison of the algorithms. Our purpose is to provide an insight and analyze their performance with respect to different metrics. The results are based on an Underwater Vehicle Manipulator System *UVMS*, although they can be extended to terrestrial and aerial robots as well. The paper uses these results to formalize a set of guidelines for the selection process of the most appropriate approach, for a given problem/requirements.

1. Introduction

Motion planning is a fundamental topic in robotics [1–3] which deals with the problem of finding a collision-free path going from an initial to a target configuration. Its basic form is *the piano mover's problem* which assumes a robot is a point. It has evolved through time to address a number of variations, depending on the specifications of the system and the environment in which it is expected to operate.

During the last few years several works have compared and analyzed the motion planning algorithms. The basic theory and the most common motion planning approaches have been reviewed in [4]. On the other hand, [5] and [6] each focused on just one common approach, being the sampling-based and heuristic-based respectively. Similarly [7] focused on the theory of classical and heuristic-based motion planning for navigation. In [8], the authors presented an analysis of probabilistic-based algorithms, discussing their performance regarding dynamic obstacles and narrow passages. Some other studies have been concerned with different robotics domains. For instance, the work presented in [9] focused on Unmanned Aerial Vehicles, [10] dealt with autonomous vehicles and [11] presented a more thorough algorithms taxonomy for 3D path planning. A quantitative comparison of motion planning algorithms for an aerial manipulator using *MoveIt!* was presented in [13] through a single test case. Similarly in [14], underwater bench-marking was covered. This paper focused on explaining a framework for simulating the environment using UWSIM [15] through one case study. Other surveys addressed different research problems related to motion planning. In this direction, [16] studied the coverage path-planning problem and [17] dealt with motion planning in dynamic environments, which is becoming a challenging topic of research.

Regarding benchmarking, some work has been done related to the quantitative evaluation of the techniques. For instance, in [18] the authors defined their own benchmarks to carry out a comparative study among a set of optimal motion planners. In [17], the focus was reviewing techniques for motion planning in dynamic environments. A ROS-based [19] framework using *MoveIt!* [12], was proposed in [20] for benchmarking sampling-based motion planning. Their purpose was to set-up guidelines for a terrestrial robots benchmarks database, along with the infrastructure to get comparative results. In contrast, the work in [21] is one of a kind, being focused on dual-arm manipulation; summarizing the state of the art of different approaches, ranging from low-level control to high level task planning and execution. Finally, [22] presented an early but unique work through an experimental evaluation of different collision detection mechanisms as well as their impact on the motion planning problem.

The intended contribution of the present work, with respect to the state-of-the-art surveys, is to advance beyond a theoretical-only review of the literature. This paper provides extensive comparisons based on kinematic simulations of the UVMS system. New statistical measures are proposed to provide a meaningful

* Corresponding author.
E-mail address: dina.isaac@udg.edu (D. Youakim).

https://doi.org/10.1016/j.robot.2018.05.006
0921-8890/

qualitative/quantitative comparison for existing/new planners. In addition, we present new underwater benchmarks, with the purpose of complementing those already available in the literature. Finally, to maintain a standardized and re-usable solution, the results presented are based on the latest ROS/*MoveIt!* framework for mobile manipulation. A total set of seventeen representative algorithms have been tested using five illustrative benchmarks and compared against seven metrics; with the purpose of providing clear and structured guidance for other researchers, on how to choose the most suitable motion planner for their application. To implement a full solution for mobile manipulators, it would be common to use hybrid frameworks (as will be summarized in Section 5) merging motion planning and reactive control techniques. For instance having two levels of obstacle avoidance: with reactive avoidance at the low level, in addition to the motion planning one. In this paper we will focus only on comparing the performance of the motion planning algorithms. It is worth noting that all the surveyed algorithms could be integrated with existing reactive control techniques.

The paper is organized as follows: Sections 2 and 3, present general motion planning concepts and requirements to be taken into account before deciding the approach to be used. Special attention is given to manipulation planning as explained in Section 4. Section 5 presents the theory of different motion planning approaches, with highlights of the pros and cons of each, based on the literature. In Section 7, we detail our own UVMS system as well as the tools and methodology we have followed. The main contribution of the work is in Section 8 where the comparative results and analysis are detailed. Finally, we conclude with guidelines and discussion in Section 9.

2. Common concepts

2.1. System properties

The feasibility of a motion planning approach is highly dependent on the characteristics of both the robot and the environment. Below are the common terminologies used to define a planning problem:

- **Configuration-Space (CS):** alternatively called Joint Space. A robot configuration is a vector of values representing the state of each movable joint, usually expressed as a vector of joint positions (either angular or prismatic): $q = (q_1, q_2, \ldots, q_n)$. The number of Degrees Of Freedom (DoFs) is the number of joints (n). The set of feasible configurations that avoids collision with obstacles is called the free space C_{free}. Its complement in CS is called the obstacle region $C_{obstacle}$. Planning in the Configuration Space may constitute a bottleneck for systems with many DoFs like mobile manipulators or humanoids.
- **Work-Space (WS):** interchangeably named End-Effector space, refers to the environment where the robot is allowed to move. It is either 2-D where the robot moves in a plane or 3-D which is equivalent to motion in the real world. A redundant system is a system for which the same end-effector pose may be reached with more than one valid configuration.
- **State-Space (SS):** a state represents the condition of the robot. It is the set of all feasible states and this is usually infinite. A state can be represented in either the configuration-space or the work-space, it can also be discrete or continuous.
- **Path:** in the *CS*, a path is a continuous curve C connecting two configurations q and q'. A *trajectory* is a path $C(t)$ parameterized by time t.

Any motion planning problem can be represented by an initial start state S_{start} and a goal state S_{goal}. A goal can be represented either in the *CS* where each joint has to attain a specific value, or in the *WS* where the end-effector of the robot has to reach a defined pose (2-D or 3-D). A plan constitutes a sequence of actions (either in the CS or WS depending on the planning space) to be taken to move the robot from the start to the goal, successfully taking into account one or more constraints. Section 3 will discuss the constraints in greater depth.

2.2. Motion planner properties

- **Computation Time:** the computational cost of the planner in terms of running time, i.e. the time spent by the planner to generate a valid plan.
- **Completeness:** An algorithm is complete if it finds a solution whenever one exists. Normally, it is a result of exact algorithms that build an exact representation of the world without losing any information. Usually completeness is traded-off against efficiency, as an accurate representation of the world may decrease efficiency. Two weaker notions exist to relax these conditions: "Resolution-Completeness" which means that the planner is able to find a plan if one exists, and if the resolution of the environment discretization is fine enough to capture relevant information. "Probabilistic-Completeness" planners refer to those that as more as they spend time planning, the probability of finding a solution, if one exists, increases to 1. Usually their performance is measured by their rate of convergence.
- **Optimality:** An algorithm is said to be optimal with respect to a certain criterion if it is able to find a plan that reaches the goal while optimizing such a criterion (e.g. length, execution time, energy consumption). As in the case of completeness, there are two weaker notions of optimality: "Resolution-Optimality" and "Probabilistic-Optimality".
- **Offline vs. Online:** An offline-planner is the one which does offline pre-processing to easily provide an online plan. On the other hand, the online planner incrementally processes the information to compute the plan at the time it is requested. Online planners can adapt to unexpected changes in the system/environment, an essential requirement for autonomous systems. Designing such a planner, and simultaneously keeping time-efficiency remains a challenge.
- **Local vs. Global:** local planners rely on local information in the neighborhood of the current state of the robot, contrarily global ones rely on system sensors to be able to perceive the global state of the environment and plan accordingly.

2.3. Path quality

An important aspect to be considered when evaluating and choosing a planner is the quality of the plans/paths produced. As in [23], the common metrics to evaluate path quality are:

- **Path Length:** typically, it is desirable to produce short paths, while maintaining planning efficiency. Since maintaining both is challenging, some techniques reduce the path length in a post-processing step after the plan generation.
- **Path Clearance:** the aim of any autonomous system is to generate collision-free paths. It is desirable that the generated path keeps at least a minimum distance away from obstacles. Moreover, traveling along high clearance paths reduces the chances of collisions due to various uncertainties (e.g. robot localization). This can also be treated as a post-processing step.

- **Path smoothness & stability:** describes avoidance of jerky motions for critical joints. In addition, it ensures stable transition through the path, specially for humanoids.

3. Motion planning requirements

A basic functionality of autonomous robots is to be able to generate collision-free paths. It becomes more challenging for high dimensional systems and in unstructured environments. Moreover, The various sources of uncertainties in perception and robot localization add to the challenges. As suggested in [24], to succeed in unstructured environments, robots have to carefully select task-specific features and identify relevant real-world characteristics to reduce the state space without affecting the performance of the task (e.g. using *projection* from the state space to a low dimensional Euclidean space, and plan in that Euclidean space instead). In this direction, a categorization of additional motion planning requirements was detailed in [25] together with their impact on the planner's choices. A summary is given below.

3.1. Planning in dynamic environment

Two types of dynamic environment relate to: (1) Robot knowledge about the environment ahead of planning: partial/uncertain knowledge and unexplored parts constitute a dynamism that will lead to changes in the robot's perceived map of the world. Thus, planning has to adapt accordingly. (2) Time-variant environments, which regardless of the current level of robot knowledge, there is a possibility of future changes, e.g. moving obstacles. For the system to deal with such changes, it has to be equipped with real-time replanning capabilities to adapt accordingly.

3.2. Planning with uncertainties

Robots cannot rely completely on their knowledge about their surroundings since perceiving the environment is a challenge in itself. They have to autonomously and continuously acquire the information necessary for decision making while navigating. Moreover, they cannot assume that their actions will succeed reliably. Instead, they have to continuously monitor their effect on the environment and sometimes take corrective actions. Many existing, well-established motion planning techniques rely on perfect knowledge of the world and perfect control of the environment, which has become an unrealistic assumption and makes them unable to cope with fully autonomous system challenges. Probabilistic robotics is a recent approach which is able to incorporate different sources of uncertainties within robot navigation and motion planning.

3.3. Planning with constraints

In addition to achieving the goal state, motion planning may involve one or more constraints. Some of those constraints are not optional, like avoiding collisions and keeping the joint angles within limits. Others, may arise due to the mechanical nature of the robot or related to the task to be executed.

- Differential Constraints: arise from both the kinematics (e.g. non-holonomic robot) and the dynamics (e.g. high-order constraints like acceleration) of the robotic system.
- Task Constraints: related to the nature of the task to be executed. For example, an end-effector pose constraint is needed for the task of transporting a glass filled with liquid, the robot has to keep the glass in an upright vertical position in order to successfully carry out the task. Another example is visibility constraints, like keeping a target object within the field of view of the robot.

4. Manipulation — planning in high dimensional space

We place emphasis on autonomous mobile manipulation, especially on collision-free motion for the end-effector (i.e. goals defined in the 3D-workspace). Manipulation includes moving objects of varying dimensions by pushing or pulling, and prehensile and non-prehensile grasping of smaller objects. Both, [24] and [26], discussed additional challenges of motion planning for manipulation:

(1) Adding a manipulator to a mobile robot increases its reachable workspace and its DoF accordingly. Motion planning for robots with many degrees of freedom is computationally expensive, even in structured environments, due to its high-dimensionality.
(2) Considering unstructured environments imposes additional difficulties. In addition, manipulation tasks may require the end-effector to move on a constrained trajectory rather than simply to reach a specific location. Each of these increase the challenge of the motion planning problem.
(3) For successful grasping and manipulation to occur, a minimal to zero tolerance around the goal constraint may be allowed, suggesting the need for a fine discretization of the state space.
(4) While goal constrains are usually defined in the workspace of the end-effector, the feasibility check for a given state is performed in joint space. An effective and fast kinematic conversion between both is thus required. Probably, most complex task requirements are those imposing a real-time response to a rapidly changing world. Often, existing motion planners make assumptions that are too restrictive for unstructured environments and too computationally expensive to satisfy such requirements. Researchers typically make assumptions to relax such constraints. For example, assuming that complete models of objects in the environment are available *a priori* (or may be acquired through sensors), or that the environment remains static during the interaction.

5. Motion planners categories

The various approaches in the literature, differ in their choice of the free-space representation, as well as the efficiency metrics they consider. When choosing an approach to solve a given problem, we might consider one or more of those factors (e.g. how exact is the free-space representation, time/memory usage ...). For manipulators, where the major challenge is the high dimensionality, the balance between trajectory quality and performance is a key trade-off, given the real-time and computational constraints of an autonomous robot. In this section, we present a brief description of the most common motion planning techniques, including some basic algorithms, not necessarily suitable for high-dimensional systems, but presented for completeness.

5.1. Potential fields

Given the difficulty of fully representing the free-space, the Potential Fields method is an alternative approach which explores the space incrementally moving towards the goal. The incremental exploration is guided by assuming the robot is a particle that moves according to the sum of defined forces applied to its mass. These potential forces are computed by defining a set of functions as detailed in [27,28]. Those functions can be divided into two main categories: attractive & repulsive. The role of attractive functions is to guide the robot to the goal, while the repulsive ones pushes it away from obstacles. The resultant force is what leads the robot through a collision-free motion. Other types of functions exist

(detailed in [2]). This method is easy and efficient, because at any moment the motion of the robot is determined by the potential field at its location. Another benefit is its extensibility when adding a new obstacle, which is simply incorporated by summing its field to the previous ones. However its main disadvantage is getting trapped in local minima. A way to escape such local minima is using "Randomized Potential Field", where random motion is generated whenever stuck, to guide the robot outside the dead region.

5.2. Cell decomposition

Cell decomposition methods represent the free space by a collection of non-overlapping cells, where the cells result from a non-uniform process of dividing the C_{free} given the obstacle information available. Different decomposition techniques have been proposed: *The trapezoidal decomposition* being the most popular exact cell decomposition, the environment and the obstacles which are represented as polygons are divided into trapezoidal and triangular cells by simply drawing parallel line segments from each vertex of each polygon in the configuration space to the exterior boundary of the space. A more general decomposition class is termed Morse Decomposition like *brushfire* and *wavefront* [2]. Another variation is the *visibility-based decomposition* that uses changes in line-of-sight information to define cells. For instance, moving from one cell to another corresponds to a change in visibility (e.g. obstacle or target appears or disappears). Approximate Cell Decomposition on the other hand, uses a recursive method to continue subdividing the cells until either: (1) each cell lies either completely in C_{free} or in $C_{obstacle}$. (2) and an arbitrary limit resolution is reached. Examples of such technique are quadtree, octree, or 2n-tree [2].

After performing the decomposition, a path is computed in two steps. First, the planner determines the cells that contain the start and goal, and then it searches for a path within the adjacency graph. In this graph a node corresponds to a cell and an edge connects adjacent cells. It encodes the adjacency relationships of the cells, and thus serves as a roadmap of the free space. Therefore, planning can be achieved by incrementally expanding this graph.

Cell decomposition is conceptually easy, however it does not scale due to its complex implementation and poor time efficiency. For example, for the trapezoidal decomposition, there is a complexity residing in constructing the decomposition itself, with overall time estimation of $O(n^2)$ and storage of $O(n)$, where n is the number of nodes in the graph.

5.3. Roadmaps

Considered as a complete general method that applies to spaces of any dimension [29]. Instead of focusing on solving (start, goal) problems, they are concerned with capturing the connectivity of the whole free space. As a consequence this map contains information about all feasible routes in the environment, hence it can be queried for multiple (start, goal) pairs. A roadmap (R) is defined as a subset of the configuration space that results from the union of one-dimensional curves, in which any (start, goal) contained in C_{free} can be connected by a path that is:

(1) Accessible: there is a free path in R from start to another configuration q'_{start}.
(2) Departable: there is a free path in R from some q'_{goal} to the goal.
(3) Connected: there is a free path in R that connects q'_{start} to q'_{goal}.

There are various types of roadmaps: the *visibility graph* [30] assumes a 2D configuration space with polygonal obstacles, the set of the visibility nodes v_i is composed of the (start, goal) pair and all the vertices of the obstacles. Its edges, e_{ij}, are straight-line segments that connect any possible combination of two nodes v_i and v_j, without colliding with the obstacles. Once the visibility graph is fully defined, the solution path can be obtained by implementing any graph search method (e.g. Dijkstra's [31]). It suffers from a clearance issue, as usually the solution found touches obstacles at the vertices or even edges. This drawback is addressed in the *Voronoi diagrams* [32], where the diagram is defined for sets of points equidistant to at least two obstacles. This means that they are the mid-point between two obstacles to be avoided, and the edges correspond to the possible channels that maximize the distance to the obstacles [2]. Other variations exist such as the *freeway* net [33], and the *silhouette* [2].

5.4. Sampling-based

Each of the previous methods relies on an explicit representation of the geometry of the free space. As a consequence, with the growth of the dimensionality of the configuration space, they might become impractical. In contrast, the sampling-based methods create a sampled (discrete) representation of the configuration space that captures the connectivity of C_{free}. They exploit the fact that, while explicitly building C_{free} is expensive, checking a given configuration for collisions is quicker. To build such a discrete representation, they follow a two-stage technique: "sample" & "connect". In the first, they generate random sample routes q_{ri} that are checked for collision. Secondly, they interconnect the random and collision-free configurations, thus establishing different routes (paths) to solve queries. Their basic version is focused on finding any feasible path rather than minimizing the solution cost. However, this methodology may often result in solutions of unpredictable length with superfluous movements, motions that graze the obstacles and jerky trajectories that may be potentially hard for the manipulator to follow. In order to compensate for this quality loss, various post-processing techniques (e.g. shortcutting, smoothing) became essential. Furthermore, they may fail in cluttered environments, and narrow passages (i.e. areas with less samples). Most of the sampling-based approaches provide no optimality guarantees of the solution and only probabilistic-completeness (the probability of finding the solution increases with the sampling density)

Two main categories exist:

- *Multi-query planners:* similar to the Roadmap technique, as they are able to answer multiple queries through building a Probabilistic Roadmap *PRM* [34].
- *Single-query planners* solve a query specified by a (start, goal) pair by growing a tree of configurations between them.

Because there is a wide variety of sampling-based algorithms, we will summarize only those to be compared in Section 8, while a more exhaustive list of planners (including those taking into account planning with constraints) and their implementation guidelines can be found in the *OMPL* library documentation [35], a state-of-the-art library for sampling-based motion planning.

- Rapidly-exploring Random Trees *RRT* was first presented in [36] and is now the most widely used single-query method, along with its bi-directional version *RRTConnect* [37] which grows two RRT trees (one at the start, one at the goal) in an attempt to connect both and find a path.
- Expansive-Space Trees *EST* [38]: during the construction phase the algorithm selects the node q to be connected in a way that prioritizes less-explored regions (e.g. narrow passages), and then it randomly samples a configuration q_{rand} around q, thus attempting to overcome one of the

weaknesses of the basic *RRT*. In order to estimate less explored regions, they apply a linear projection to reduce the high-dimensional configuration space. It has a bi-directional (start & goal), lazy collision-checking (i.e. collision check along connections in the tree deferred until they are needed not during tree expansion) version: the Single-query Bi-directional Lazy collision checking planner *SBL* [39].
- Search Tree with Resolution Independent Density Estimation *STRIDE*: inspired by *EST*, instead of linear projection, they rely on a tree data structure to search for nearest-neighbor information and estimate the sampling density needed in a given region. It is useful for high-dimensional systems where the free space cannot easily be captured with a low-dimensional linear projection [40].

Since the original sampling-based techniques trade optimality for speed, they do not provide any guarantee of the produced path optimality, for a given cost function. A set of optimal planners were introduced like *RRT** and *PRM**, for single and multi-query planners [41] in order to improve the quality of the produced paths in terms of different criteria (e.g path length). The improvement happens in the "connection" step of the algorithm, where new configurations are connected to the closest and best configuration given the cost function.

5.5. Search-based

Also called "Grid-based", they use a discrete representation of the configuration-space through a uniformly discretized grid. Then the planning problem is formulated as a graph-based search to find an optimal path between a given (start, goal) pair, with collision-checking occurring only when expanding nodes of the graph. Assuming S denotes the finite set of states of the domain, $c(s, s')$ denotes the cost of the edge between states s and s' ($s, s' \in S$); if there is no such edge, then $c(s, s') = \infty$. $Succ(s) := \{s' \in S \mid c(s, s') \neq \infty$ denotes the set of all successors of s. $g(s)$ denotes the current best path cost from s_{start} to s, and $h(s)$ denotes the heuristic for state s, which is an estimate of the best path cost from s to s_{goal}. A priority queue (commonly known as OPEN list) holds the states eligible for expansion, ordered by their estimated path cost. A CLOSED set holds those states that have already been expanded. A typical search-based algorithm runs first by picking the minimum estimated-cost node from the OPEN queue, and expands this node by generating its successors. For each successor $s' \in Succ(s)$, if a less costly path to s' is found through this new expansion, its $g(s')$ is lowered and, if it has not been expanded yet, s' is added to the OPEN queue.

Heuristic-based techniques typically come with strong theoretical guarantees on completeness and optimality or at least bounds on sub-optimality [42]. In addition, the generality of heuristic searches allows incorporation of complex cost functions and constraints to easily represent arbitrarily shaped obstacles with grid-like data structures [43]. They also provide consistency in the solutions, given the deterministic nature of the grid as opposed to random samples in sampling-based techniques.

Different variations of the basic technique have been developed:

- *Static Planners* like simple *A**, useful for simple low dimensional problems.
- *Anytime Planners* such as *ARA** (Anytime Repairing A*) [44], they aim to find the best plan they can within the amount of time provided. They operate by quickly finding an approximate and possibly highly sub-optimal plan first, and then keep improving it until no more time is available. The time spent on improvement is called *Repair Time*. In addition to being able to meet time constraints, many such algorithms make it possible to interleave planning and execution: while the agent executes its current plan, the planner keeps improving it.
- *Dynamic Planners* are able to adapt with changing environments, due to their re-planning capabilities, an example is *D** *Lite* [45].
- *Hybrid Planners* like the anytime re-planning algorithms (e.g. AD* [46,47], and *R**[48]) combine a range of features to cover a wider range of problems.

Search-based methods have long been known to struggle when it comes to high DoFs systems. They become inefficient due to the increase of the graph size, which is reflected in the search time (i.e. high planning time).

The work presented in [26] shifted the paradigm by introducing the notion of lattice-based graphs (originally introduced in [49]) to overcome this inefficiency. Lattice-based graphs use what is called lattice states, i.e. states that discretize the possible robot actions into *Motion Primitives*. This notion removes the bounds on using search-based planners only with 4/8-connectivity, and allows robot actions to not be tied to the grid and are now free to cover more than one grid cell. For this new paradigm, a search-based algorithm is labeled in terms of three aspects: *search algorithm*, *heuristic*, and *graph representation*. The design of these core components determines several important factors including the planning time, memory footprint, smoothness of the trajectory and kinematic/dynamic feasibility of the generated path. Our study focuses on two of these search techniques: anytime A* (*ARA**) and its lazy version (*LARA**) where the states are checked for collision only when extracting the final solution.

In addition, two graph representations are used:

- *Manipulation Lattice* (indicated shortly as CS or configuration-space lattice), where the Motion-Primitives (MP) are defined in the configuration space of the system (i.e. each MP modifies one or more DoF). Two types of MPs can be defined: long and short, where the short ones are usually used around the goal regions (a configured threshold allows use of one or the other). Usually these short MPs imply relatively longer motions to jump straight to the goal, while the long ones help in exploring the space.
- *Workspace Lattice* (indicated shortly as WS) where the Motion-Primitives are defined in a 3-D workspace instead $(x, y, z, \phi, \psi, \theta)$. In order to produce a final kinematically-valid/collision-free trajectory, periodic inverse-kinematic calls are needed during the graph search/expansion process.

The trade-off respectively is speed versus coverage of a wider variety of domains. If the environment is known in advance then designing the MPs for the manipulation lattice is easy and one would profit from a fast-performing planner. However, the same MPs set would probably fail or at least result in a poorer performance for a completely different environment. Whereas for the workspace lattice, the fact of using the end-effector enables covering various environments with higher maneuverability, at the expense of frequent inverse-kinematic calls which increase the planning time.

And finally, we focus on the following three heuristics:

- *Euclidean:* computes the euclidean distance to the goal either in the configuration space or workspace depending on the graph representation used. They are goal-directed, which helps in computing short paths, but not aware of the environment and information on obstacles. This may lead to colliding paths and frequent failure in complex environments.

- *Breadth-First Search (BFS):* the idea is to keep track of the obstacles information in the environment through a 3-D grid with cells marked free/occupied. This grid has the x, y, z end-effector goal as the origin cell for the BFS. An inflation radius is specified for the end-effector – given its dimensions – and it is used to inflate the grid cells. Those cells lying in the area within this radius are marked as occupied, helping to guide the end-effector in cluttered regions. Finally, in a breadth-first manner, the heuristic computes distances for all the cells in the constructed 3-D grid that correspond to the lengths of paths for the end-effector to reach its goal while avoiding obstacles
- *Multi-frame Breadth-First (MFBFS):* provides the possibility of running multiple parallel BFS in order to fasten and improve the search process. Each of the new BFS has its own reference frame (instead of the usual end-effector) frame (e.g. using two reference points on the arm end-effector given its shape to help manoeuvring around obstacles). It is a domain-dependent heuristic, and in common cases might not demonstrate any improvements over single BFS.

5.6. Optimization-based

They formulate the motion planning problem as on optimizing problem to be solved by defining one or more cost function(s) to be optimized through an iterative process [50–53].

Common cost functions include *path smoothness*, *obstacle avoidance*, and *joint limits* as essentials. Although, others can be added like system dynamics, stability (for humanoids), manipulability (for arms), any other task specific constraints, or a combination of them.

The **C**ovariant **H**amilton **O**ptimization **M**otion **P**lanning (*CHOMP*) [51] works by creating a naive initial trajectory from the start position to the goal and then, uses a modified version of gradient descent for the cost function to optimize the trajectory. Some advantages over the sampling-based approaches include the ability to optimize trajectories for smoothness and to keep a pre-defined minimum distance to obstacles when possible. On the other hand, gradient-based methods might suffer from getting stuck in local minima. As a consequence, [54] introduced the use of Hamilton Monte Carlo algorithm to apply perturbations to the path and restart the optimization process to deal with this issue. However, this approach introduces randomization in *CHOMP* and reduces its deterministic nature.

Similarly, **S**tochastc **T**rajectory **O**ptimization **M**otion **P**lanning (*STOMP*) [53] offers the capability to optimize general cost functions providing good performance for manipulation problems. It relies on generating noisy trajectories to explore the space around an initial (even unfeasible) one. Those noisy trajectories are later combined to generate an updated lower-cost solution. An advantage of the optimization technique used by *STOMP* is that no gradient information is required, thus allowing optimizing generic cost functions with no derivatives computation. In addition, its stochastic nature is able to overcome the local minima that gradient-based methods usually suffer from.

5.7. Hybrid techniques

Some methodologies have been reported which do not specifically fall within one category or another, but adopt hybrid mechanisms:

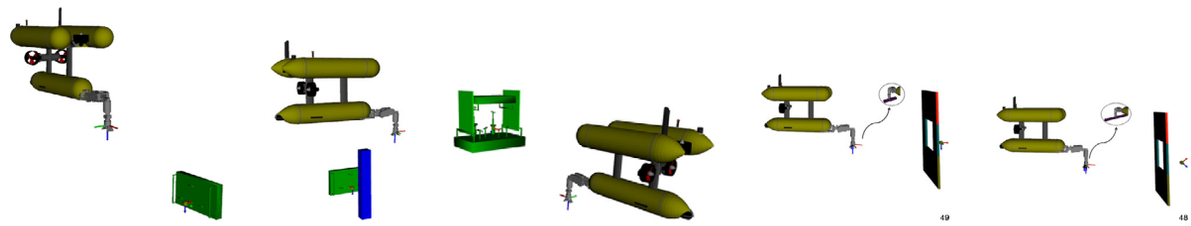
- *Dynamic Roadmaps: [*55] based on the sampling-based roadmap concept, they maintain an online dynamic version of the roadmap in order to cope with dynamic changing environments.
- *Real-time Planners: [*56] Focusing on real-time motion planning with dynamic obstacles, these authors suggested two approaches. One is to modify a leading motion planning algorithm to make it real-time, a solution called R*. The other is to modify a leading real-time algorithm to be better suited for motion planning. They came out with what they called Partitioned Learning Real-Time A^* (*PLRTA**).
- Elastic bands: a technique based on integrating motion planning and control to achieve real-time path generation, with dynamic obstacle avoidance and re-planning capabilities. In addition, the use of task-based control, enables the possibility of embedding other task specific requirements (i.e. stability behavior for a humanoid, or orientation constraint for tray transporting). Different variations can be found in [57–60]

6. Benchmarks

In this section, the various benchmarks (see Fig. 1) as well as the philosophy behind choosing them is explained, to provide insight on how to benchmark a given motion planning problem.

The five benchmarks have been carefully created to demonstrate when randomness is a strength, and when the deterministic approach is the way to go. For each scenario an end-effector goal was chosen by specifying its position x, y, z and orientation ϕ, θ, ψ. The idea behind choosing the start posture in all scenarios is to be around the manipulation region, not too close to over simplify the scenario and not too far so that the problem would be more concerned about navigation instead of manipulation. The arm configuration was set to zeros in all benchmarks, and the depth of the AUV was set in the same range of the manipulation object. The main difference between the scenarios was either in the AUV orientation or side positioning. For instance in BM#1 & BM#2, we chose the orientation to be perpendicular to the valve and on the left side (as opposed to the blue bar obstacle in BM#2), to add a challenge to the scenario and see how the different planners will deal with the AUV orientation change. For the same reason, in BM#3, the AUV was initially positioned on the right side with respect to the plug to be picked up. In contrast, for BM#4 & BM#5, the AUV was centered in front of the window, to avoid more complexities to an already challenging scenario. Different initial configurations would lead to different results, but we believe it should be tailored to reflect the purpose of the benchmarking. In addition, they should not be neither over-simplified (e.g. positioned straight in front of the object to be manipulated) nor over-complicated.

A subset of the planners explained in Section 5 were chosen for comparison. Others have been omitted, either because they are too theoretical (e.g. cannot scale to high-dimensional systems) or because there is no open-source ROS-based implementation of the algorithm. Having an open-source ROS-based implementation is important to allow fair comparison using the latest ROS standard tools. Three categories are covered: (1) sampling-based, (2) optimization-based, and (3) search-based planners. For each category, our choice aims to cover the basic approach in addition to the various features provided by each planning family (e.g. lazy collision checking for both sampling-based and search-based), with a total of 17 planners (listed in Table 1). For the search-based planners, the different combinations of the different core components: search, graph & heuristic (as explained in Section 5) were compared. In order to reduce the number of results presented, and because for some scenarios a given combination was not showing improvement; we omitted some search-based combinations and ended up dividing the benchmarks into two sets: *Set(1)* [BM#1,BM#2,BM#3] & *Set(2)* [BM#4,BM#5]. The combination of



(a) BM#1: Simple Valve Turn.

(b) BM#2:Valve Turn With Obstacle.

(c) BM#3: Cluttered Environment.

(d) BM#4: 50 cm Stick Manipulation.

(e) BM#5: 1 m Stick Manipulation.

Fig. 1. Benchmarks with the I-AUV Start configuration & the goal pose ($x, y, z \rightarrow$ yellow sphere, $\phi, \theta, \psi \rightarrow$ RGB Axis).

Table 1
Planners choice & benchmarks coverage.

Categories	Planners	Benchmarks
Sampling-based	RRTConnect	All
	SBL	All
	STRIDE	All
	RRT*	All
	PRM*	All
Optimization-based	STOMP	All
	CHOMP	All
Search-based	ARA*.BFS.CS	All
	ARA*.MFBFS.CS	BM#4 & BM#5
	ARA*.EUCLID.CS	All
	ARA*.BFS.WS	All
	ARA*.MFBFS.WS	BM#4& BM#5
	ARA*.EUCLID.WS	All
	LARA*.BFS.CS	BM#1,BM#2,BM#3
	LARA*.EUCLID.CS	BM#1,BM#2,BM#3
	LARA*.BFS.WS	BM#1,BM#2,BM#3
	LARA*.EUCLID.WS	BM#1,BM#2,BM#3

(*ARA**) search, *CS* & *WS* graph, and heuristics (*EUCLID* & *BFS*) was tested for both sets, while the lazy-collision checker was tested for *Set(1)* with a total of 15 planners, and the multi-frame BFS was tested for *Set(2)* with a total of 13 planners. This division allowed us to minimize the amount of data presented in this work while still showing the reader the existence of different features and their possible usage.

The benchmarks have been inspired by the work done so far in terrestrial robotics, with some adaptations to fit the underwater environment. Nevertheless, we consider that the results presented are applicable to other robotics domains including ground as well as aerial robotics.

6.1. Simple "table-top" like manipulation

One of the most common underwater intervention tasks, consists of turning a valve on an offshore sub-sea panel like those used in the offshore industry. It is considered of interest for the Oil&Gas industry, which is why it has been used as a representative task for intervention. Moreover, its simplicity allows comparison of the performance of the motion planners under relaxed conditions. This task has resemblances to the operation in table-top or what is commonly known as pick & place tasks for ground robots. Although in this case the "table" is rather vertical. Two variations have been implemented:

- BM#1: Obstacle-Free Valve Turning The benchmark is shown in Fig. 1(a). The unique obstacle to be avoided is the sub-sea panel itself.
- BM#2: Valve Turning in presence of Static Obstacles The benchmark is shown in Fig. 1(b). In this case, a rectangular bar has been added to the scene, located at the right side of the valve to be turned. It makes the intervention more challenging.

In both cases, the robot starts in a pose parallel to the panel and has the goal of bringing the end-effector to the middle lower T-shaped valve to turn it.

6.2. Manipulation in cluttered environment

A significant challenge for mobile manipulation arises when the manipulation target is located within an environment cluttered with obstacles. This type of problems can be extended to dynamic environments with moving obstacles or when there is uncertainty in the knowledge of the environment, which is not covered in the present work. In such circumstances, a high accuracy is required to manoeuvre around the obstacles to reach the object to manipulate. Moreover, it might be required to pass through narrow passages, adding complexity to the problem. For ground robots, this has been usually represented through manipulation of objects within a shelf cluttered with obstacles.

BM#3 benchmark shown in Fig. 1(c), represents the underwater version of a cluttered environment. The task consists of unplugging a hot-stab connector. The same panel of BM#1&2 has been used, but located horizontally and equipped with a set of hot-stab connectors and valves. A secondary horizontal panel and 2 vertical side panels complemented the scene. Their role is to constrain the environment and to force the generated plan to go through the clutter instead of reaching it from above. The target end-effector attitude was selected to be on the plane perpendicular to the connector ($\psi = 1, \theta = -0.2, \phi = 2.4$), assuming the connector can be moved up for unplugging. .

6.3. Manipulation through narrow passage

The Narrow-Passage is one of the most challenging problems that have been addressed in motion planning. For this reason, we give it more attention, and we carefully reviewed the literature to find the best representation of this problem. The selected scenario for this benchmark is shown in Fig. 1(d) & 1(e)), where a stick whose size can be chosen within specified limits, has to be manipulated to pass across a window of size 60 $\times$ 60 cm, simulating an object manoeuvre through a narrow passage. This benchmark is inspired by the work presented in [61]. This scenario is quite common for ground robots where an object is to be moved from one side to the other through a narrow shelf, or for robots working in rescue where narrow tunnels/holes might exist due to the nature of the disaster through which they need to operate. For the underwater domain, this could be similar to a structure inspection task, where a camera attached to a bar needs to be placed into different sections of a complex sea structure where

there are narrow passages. Four test cases with different stick sizes (25, 50, 75, 100 cm) have been tested, although only the second and the last are reported (BM#4 & BM#5), as they represent interesting findings.

In BM#4, the 50 cm stick was used, and the goal position was carefully chosen (20 cm from the window center in the positive X-direction, taking into account the wall thickness $T = 4$ cm) to challenge the planning problem and at the same time ensure it is reachable with $\psi = 0, \theta = 0, \phi = 0$ orientation.

While in BM#5, the stick length was increased to 1 m in order to make it longer than both the height and the width of the window. In this case, the goal was modified to ensure its feasibility with the large stick. An additional 20 cm with respect to BM#4, were added in the positive X. Additionally, the orientation was modified to $\psi = 0, \theta = 0.7, \phi = 0$. It is worth noting that at this X-position, the $\psi = 0, \theta = 0, \phi = 0$ was failing for the heuristic-based planners, as the heuristic guides the search though the window, and it was impossible to pass the large stick through the window with 0 orientation.

7. System, tools & methodology

7.1. Underwater vehicle manipulation system

The system used in the tests is an underwater mobile manipulator, where a 4-DoF ECA/CSIP Micro-Arm arm is attached to Girona500 vehicle [62]. The vehicle has 4 controllable DoF x, y, z, ϕ, being passively stable in roll and pitch (ψ, θ), resulting in an 8-DoF system. All the DoFs were modeled in *MoveIt!* as a single planning group. This UVMS has been used for several intervention tasks, but specifically with *MoveIt!* in [63].

7.2. Tools & methodology

The results conducted in this survey are based on the benchmark plugin provided by *MoveIt!* [12]. It is an ROS-based [19] plugin based on the benchmark plugin provided by the Open Motion Planning Library for sampling-based planners [20,35]. The plug-in uses the warehouse concept of *MoveIt!*, where a database is used to save different planning scenes (i.e. test scenarios), the starting and the goal states. From the plug-in side, a connection to this database is established and the scenes are loaded together with the corresponding start-goal queries. The (start-goal) query(ies), the scene to be loaded, the number of runs, and the planners to be tested are defined in a configuration file (.yaml) as explained in [64]. A generic motion planning plug-in implemented in [65] was used in order to load planners from different categories (*OMPL*, *SMPL*, *STOMP*, & *CHOMP*), each with its specific configuration.

Some modifications had to be done to the benchmark plug-in to support the test cases as designed (e.g. support for: multiple workspace goals, updating world objects status, collision & non-collision objects, adding attached objects, metrics logging tool), and the corresponding modified code can be found in [66]. Each benchmark was repeated M times ($M = 20$) for each planner, with a maximum allowed-time of 20 s. The goal query was defined in the 3-D workspace $(x, y, z, \psi, \theta, \phi)$, with a tolerance of 0.01 for each.

The nature of the optimization-based planners does not allow providing goals in the end-effector space. As a consequence, an automated pre-call to an inverse kinematic service was added to convert the goal to the configuration space. This is considered a limitation, since only one of the redundant configurations leading to the considered end effector goal-pose is taken into account. In our specific case, with limited redundancy (only 2 DoF), we believe that it does not dramatically affect the comparison result.

Parameter Tuning: each planner, especially search-based ones, has dependency on the setting/tuning of some parameters. Here is a brief explanation of those parameters:

- Search-based planners: The grid resolution remains a critical parameter, as well as the motion primitives set chosen (as explained earlier). The *Repair Time* was set in all tests to 1 s. In addition, the cost of the edges and nodes of the graph was uniform.
- Sampling-based planners: a considerable advantage of this category is the possibility to use the default parameters provided by the OMPL with a wide variety of scenarios, and still perform pretty well, as will be shown. It is possible, that further tuning per test-case would improve the outcome, but we choose to leave the default tuning and consider it an advantage over other planners' implementations.
- Optimization-based planners: similar to sampling-based, no specific parameter tuning was needed, except for STOMP in BM#5 (this will be explained later).

Metrics: In the result sections, a set of plots per benchmark are presented. Five metrics are compared, four of them are shown as box plots, in order to capture the variation along the different runs of the same planner for the same test case. Details of the computation of each metric are given below:

(1) **Planning time:** represents the time needed to find a solution, or the first solution for optimal sampling-based planners (i.e. *RRT** & *PRM**). It is separately computed by each planner given its own implementation, and it includes post-processing time (e.g. smoothing or short-cutting).

(2) **Path length:** computed as the *Manhattan Distance* between consecutive way points. Reported in Eq. (1) where N is the total number of way points in the path.

$$L = \sum_{k=2}^{N} d_1(\mathrm{q}_k - \mathrm{q}_{k-1}) \; ; \; \mathrm{q} = [x\, y\, z\, \psi\, q_1\, q_2\, q_3\, q_4] \tag{1}$$

(3) **Path clearance:** computed as in Eq. (2), the higher the value the safer the path. A *MoveIt!* provided routine, based on an octomap representation of the environment [67], is used to compute the minimum distance to the world objects for a given robot configuration (q). Usually the clearance is computed over the whole trajectory, but since our benchmarks are concerned with planning for manipulation, and the obstacles are found only in the manipulation region; we noticed that considering the full path for clearance computation can lead to unfair comparison. For example for some sampling-based planners, and in the narrow passage benchmarks, the resulting path was expanding more in free areas (see Fig. 15) before getting to the target pose, which might lead to better clearance even though it is not a matter of keeping a reasonable distance to obstacles, but rather a sampling issue. As a consequence, we decided to apply a threshold to the trajectory, considering only the portion of it lying in the neighborhood of the goal region defined with a threshold radius R (In our case, in order to suit our benchmarks, we chose R as $dist/2$ with $dist$ being the start to goal euclidean distance).

$$C = \frac{\sum_{k=1}^{N} d_{min}(\mathrm{q}_k)}{N} \quad \text{where } N = \#\{\mathrm{q}_k \,/\, d(\mathrm{q}_k, \mathrm{q}_{goal}) < R\}, \tag{2}$$

$(\mathrm{q}_k, \mathrm{q}_{goal})$ are the kth and goal configurations

(4) **Path smoothness:** a path is viewed as a sequence of segments, and the angle between consecutive segments is computed. The smoothness (as computed in [35]) is considered

$$S = \sum_{k=2}^{N-1} \frac{[2(\pi - \arccos(\frac{A^2 + B^2 - C^2}{2AB}))]^2}{A + B} \quad (3)$$

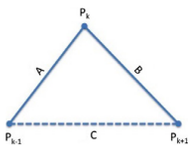


Fig. 2. Path segments for smoothness computation.

as the complementary to this angle (Eq. (3)/Fig. 2). The smaller the value the smoother the path.

(5) **Success rate:** displays the success vs. failure rates for each planner. A legend of three colors (green, red and orange) is used. Green reports the successful (i.e solution found) runs percentage. Red represents the percentage of non-collision free solutions found. Finally, the orange represents the runs where the algorithms was not able to find a solution within the given time.

To complement the quantitative assessment, a qualitative evaluation is also presented in the last plot **Distance to objects color map:**

This plot (e.g. Fig. 4(e)) captures the evolution of the trajectory with respect to the obstacles. The $y-axis$ represents the length of the trajectory in number of way-points, the bottom of the bar being the beginning of the trajectory and the top its end. The bar progress is displayed as a colored heat bar, where the color of a box reflects how distant/close is the robot (its closest joint) to the obstacles. This graph visually assesses the behavior of the planner with respect to obstacles, and also allows perceiving the change of planner behavior with respect to environment complexity. For example, transitions from blue to orange and back to blue illustrate unnatural behaviors where the system does not evolve in a smooth way towards the goal. It is worth noting that, as a difference with respect to the rest of the plots, this plot illustrates the results of only one of the M runs, in particular the one corresponding to the longest path. All non-collision free paths were ignored. The range of each color ($far \longrightarrow collision$) is not fixed but computed based on the minimum and maximum distance found in the selected runs of all the planners for a specific scenario.

8. Results & analysis

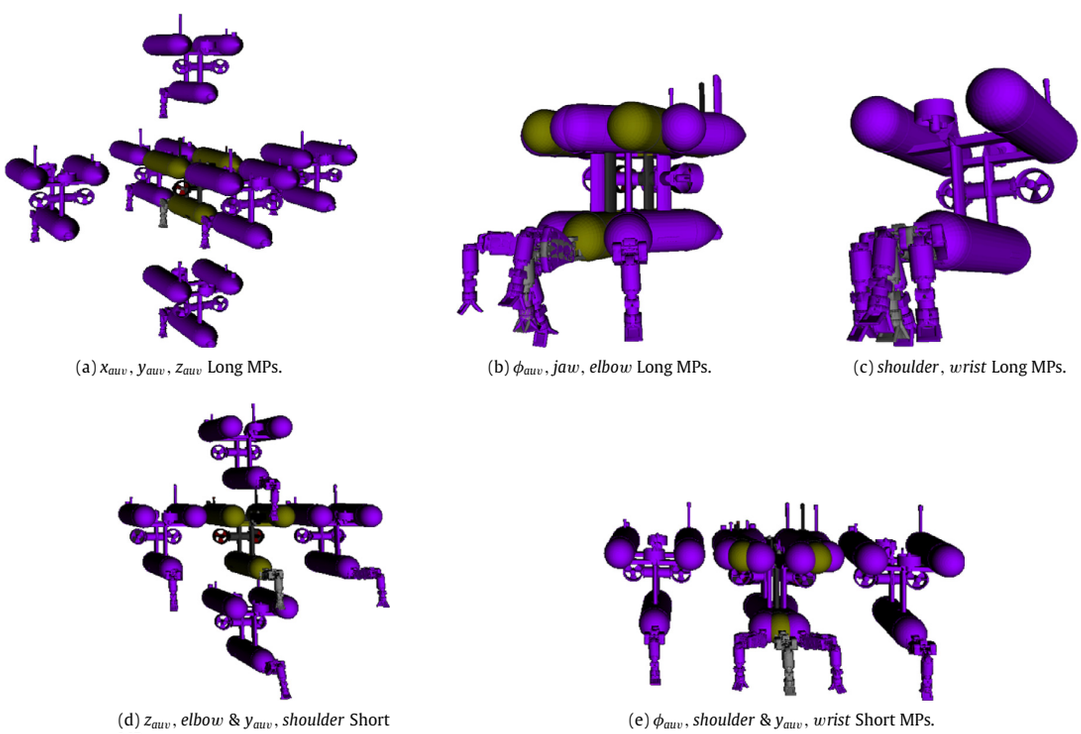
In this section we present analysis of the previously explained metrics, applied to the benchmarks motion planners combinations as in Table 1). It is worth noting that some planners showed a 100% failure rate for some benchmarks, and for such cases they have been omitted from the plots to avoid confusion (e.g. CHOMP for BM#2).

8.1. "Valve Turning" with & without obstacles

The motion primitives used by the manipulation lattice search-based planners, for these two benchmarks and the cluttered test, are shown in Fig. 3, with a total of 26 long motion primitives and 10 short ones.

The results for BM#1 and BM#2 are reported in Figs. 4 and 5

Planning Time: [Figs. 4(a) and 5(a)] In both test cases, the sampling-based methods outperformed the other planners, while the highest time is demonstrated by *STOMP*. It is worth noting that for both benchmarks, both *ARA***.BFS*.*WS* and *LARA***.BFS*.*WS*



(a) x_{auv}, y_{auv}, z_{auv} Long MPs.

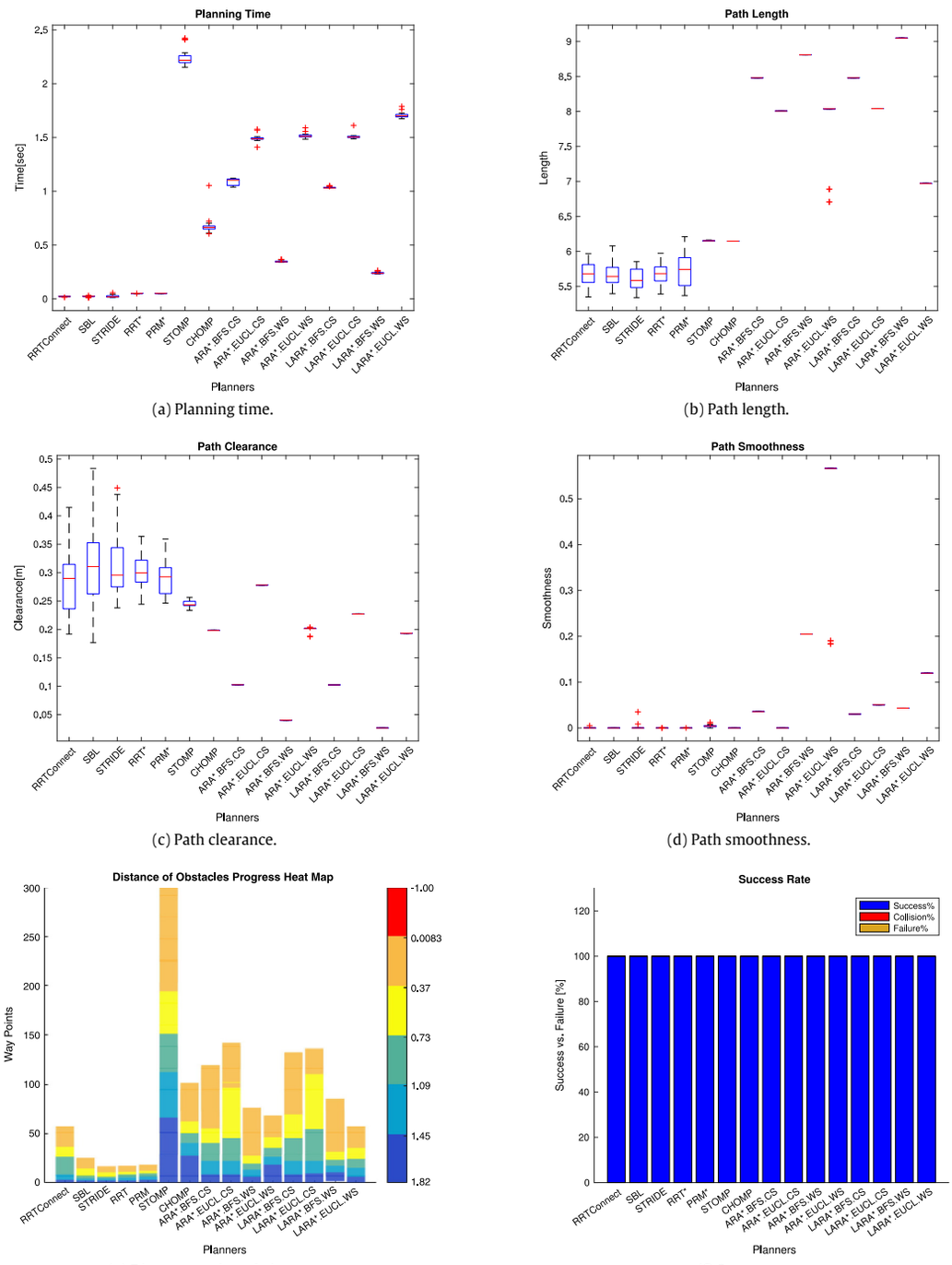
(b) ϕ_{auv}, jaw, $elbow$ Long MPs.

(c) $shoulder$, $wrist$ Long MPs.

(d) z_{auv}, $elbow$ & y_{auv}, $shoulder$ Short MPs.

(e) ϕ_{auv}, $shoulder$ & y_{auv}, $wrist$ Short MPs.

Fig. 3. Manipulation lattice motion primitives for BM#1, BM#2, BM#3. The yellow is the current I-AUV configuration & the purple are all possible motions from the current. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) Planning time.

(b) Path length.

(c) Path clearance.

(d) Path smoothness.

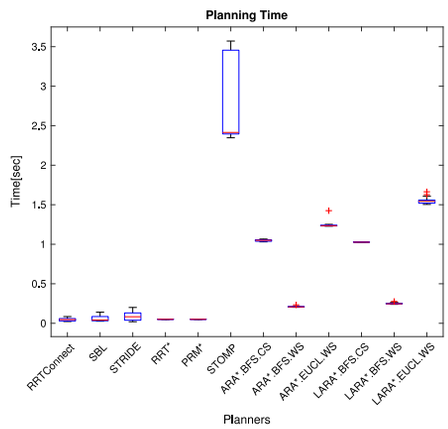
(e) Distance to obstacle heatmap.

(f) Success rate.

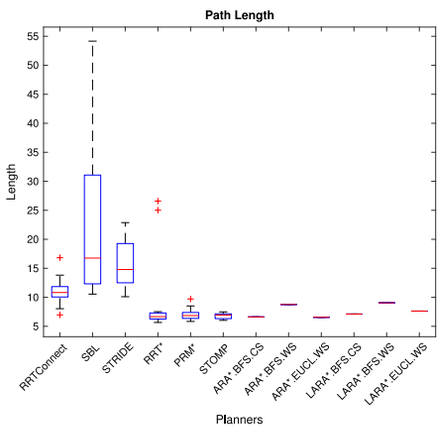
Fig. 4. Benchmark results for the simple valve turning.

search-based methods provide times pretty close to that of the sampling-based methods. In addition, the *BFS* versions systematically surpass their *EUCLID* counterparts due to the nature of the heuristic. On the other hand, from the results it cannot be concluded which of the graph representations (WS or CS) behaves better. A variability along the runs appears in the presence of obstacles (BM#2), for the sampling-based, and was significantly noted for *STOMP* where planning time increases by 1 s for the same benchmark. In addition, the lazy *EUCLID.WS* experienced longer planning time compared to the non-lazy version.

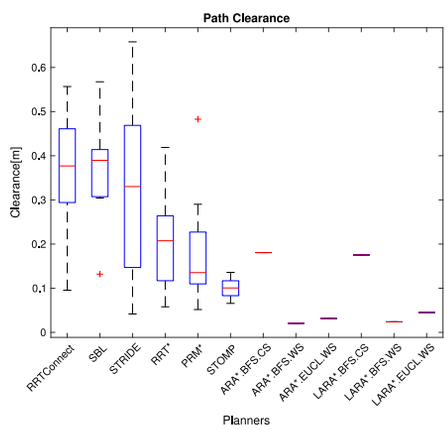
Path Length: [Figs. 4(b) and 5(b)] A major degradation was encountered moving from BM#1 to BM#2 for the sampling-based planners. The average range increased, as well as the variability along the runs (specially for *SBL*), also outliers appeared. While



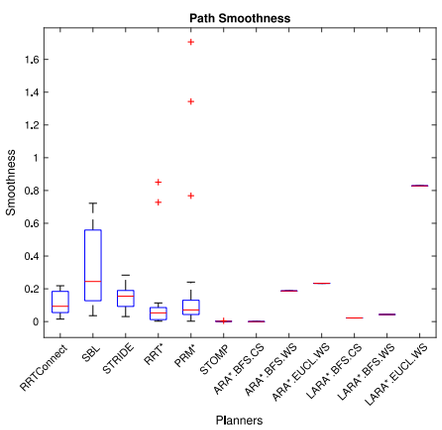
(a) Planning time.



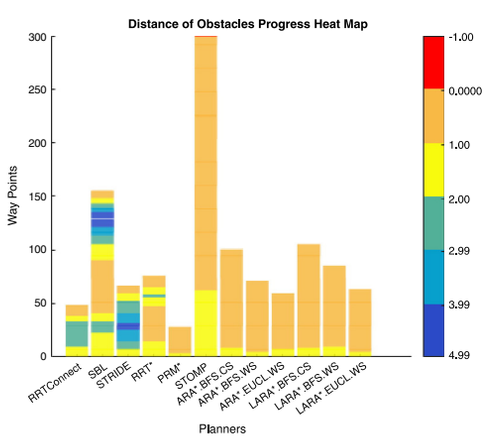
(b) Path length.



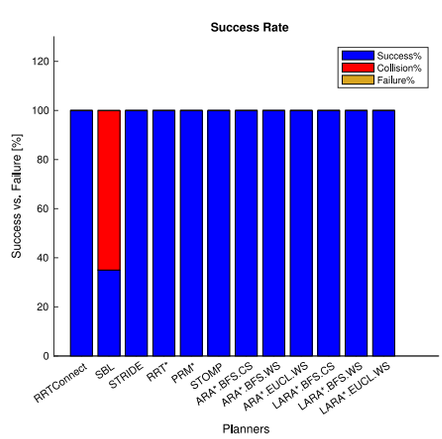
(c) Path clearance.



(d) Path smoothness.



(e) Distance to obstacle heatmap.



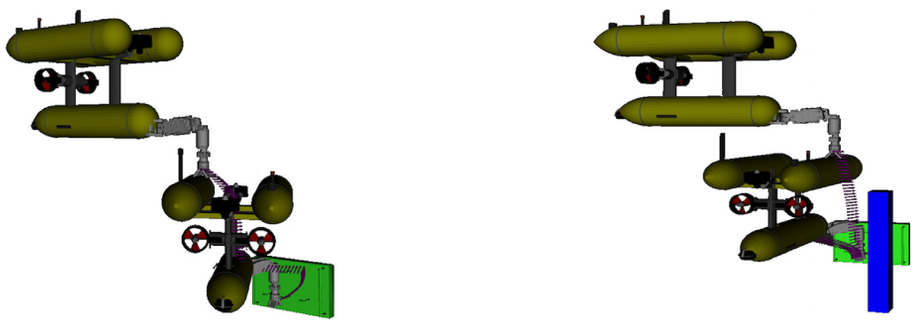
(f) Success rate.

Fig. 5. Benchmark results for valve turning with bar obstacle.

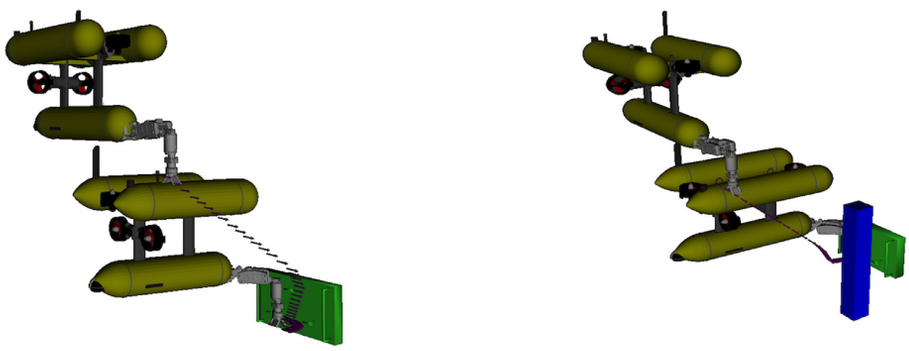
they outperformed the rest of the planners in the simple case. The path length for the optimization-based and the search-based planners was not highly affected by the bar addition, and they were able to circumnavigate the obstacle as shown in the different trajectories in Figs. 6 and 7.

Path Clearance: [Figs. 4(c) and 5(c)] The search-based methods exhibit worse clearance even in the simple BM#1. In contrast, sampling-based and optimization-based clearance decreased from BM#1 to BM#2, and the data standard deviation increased.

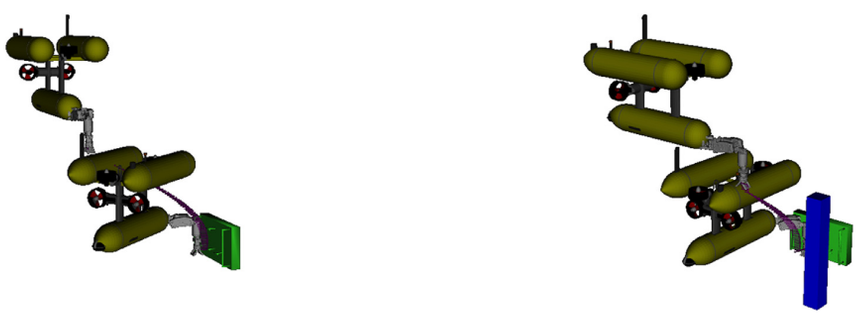
Path Smoothness: [Figs. 4(d) and 5(d)] *ARA*.EUCL.WS* showed higher smoothness for both benchmarks, at the same time the other methods' smoothness remain low. Trajectories obtained in BM#2 are slightly less smooth, which is expected due to the bar obstacle to be avoided.



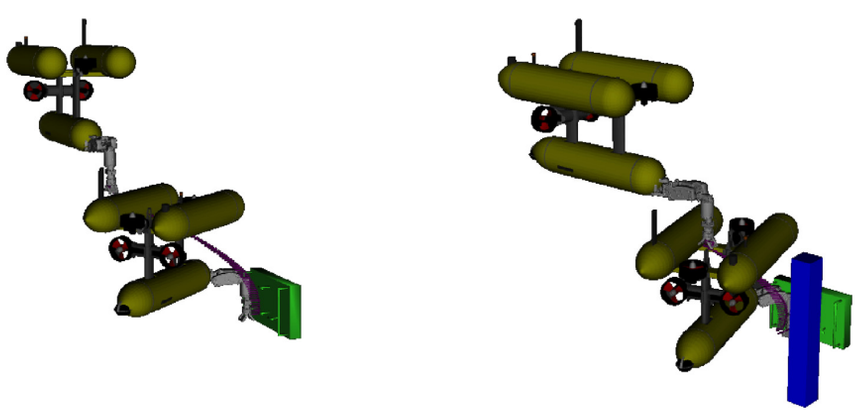
(a) *ARA* — BFS — Manipulation.*



(b) *ARA* — BFS — Workspace.*

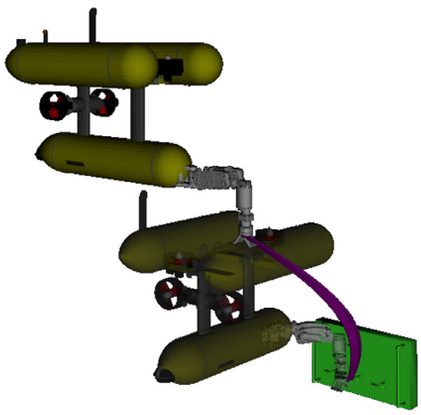


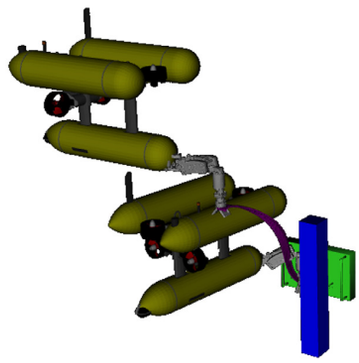
(c) *ARA* — Euclidean — Workspace.*



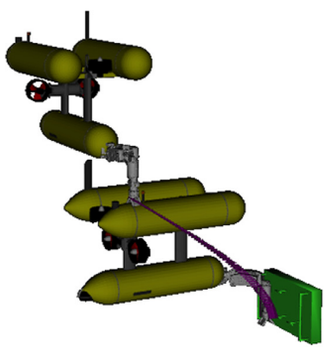
(d) *Lazy ARA* — Euclidean — Workspace.*

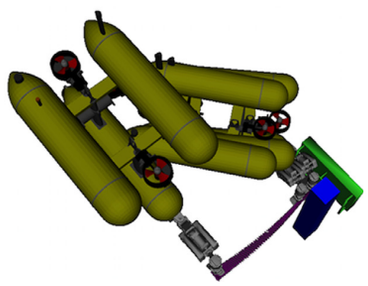
Fig. 6. Search-based: I-AUV end effectory trajectory changes due to obstacle addition — BM#1 vs. BM#2.



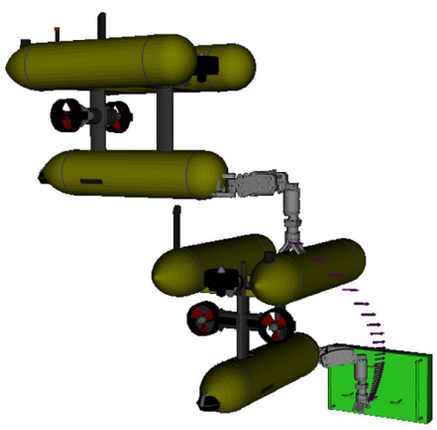


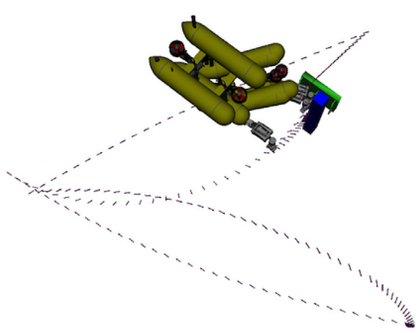
(a) STOMP.





(b) CHOMP.



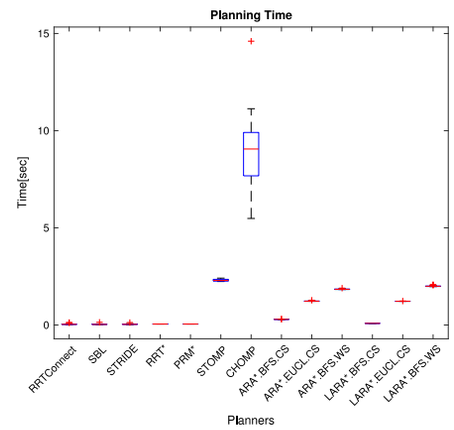


(c) **S**ingle **Bi**-directional **L**azy collision checking.

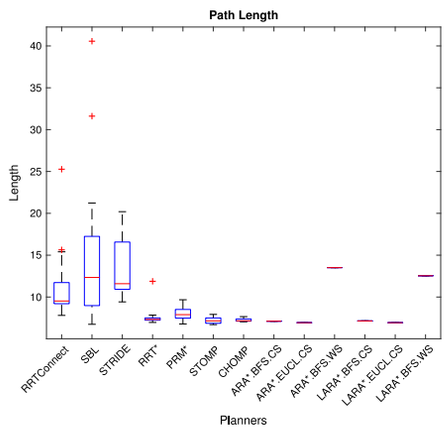
Fig. 7. Sampling & optimization based: I-AUV end effectory trajectory changes due to obstacle addition — BM#1 vs. BM#2.

Finally, for BM#1, the **Distance To Obstacle Color Map** [Figs. 4(e) and 5(e)] shows an intuitive behavior of all the algorithms, starting in blue and evolving smoothly to orange. Nevertheless, transiting to BM#2, an increase in the number of points for the sampling planners (reflected in higher path length and a taller bar) is observed, as well as an increase of the distance to obstacle range. The explanation of the second finding, is found in the tendency of some planners (*SBL*, *STRIDE* and *RRTConnect*) of finding longer paths moving away from the goal, in wider areas (i.e. more samples), then moving again in the direction of the goal. This interpretation can also be seen in the alteration of the bar colors of these planners, and the unnatural transitions between the different regions for BM#2.

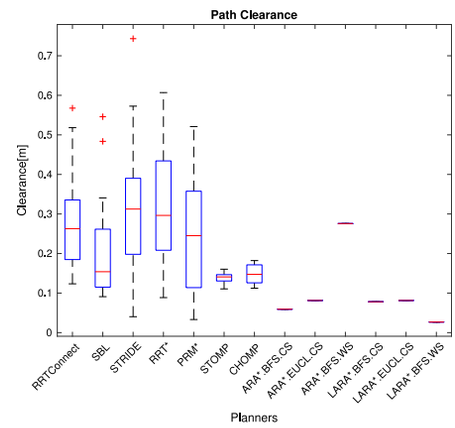
Finally, a 100% *success ratio* is achieved for all planners in BM#1, while conversely the *SBL* fails 60% of the runs, and *CHOMP* was not able to plan in BM#2.



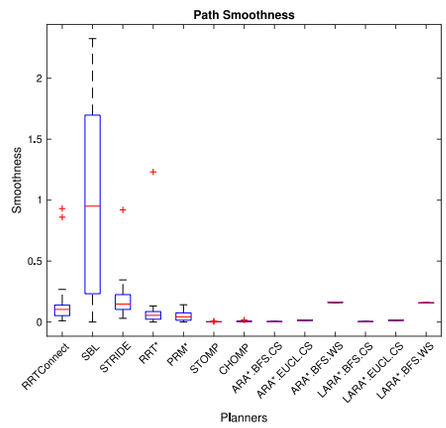
(a) Planning time.



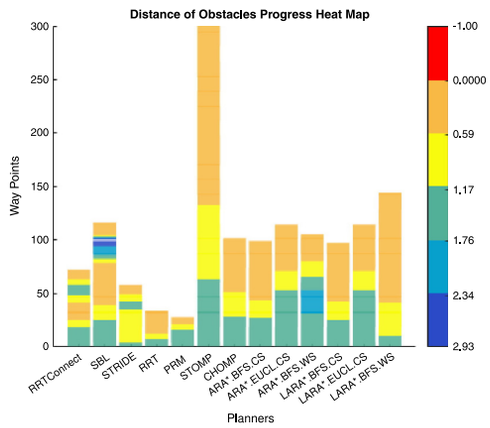
(b) Path length.



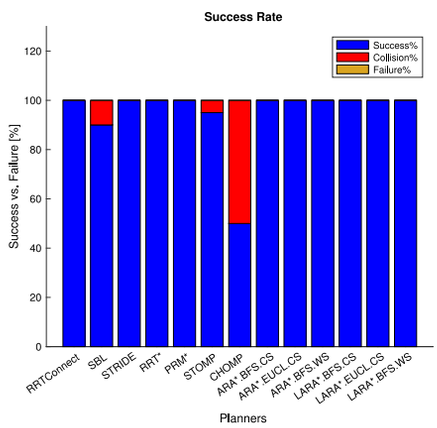
(c) Path clearance.



(d) Path smoothness.



(e) Distance to obstacle heatmap.



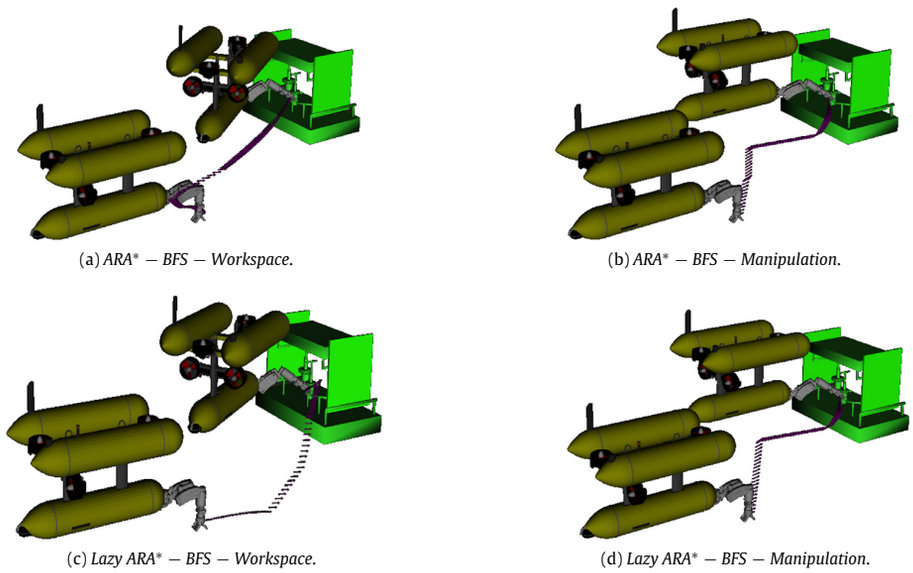
(f) Success rate.

Fig. 8. Benchmark results for unplugging a connector in a cluttered environment.

8.2. Intervention in cluttered environment "unplug connector"

In order for this intervention to succeed, the target connector had to be removed from collision check (i.e. allowed to touch) with the end-effector. It is worth noting that both normal and lazy *ARA*.EUCLID.WS* constantly failed to find a collision-free path within the 20 s allowed. It is highly probable that providing additional time would lead to successful planning. In addition, some planners encountered higher failure rates due to collision, especially *CHOMP* as in Fig. 8(f).

Planning Time: [Fig. 8(a)] *CHOMP* recorded the highest planning time (between 6 and 12 s) with the highest variation along the runs as well. Sampling-based planners outperformed the rest, keeping the same time range as in BM#1 & BM#2. Only *BFS.CS*



(a) *ARA* — BFS — Workspace.* (b) *ARA* — BFS — Manipulation.*

(c) *Lazy ARA* — BFS — Workspace.* (d) *Lazy ARA* — BFS — Manipulation.*

Fig. 9. Different search-based planners generated trajectories for BM#3.

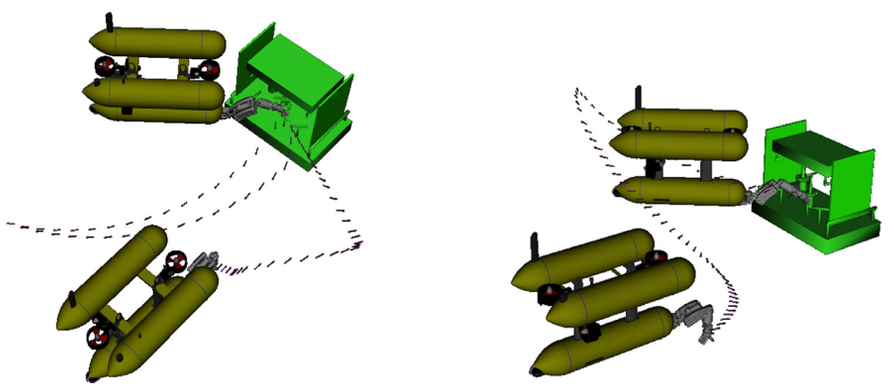


Fig. 10. RRTConnect & STRIDE trajectories for BM#3.

was able to compete with its sampling-based counterparts, while *STOMP* and the *BFS.WS* (normal and lazy) recorded similar planning times, higher than the others.

Path Length: [Fig. 8(b)] All planners other than the sampling-based ones, kept almost the same path length over the runs, demonstrating comparable values to previous benchmarks. *RRTConnect*, *SBL*, *STRIDE* suffered the most from outliers and recorded higher path length. However, it is worth noting that both *RRT**, *PRM** were able to keep a lower range by successfully optimizing their cost (i.e. path length) compared to non-optimal sampling-based planners.

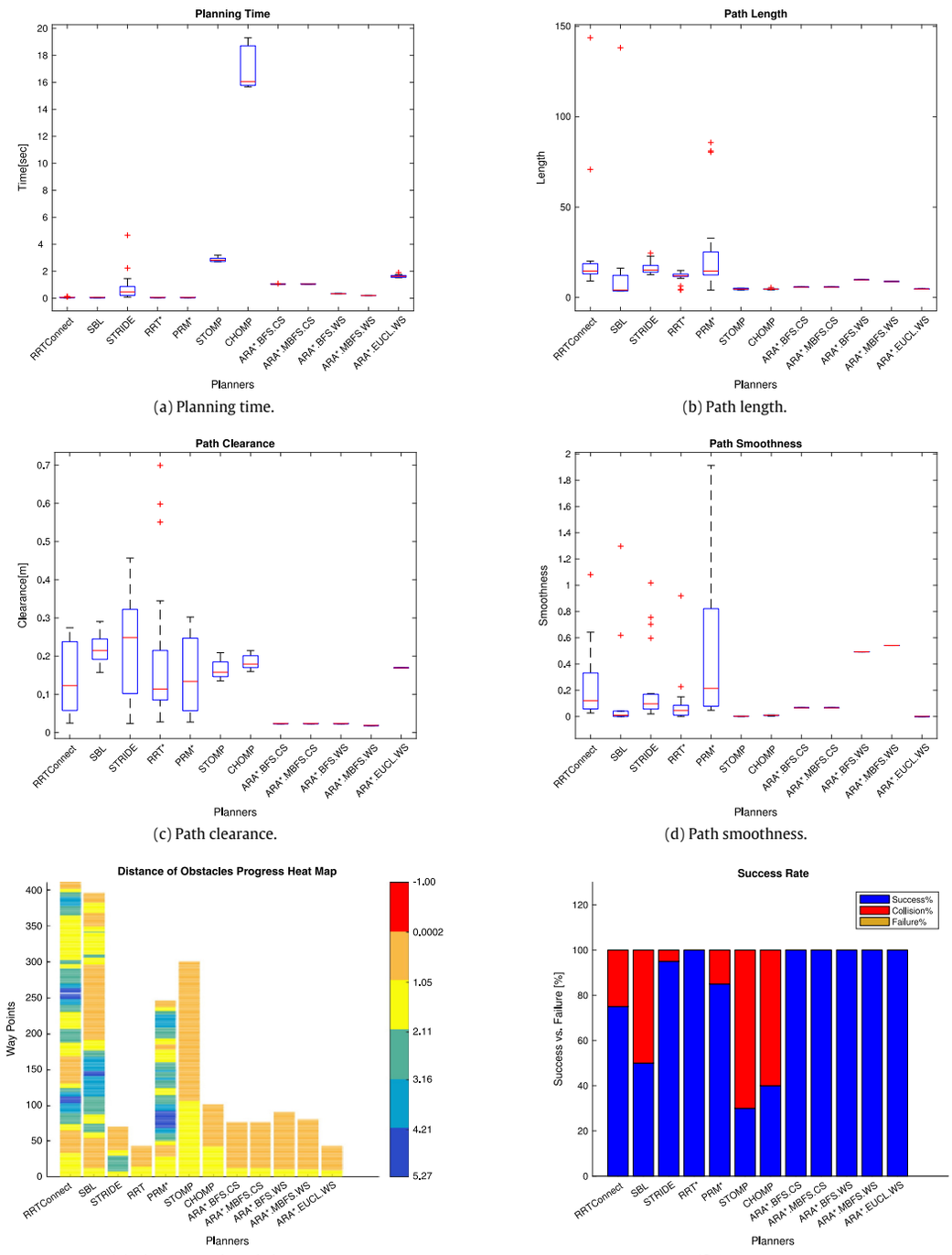
Path Clearance: [Fig. 8(c)] Similar to BM#1 & BM#2, sampling-based and optimization-based planners outperformed their search-based counterparts. The lazy *BFS.WS* showed lower clearance compared to the non-lazy version, that actually achieved the highest clearance among all search-based planners (which justifies its higher path length). The trajectories generated by both *BFS.WS* and *BFS.CS* are shown in Fig. 9.

Path Smoothness: [Fig. 8(d)] Surprisingly, sampling-based planners recorded higher smoothness specially the *SBL*, even though they suffered from high variation across the runs. Only the *BFS.WS* planners (both normal and lazy) were able to compete.

For the **Distance To Obstacle Color Map** [Fig. 8(e)], similar to BM#2, some of the sampling-based planners suffered swings both close to and far from the obstacles region. An example trajectory of *RRTConnect* is shown in Fig. 10, where the path gets close to the target but is almost touching one of the surrounding valves, and could not find a way out except by moving away from the obstacles region. This is basically due to the nature of this planner which grows two trees (from start and goal configurations) and tries to connect them, and added to this the uniform sampling, which generates more samples in wider areas. A minor jump as well is shown in the BFS workspace case (seen also in the generated trajectory) where the planner adapted its depth, by a backward upward motion, before proceeding to the goal. The interpretation of this behavior is highly likely related to the motion primitives set and the grid resolution.

8.3. Intervention in narrow passage "stick manipulation"

The various results are shown in Figs. 11 and 12. The complexity of this benchmark is reflected in higher failure rates for some planners, as in Figs. 11(f) and 12(f). For the 1 m stick manipulation, there is a percentage of the failure rate which is not due



(a) Planning time.

(b) Path length.

(c) Path clearance.

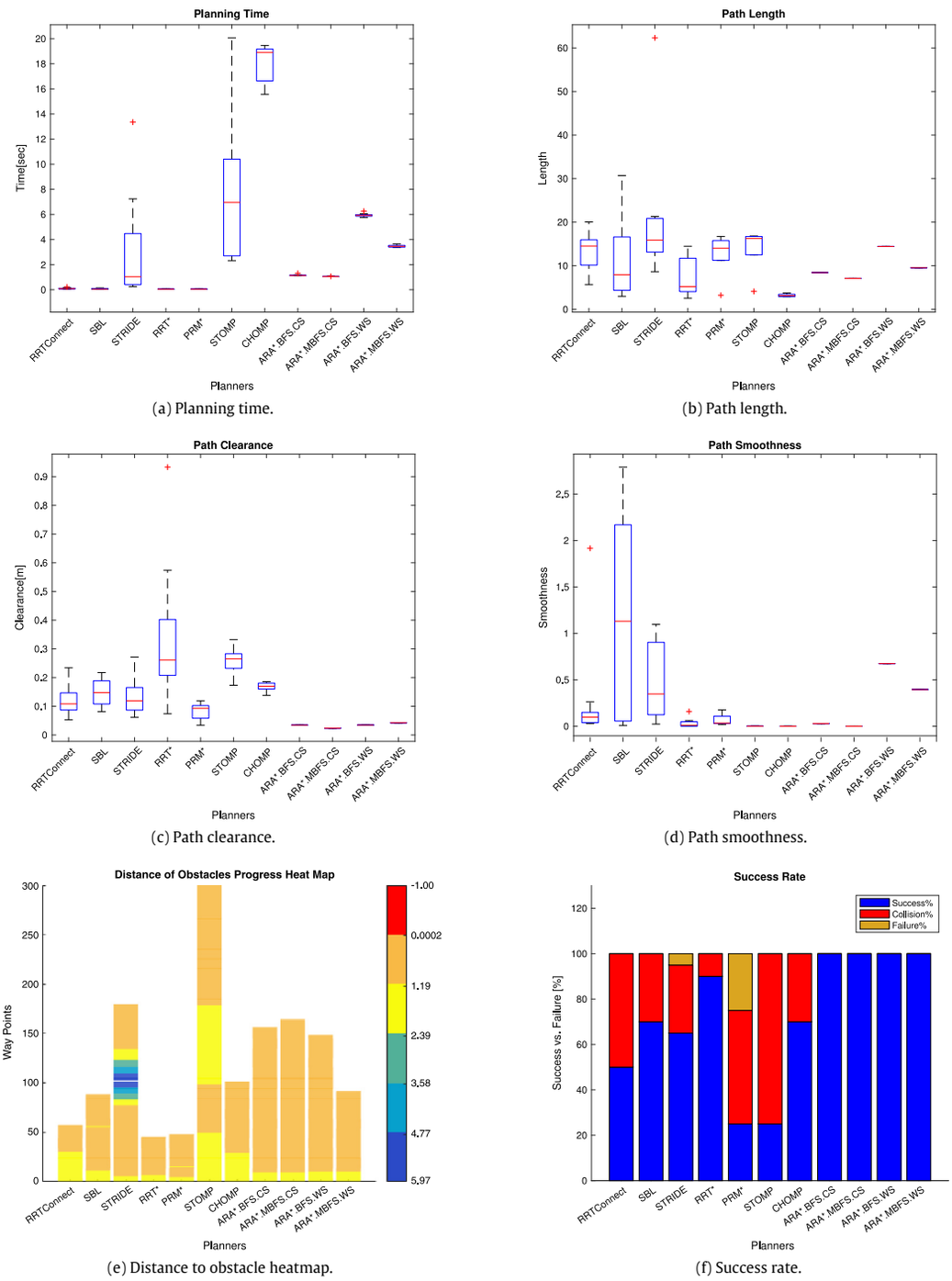
(d) Path smoothness.

(e) Distance to obstacle heatmap.

(f) Success rate.

Fig. 11. Benchmark results for 50 cm stick manipulation across a narrow window.

to collision, but rather to the inability of the planner to find a solution (e.g. being unable to sample valid states around start/goal configurations). The search-based planners were also affected by the complexity, which was reflected in less planners (with respect to previous benchmarks) succeeding to find a solution within the 20 s time frame. This is mostly seen in BM#5 where all euclidean search-based methods failed. In addition, for the workspace lattice, different end-effector resolution for BM#4 & BM#5 was used. For the multi-frame heuristic, a second frame was defined shifted 0.1 unit in both X & Y directions. The motion primitives used by the manipulation lattice are a sub-set of the ones in Fig. 3, with a decrease by a factor to allow short-range motions and more accurate manoeuvring across the narrow window. The total number of motion primitives are 18 long ones, and 2 short ones. In addition, *STOMP* was highly affected by the complexity of the benchmark especially for the 1 m-stick case. Using the same configuration



(a) Planning time.

(b) Path length.

(c) Path clearance.

(d) Path smoothness.

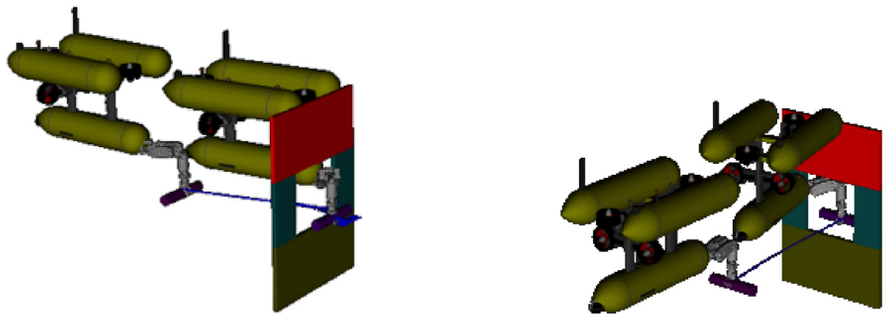
(e) Distance to obstacle heatmap.

(f) Success rate.

Fig. 12. Benchmark results for 1 m stick manipulation across a narrow window.

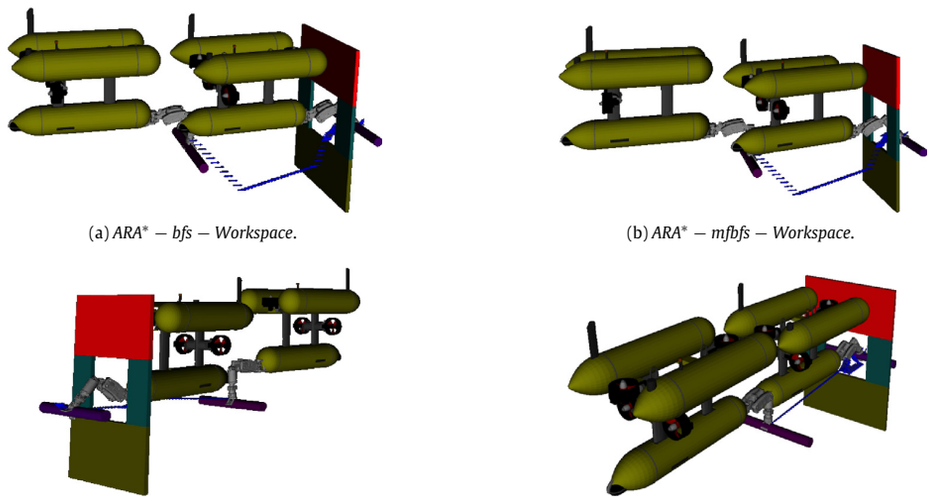
as for the 50 cm-stick leads to 100% failure, we had to increase the standard deviation of the noise generation, from 0.1 to 0.5 for all the joints. This increase allowed higher randomness in the explored paths and finally it was able to find solution in 20% of the cases, regardless of the path quality. The random trajectories generated are shown in Fig. 16, where it was allowed to explore the area around the window, instead of only exploring paths going through the window. Looking at the sampling-based plots for BM#4 & BM#5, the variations between the runs is more evident. As a general note, the sampling-based performance decayed compared to BM#1 & BM#2, as expected, since the *Narrow Passage* remains a challenge for those techniques.

Planning Time: [Figs. 11(a) and 12(a)] The complexity of the two tests with respect to each other is reflected in a wider range of planning times for BM#5. Sampling-based planners were still able to bypass other planners. There was a noticeable variation



(a) *ARA* — euclidean — Manipulation.* (b) *ARA* — euclidean — Workspace.*

Fig. 13. Search-based: I-AUV end effectory trajectory for the 50 cm stick manipulation.



(a) *ARA* — bfs — Workspace.* (b) *ARA* — mfbfs — Workspace.*

(c) *ARA* — bfs — Manipulation.* (d) *ARA* — mfbfs — Manipulation.*

Fig. 14. Search-based: I-AUV end effectory trajectory for the 1 m stick manipulation.

along the runs of *STRIDE, CHOMP* & *STOMP*, with higher variation for the 1 m-stick. *CHOMP* recorded the slowest planning time for BM#4 & BM#5, whereas *STOMP* had a similar performance for the second. The search-based *CS* showed comparable performance to sampling-based ones for the 1 m-stick, yet the *WS* outperforms the *CS* for the 50 cm-stick.

Path Length: [Figs. 11(b) and 12(b)] Optimization-based & search-based outperformed sampling-based as in previous scenarios. The latter shows variations and outliers along the runs. Similar variations appeared for *STOMP* in the 1 m-stick case. The average path length for the search-based planner increased for BM#5 specially the *BFS*.*WS*.

Path Clearance: [Figs. 11(c) and 12(c)] The search-based planners demonstrated the least clearance. This may be due to the fact that the generated trajectories go through the window creating a challenge to maintain high clearance given the length of the stick and the arm dimensions, a reason why the clearance decreases even more for the 1 m-stick. Conversely, in most cases the sampling-based and optimization-based generate a trajectory going around the window, thus leading to higher clearance.

Path Smoothness: [Figs. 11(d) and 12(d)] Similarly to path length, optimization based & search-based outperformed sampling-based.

In the **Distance To Obstacle Progression:** [Figs. 11(e) and 12(e)] plot, the sampling-based planners can be clearly seen to have difficulties with the narrow passage and their inability to sample in narrow areas. This is observed in the swing in and out of the intervention region. *STOMP* suffered similar swings in the 1 m-stick case. Different trajectories are shown in Figs. 13–14. As an overall observation, the search-based planners performance was not highly affected by the increasing complexity between the two cases except for the planning time. However, optimization-based & sampling-based planners demonstrated a degradation in almost all metrics.

8.4. Across benchmarks quantitative comparison

In the present section, we cover additional quantitative measures in order to conclude how the complexity of the benchmarks affect the planning output. We are mainly concerned with: (1) Planning Consistency, and (2) what can be considered natural human-like motion. The latter is specifically important in applications that involve human interaction or human-cluttered environments, where unexpected or unnatural behavior is dangerous.

8.4.1. Consistency index

In most of the work presented in the literature a separate benchmark was dedicated to measure the consistency of a given motion planner, and reported through measuring the mean path

length. Such a benchmark consists of repeating a motion between the same (start-goal) configurations or same start and various goals distributed on a cube separated by small distances. In this work, we propose a new measure "consistency index **C.I.**" to quantify the consistency of a given motion planner. The *C.I.* defined as inversely proportional to the normalized mean of the amount of variation of the distance traveled along the M runs by the N DoF of the system. Mathematically it is computed as in (Eq. (4)). Given a certain run j $(0 < j < M)$ of a given planner, a joint q_i $(0 < i < n)$ and a way-point q_{ij}^k $(< 0 < k < N)$ of the τ_j trajectory of run j, μ_{ij} is the average of the distance traveled by q_i along τ_j. Then, μ_i and σ_i are respectively, the mean and the standard deviation of μ_{ij} along the M runs. They represent how the average distance μ_{ij} of joint q_i along τ_j changes over the M runs. For consistent trajectories μ_{ij} is not expected to change significantly over the M runs, so σ_i should be small. Since we are interested in a single index to evaluate a planner, we need a single consistency value for all the DoFs. For this reason we averaged σ_i for all DoFs getting μ_σ. In addition, to obtain an index in the $[0-1]$ range, μ_σ was normalized by dividing by η, the maximum σ_i of all joints, which was then subtracted from 1 to make the index higher for more consistent trajectories. Finally, in case of failure, the consistency index is penalized by a factor $\epsilon = f/M$ where f is the number of failures. Table 2 shows the consistency index of all tested planners for the five benchmarks. We can observe that the search-based planners have been consistent for all tested scenarios, with one exception for the *ARA*.EUCLID.WS* in BM#1. A logical interpretation is that this is due to the redundancy of the system, and because this planner relies on frequent inverse kinematics calls to go from the end-effector space to the joint-space, it is possible to have a source of randomness due to the different IK (inverse-kinematic) responses each time. Whereas the optimization-based planners showed high consistency for simpler benchmarks (BM#1 for *CHOMP & STOMP*, and BM#2 for *STOMP*), the index noticeably decreased for BM#4. In addition, *STOMP* recorded lower consistency compared to *CHOMP*, as it involves higher randomness in its nature (the stochastic process of generating noise). Finally, the sampling-based planners showed the least consistency and especially with BM#5 which presented the smallest index among all benchmarks.

$$C.I = (1 - \frac{\mu_\sigma}{\eta}) - \epsilon \; ; \; \mu_\sigma = \frac{\sum_{i=1}^{n} \sigma_i}{n} \; ; \; \eta = \max(\sigma_i)$$
$$\mu_i = \frac{\sum_{j=1}^{M} \mu_{ij}}{M} \; ; \sigma_i = \frac{\sum_{j=1}^{M} |\mu_{ij} - \mu_i|}{N} \; ; \; \mu_{ij} = \frac{\sum_{k=1}^{N} |q_{ij}^k - q_{ij}^{k-1}|}{N} \quad (4)$$

8.4.2. Natural motion index

For some applications (e.g. involving human–robot interaction), we would like to achieve what could be called a natural motion, or a predictable human-like behavior. For example, in BM#4 & BM#5, a human-like motion would manoeuver the stick through the window instead of circumnavigating around it.

We define a new statistical measure "Natural-Motion Index **N.M.I.**", to quantify how close/far is the motion from what would be considered ideal by human. Considering a set of concentric spheres centered in the goal with increasing radius (Fig. 17 where the heatmap-like color reflects distance to the goal). A natural end-effector motion would successively move forward (transiting from colder to warmer colored regions) to the goal and perform backwards motion only in case of avoiding obstacles. In this case, the maximum distance between 2 consecutive way-points of the trajectory is not expected to transit more than one region. However, we consider an oscillatory motion between 2 consecutive regions to be natural (e.g. to avoid obstacles), while back and forth motions with a length greater than a region are considered unnatural. For this metric, we have chosen to work in the workspace, instead of the configuration-space due to the system redundancy, resulting in different joint configuration for different planners/runs for the same goal. An example of natural vs. unnatural motions is shown in Fig. 17. In this example, we consider a trajectory of 6 segments (a total of 7 way-points), the segments indicated with the bold black arrows are the natural ones, where the trajectory progresses in the direction of the goal, moving forward between one or more transition regions. While the dotted arrows (i.e. S3 & S5) demonstrate a backward transition over more than one region (e.g. from yellow to light blue).

Table 2
Planners consistency index across Set(1) on the left & Set(2) on the right for the 5 benchmarks.

Planner	BM#1	BM#2	BM#3
RRTConnect	0.59	0.66	0.56
SBL	0.62	0.0	0.4
STRIDE	0.68	0.59	0.42
RRT*	0.85	0.7	0.57
PRM*	0.83	0.69	0.52
STOMP	0.99	0.99	0.93
CHOMP	1	–	0.47
ARA*.BFS.CS	1	1	1
ARA*.EUCLID.CS	1	F	1
ARA*.BFS.WS	1	1	1
ARA*.EUCLID.WS	0.96	1	F
LARA*.BFS.CS	1	1	1
LARA*.EUCLID.CS	1	F	1
LARA*.BFS.WS	1	1	1
LARA*.EUCLID.WS	1	1	F

Planner	BM#4	BM#5
RRTConnect	0.56	0.17
SBL	0.15	0.23
STRIDE	0.76	0.27
RRT*	0.75	0.44
PRM*	0.63	−0.17
STOMP	0.29	0.19
CHOMP	0.39	0.68
ARA*.BFS.CS	1	1
ARA*.MFBFS.CS	1	1
ARA*.EUCLID.CS	1	F
ARA*.BFS.WS	1	1
ARA*.MFBFS.WS	1	1
ARA*.EUCLID.WS	1	F

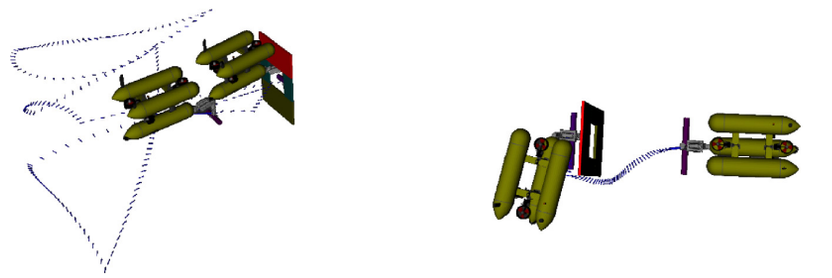
In order to compute this index, we first compute the radius of each of the 5 regions, by splitting the start to goal euclidean distance into two equal parts. Then, a label $Region(q_k)$ is assigned to a way-point q_k by computing the Euclidean distance between the workspace pose of the configuration q_k (let it be $x_{ee}(q_k)$) and the goal pose and comparing this distance with the previously computed region's radius. For example, on Fig. 17, the two way points of $S1$ will be labeled as $R5$. The final index is computed by comparing each way-point label with its previous one. The index is penalized (increased by 1) when a non-smooth transition is found.

Mathematically, the *N.M.I.* is computed as **card**(S) which is the cardinality of the set S, with S containing those way-points whose region $Region(q_k)$, compared with the region of the previous way point $Region(q_{k-1})$, transits mores than a single region (see Eq. (5)).

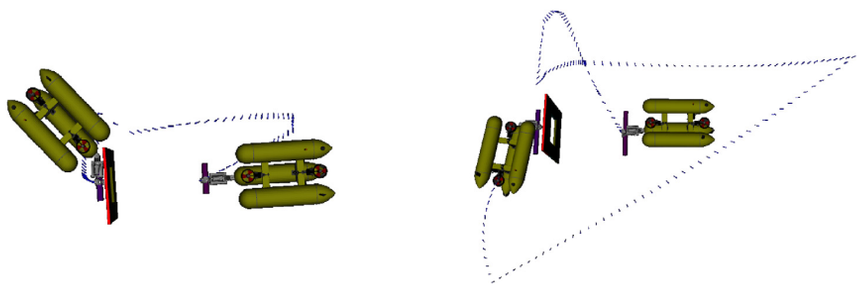
$$\text{Let } S = \{x_{ee}(q) \,/\, Region(x_{ee}(q_k)) - Region(x_{ee}(q_{k-1})) < -1\}; NMI = \textbf{card}(S) \quad (5)$$

Table 3 shows: the minimum, the maximum, and the mean of the *N.M.I.* of the planners along the 20 runs and for the 5 benchmarks as in Eq. (5). The higher the index value, the more transitions it exhibits and the less natural is the motion. *STOMP* & *CHOMP* suffered the most for the cluttered environment in BM#3. Similarly, the search-based planners showed less natural motion

(a) *RRT**.



(b) *PRM**.



(c) STRIDE.

Fig. 15. Sampling-based: I-AUV end effectory trajectory changes due to obstacle addition — BM#4 vs. BM#5.

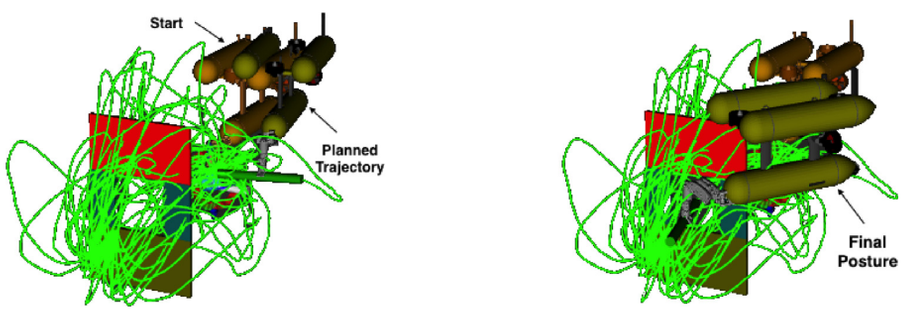


Fig. 16. STOMP stochastic trajectories for BM#5.

for the cluttered environment, with the *EUCLID* heuristic being less affected. It is possible that the behavior of this heuristic is due to its nature of being goal-directed. On the other hand, the sampling-based planners' transitions increase continuously with the benchmark complexity, with the highest value being shown for the narrow passage tests (BM#4 & BM#5), while they behave naturally for the simple BM#1.

8.5. Optimality measure for optimal planners

In this section, we analyze the optimality measures provided by the different optimal planners. The notion of optimality and the cost function optimized by each of them vary as explained below.

8.5.1. Sampling-based planners

For the asymptotically-optimal sampling-based planners [68] it is interesting to study the progress of the cost function as well as the number of iterations executed within the allowed planning time. We imported this feature to the *MoveIt!* benchmark plug-in, in order to observe how both *RRT** & *PRM** perform with respect to each other for the various benchmarks. The cost function, in this case, is the path length. For each benchmark and planner the shortest (best) and the longest (worst) paths among the M runs is shown in Fig. 18. The cost function is initialized with infinity, and with the first solution found the value changes to its path length. It is important to note that the shown path length is that of the original path before post-processing (i.e. smoothing and short-cutting), thus the values may not match those shown in the previous box plots.

Table 3
Upper → natural motion index across Set(1) BMs & lower → natural motion index across Set(2) BMs.

Planner	BM#1			BM#2			BM#3		
Transition Factor	min	μ	max	min	μ	max	min	μ	max
RRTConnect	0	0.05	1	0	9.6	37	5	16.85	44
SBL	0	0	0	0	30.8	98	0	15.67	57
STRIDE	0	0	0	1	26	52	5	23.2	43
RRT*	0	0	0	0	0.8	15	2	4.35	13
PRM*	0	0	0	0	0.8	7	2	7.5	17
STOMP	13	16.4	20	6	16.4	33	72	76	81
CHOMP	10	10	10	–	–	–	29	29.5	30
ARA*.BFS.CS	0	0	0	0	0	0	45	45	45
ARA*.EUCLID.CS	2	2	2	–	–	–	22	22	22
ARA*.BFS.WS	0	0	0	0	0	0	63	63	63
ARA*.EUCLID.WS	0	2.4	3	0	0	0	F	F	F
LARA*.BFS.CS	0	0	0	2	2	2	43	43	43
LARA*.EUCLID.CS	1	1	1	–	–	–	22	22	22
LARA*.BFS.WS	1	1	1	1	1	1	58	58	58
LARA*.EUCLID.WS	0	0	0	0	0	0	F	F	F

Planner	BM#4			BM#5		
Transition Factor	min	μ	max	min	μ	max
RRTConnect	0	50.4	356	0	12	46
SBL	9	46.1	289	0	9.78	48
STRIDE	1	20.16	46	0	22.5	101
RRT*	0	2.3	17	0	1.3	7
PRM*	0	40.58	170	0	31.4	69
STOMP	0	6	17	0	31.6	80
CHOMP	0	4	8	0	0	0
ARA*.BFS.CS	0	0	0	0	0	0
ARA*.MFBFS.CS	0	0	0	0	0	0
ARA*.EUCLID.CS	0	0	0	F	F	F
ARA*.BFS.WS	0	0	0	0	0	0
ARA*.MFBFS.WS	0	0	0	0	0	0
ARA*.EUCLID.WS	0	0	0	F	F	F

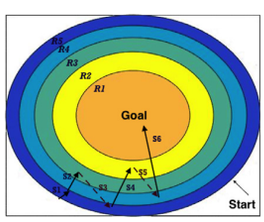


Fig. 17. An example of transitions over a heat-map representation of transition regions in the context of N.M.I computation. Orange → Close to Goal & Blue → Far from Goal. The black solid arrows shows a natural expected progress of the trajectory vs. the dotted arrows where the motion is considered unnatural.

As a general observation, the *RRT** costs for different benchmarks, for the shortest and the longest paths are quite close, indicating higher consistency in the generated plans (this can also be appreciated from Table 2). However *PRM** has wider gaps between short and long paths, specially with more complex scenarios like BM#4 & BM#5. This may be due to the nature of each algorithm, as the RRT family is tree-based goal-oriented, while the PRM is multi-query road-map based. Another observation is that *PRM** in its best case generated the same or a better cost than *RRT**, whereas for the worst case, the final path cost was higher than the *RRT**. In the simplest scenario (BM#1), both planners were able to find low-cost solutions with no further improvement over time. However for the highly complex BM#5, both planners were not able to optimize much if at all, resulting in high-cost final solution. It also happens, for the same benchmark, that the *PRM** worst case could not find a non-infinity solution cost until $T = 17$ s, being the highest cost among all test cases.

8.5.2. Search-based planners

As heuristic-based planners, their purpose is to find optimal solutions in terms of the heuristic function (i.e. path length). Three parameters were considered in order to analyze the optimality of the solution found by each combination of the search-based planners: 1- The number of expansions, indicating the size of the graph (which can also indicate the memory usage). 2- The optimality measure *epsilon* of the path found (1 indicates the optimal solution). 3- The solution cost which sums the edges and nodes cost from start to goal. Table 4 gives an impression of the best performing planner *ARA*.BFS.WS*. It was able to generate close to optimal, low cost paths even with the most complex scenarios, and also the least expansion of nodes. On the contrary *ARA*.BFS.CS* performance was highly affected for BM#5. In general the manipulation graph representation is less competent in terms of optimality, even though changing the set of motion primitives might improve its performance.

8.5.3. Optimization-based planners

CHOMP does not directly optimize the path length but it minimizes the sum squared velocities through what is called a smoothness cost function (more details about the computations in [51] &[52]), while keeping a minimum clearance to obstacles. As a result, in some cases the path length might increase in order to keep certain clearance. However, *STOMP* cost function (given the implementation integrated with *MoveIt!*) relies on optimizing the distance to obstacles (thus higher clearance).

From Table 5, we can conclude that *STOMP* is able to generate higher clearance paths with a smaller number of iterations. As expected, the cost of the solution found is proportional to the benchmark complexity, which is not necessarily the same for both planners. *CHOMP* shows a higher cost for the cluttered environment, due to the collision objects representation it uses, and given

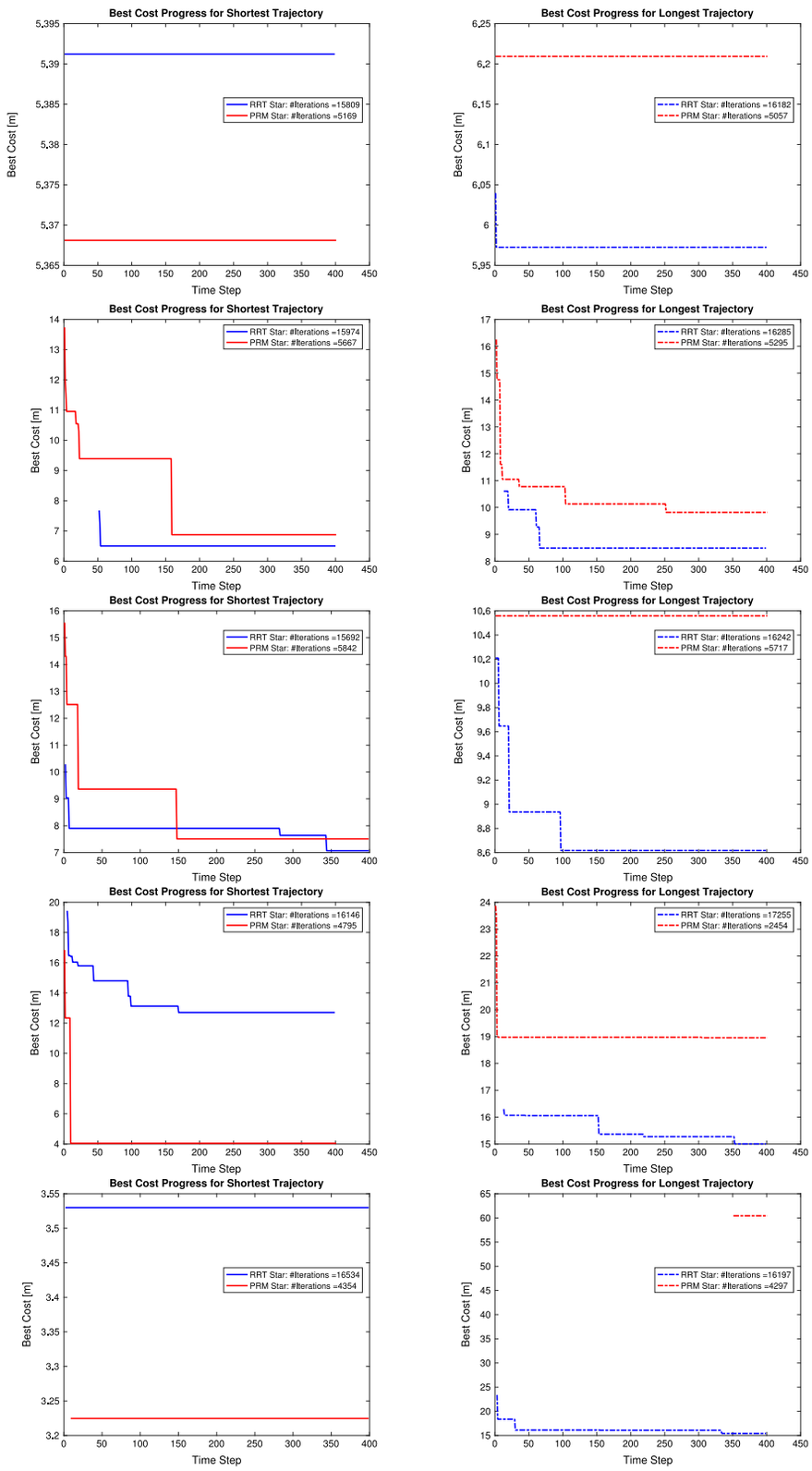


Fig. 18. Cost Function [Path-Length] Progress for Sampling-based Sub-Optimal Planners across all Benchmarks in Order. Left Column → shortest-path & Right Column → longest-path.

Table 4
Number of expansions, path optimality & cost for search-based planners.

Planner	# Expansion					Epsilon					Cost ($*10^3$)				
	BM#1	BM#2	BM#3	BM#4	BM#5	BM#1	BM#2	BM#3	BM#4	BM#5	BM#1	BM#2	BM#3	BM#4	BM#5
ARA*.BFS.CS	104	130	118	269.9	346	3	5	1	8	40	6	8	5	9	15
ARA*.MFBFS.CS	–	–	–	266.7	328.7	–	–	–	4	8.75	–	–	–	9	12
ARA*.EUCLID.CS	171	–	287.65	278	–	3	–	4	17	–	8	–	10	15	–
ARA*.BFS.WS	40	25	48	48	962	1	1	1	1	3	0.54	0.51	0.36	0.698	1.1
ARA*.MFBFS.WS	–	–	–	22	647	–	–	–	1	2	–	–	–	0.57	0.75
ARA*.EUCLID.WS	313.5	218.9	–	292	–	2	4	–	2	–	3.36	3.87	–	2.55	–
LARA*.BFS.CS	782.5	997.5	176	–	–	1.8	1	1	–	–	6	7	5	–	–
LARA*.EUCLID.CS	916	–	2076	–	–	1.6	–	2.4	–	–	7	–	10	–	–
LARA*.BFS.WS	28	28	64	–	–	1	1	5	–	–	0.54	0.51	0.36	–	–
LARA*.EUCLID.WS	447	308	–	–	–	1.4	5	–	–	–	3.36	3.81	–	–	–

Table 5
Number of iterations & cost for optimization-based planners.

Planner	# Iterations					Cost				
	BM#1	BM#2	BM#3	BM#4	BM#5	BM#1	BM#2	BM#3	BM#4	BM#5
STOMP	1	1.3	1	1	4.6	665.8	737.49	569.3	634.67	906.45
CHOMP	11	–	153.4	60.5	76.85	19742	–	35359	19371	19199

that in this benchmark there are many cluttered objects, the cost of the initial path found is already high. While *STOMP* suffers more with the 1 m stick manipulation.

9. Guidelines & conclusions

To conclude this paper, we will provide guidelines to answer the fundamental question proposed at the beginning: "Given a certain problem, which is the best motion planner to choose?". Unfortunately, there is no single technique that would perfectly fit, out of the box, all possible applications, systems and environments. Nevertheless, a valid open question would be: "Which motion planning technique to start with?". The purpose of the work presented here is to provide an insight on how to tackle this question.

It is obvious from the previous analysis, that *non-optimal sampling-based* techniques would fit the highly time-constraint applications well, as they demonstrated the fastest planning time across all benchmarks, at the expense of other quality metrics like path length and natural motion generation. The *asymptotically-optimal sampling-based methods* compromise a cost function (e.g. path length) with the planning time. An additional advantage of the sampling-based planners provided by the OMPL library, is that they are well tweaked for a wide range of applications, and their default parameters, sampling-strategy and distance metric work quite well. This leaves room for improvement to overcome their weaknesses using application-oriented parameters and sampling strategies.

On the other hand, the *search heuristic-based planners* showed an advantage in generating optimal trajectories, with more natural motion and higher consistency, while still meeting average time performance. The planning time is highly affected by the number of motion primitives and the resolution chosen. Increasing the motion primitives would lead to better coverage at the cost of spending more time searching the graph. A major disadvantage is their dependency on parameter tweaking (i.e. resolution & motion primitives). In particular, for the manipulation lattice, designing the motion primitives requires significant effort and a considerable knowledge of the environment. Using convenient heuristics, it would be possible to adapt them to deal with more complex objective functions (e.g. the manipulability index).

As for the *optimization-based planners*, overall their performance is average among the others, being highly affected by the benchmark complexity. Their strongest advantage, is that they solve the motion planning as an optimization problem, which allows the combining of complex cost functions. This raises another issue with such planners, the complexity of their mathematical background, which impacts on their implementation.

As a summary, Table 6 presents the planners categories versus the various features/requirements for a typical motion planning problem. Whenever a feature is marked possible, it means that it showed average performance against this feature in our tests; or, that simple parameter tuning would enhance its performance against such a feature. A feature is considered as not supported, if the nature of the planner does not (without major changes) support it.

The methodology followed in the work presented involves: First, study of the type of application to be covered. This involves the features of the possible environment (e.g. cluttered obstacles) and any constrained requirements (e.g. time constraints). The system available might add complexity to the problem as well. For example, in the case presented here, having a limited 4-DoF arm, was a reason why we modeled the base and the arm as one system in order to gain higher reachability to achieve full control of the end-effector. Second, having this image in mind, proceed with defining a set of application oriented benchmarks and test cases. This is essential both to give more insights into the applicability of different techniques to the application, and to augment the benchmarks database to be shared among the robotics community.

The case study we are considering for our future work, involves using the UVMS for underwater free-floating manipulation with the following requirements:

(1) Fast planning response to allow real-time execution.

(2) Cluttered non-moving obstacles, possibly leading to narrow passages in some cases.

(3) *A priori* unknown environment, this would require real-time re-planning capabilities to deal with the changes.

(4) Generated trajectories with a combined optimization function: short path, good manipulability index, a smart selection for the redundant DoFs (e.g. during the intervention, it might be preferable to move the arm and not the AUV).

(5) Consistency is preferable given the incomplete knowledge about the environment and the possible sensor error scanning the environment, even though it is not as essential as with other applications interacting with humans.

What is not covered in this case study, is moving obstacles as well as dealing with uncertainty either in the sensor or in the system localization and motion. In addition, higher clearance and smoothness are preferred but are not the main interest. Given such requirements as well as the planners vs. environment features, a

Table 6
Planners vs. environment features.

Planners categories	Real-Time Response	Cluttered	Narrow passage	Consistency	Human-Like motion	Fast Re-plan	Clearance	Objective function
Non-Optimal Sampling	✓	possible	✗	✗	✗	possible	✓	✗
Optimal Sampling	possible	possible	✗	✗	✗	possible	✓	✓
Heuristic anytime A* workspace	possible	possible	✓	✓	✓	possible	possible	possible
Heuristic anytime A* manipulation	possible	possible	✓	✓	✓	possible	possible	possible
Optimization-based	✗	possible	✗	✗	possible	possible	✗	✓

summary is provided in Table 6, where the potential of each planner is marked in green. Our own conclusion would be to go for the *Anytime A*.BFS heuristic.workspace lattice*. This planner has a high potential in all the basic features with above average performance, with consistent performance among different environments. We prefer it over the manipulation lattice even though they share similar features, due to the dependency of its performance on the motion primitives set. A second possible choice is the *RRT**, as it showed promising behavior with possible improvements using other sampling strategies. Finally, this choice does not imply using the planner as it is, but includes the possibly of adapting it to the study case requirements.

In terms of data analysis limitations, this work did not cover memory usage analysis or collision checkers profiling. They might be part of future work as we believe they have not received appropriate attention in the literature, even though their effect on choosing the convenient planner. A more advanced case study that needs further future attention, is a dual-arm or multiple cooperative vehicles system.

Acknowledgments

We would like to thank Andrew Dornbush from the Robotics Institute — Carnegie Mellon, for his great effort in providing technical support and theoretical insights about their new search-based motion planning library "SMPL", that has been used to perform the related tests.

References

[1] S.M. LaValle, Planning Algorithms, Cambridge University Press, 2006.
[2] H.M. Choset, Principles of Robot Motion: Theory, Algorithms, and Implementation, MIT press, 2005.
[3] J.-C. Latombe, Robot Motion Planning, vol. 124, Springer Science & Business Media, 2012.
[4] S.H. Tang, W. Khaksar, N. Ismail, M. Ariffin, A review on robot motion planning approaches, Pertanika J. Sci. Technol. 20 (1) (2012) 15–29.
[5] A. Sánchez López, R. Zapata, M.A. Osorio Lama, Sampling-based motion planning: A survey, Comput. Sistemas 12 (1) (2008) 5–24.
[6] D. Ferguson, M. Likhachev, A. Stentz, A guide to heuristic-based path planning, in: Proceedings of the International Workshop on Planning under Uncertainty for Autonomous Systems, International Conference on Automated Planning and Scheduling, ICAPS, 2005, pp. 9–18.
[7] A. Atyabi, D.M. Powers, Review of classical and heuristic-based navigation and path planning approaches, Int. J. Adv. Comput. Technol. 5 (14) (2013) 1.
[8] M. Saha, Motion planning with Probabilistic Roadmaps (Ph.D. thesis), Stanford, 2006.
[9] C. Goerzen, Z. Kong, B. Mettler, A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance, vol. 57, Springer, 2010, p. 65.
[10] D. González, J. Pérez, V. Milanés, F. Nashashibi, A review of motion planning techniques for automated vehicles, IEEE Trans. Intell. Transp. Syst. 17 (4) (2016) 1135–1145.
[11] L. Yang, J. Qi, D. Song, J. Xiao, J. Han, Y. Xia, Survey of robot 3D path planning algorithms, J. Control Sci. Eng. 2016 (2016) 5.
[12] S.C.I.A. Sucan, MoveIt!, Available Online:, 2013. http://moveit.ros.org.
[13] R. Ragel, I. Maza, F. Caballero, A. Ollero, Comparison of motion planning techniques for a multi-rotor UAS equipped with a multi-joint manipulator Arm, in: Research, Education and Development of Unmanned Aerial Systems, RED-UAS, 2015 Workshop on, IEEE, 2015, pp. 133–141.
[14] J. Fernández, J. Pérez, A. Peñalver, J. Sales, D. Fornas, P. Sanz, Benchmarking using UWSim, Simurv and ROS: an autonomous free floating dredging intervention case study, in: OCEANS 2015-Genova, 2015, pp. 1–7.
[15] M. Prats, J. Pérez, J.J. Fernández, P.J. Sanz, An open source tool for simulation and supervision of underwater intervention missions, in: Intelligent Robots and Systems, IROS, 2012 IEEE/RSJ International Conference on, IEEE, 2012, pp. 2577–2582.
[16] E. Galceran, M. Carreras, A survey on coverage path planning for robotics, Robot. Auton. Syst. 61 (12) (2013) 1258–1276.
[17] F. Kamil, S. Tang, W. Khaksar, N. Zulkifli, S. Ahmad, A review on motion planning and obstacle avoidance approaches in dynamic environments, Adv. Robot. Autom. 4 (2015) 134.
[18] N. Vahrenkamp, T. Asfour, R. Dillmann, Efficient inverse kinematics computation based on reachability analysis, Int. J. Humanoid Robot. 9 (04) (2012) 1250035.
[19] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A.Y. Ng, ROS: an open-source Robot Operating System, in: ICRA Workshop on Open Source Software, vol. 3, no. 3.2, Kobe, 2009, p. 5.
[20] M. Moll, I.A. Sucan, L.E. Kavraki, Benchmarking motion planning algorithms: An extensible infrastructure for analysis and visualization, IEEE Robot. Autom. Mag. 22 (3) (2015) 96–102.
[21] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D.V. Dimarogonas, D. Kragic, Dual arm manipulationa survey, Robot. Auton. Syst. 60 (10) (2012) 1340–1353.
[22] M. Reggiani, M. Mazzoli, S. Caselli, An experimental evaluation of collision detection packages for robot motion planning, in: Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on, vol. 3, IEEE, 2002, pp. 2329–2334.
[23] R. Geraerts, Sampling-Based Motion Planning: Analysis and Path Quality (Ph.D. thesis), Utrecht University, 2006.
[24] D. Katz, J. Kenney, O. Brock, How can robots succeed in unstructured environments, in: Workshop on Robot Manipulation: Intelligence in Human Environments at Robotics: Science and Systems, 2008.
[25] S. Coenen, M.M. Steinbuch, Motion planning for mobile robots–A guide, Control Syst. Technol. (2012) 79.
[26] B. Cohen, S. Chitta, M. Likhachev, Single- and dual-arm motion planning with heuristic search, Int. J. Robot. Res. (2013).
[27] O. Khatib, The potential field approach and operational space formulation in robot control, in: Adaptive and Learning Systems, Springer, 1986, pp. 367–377.
[28] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robot. Res. 5 (1) (1986) 90–98.
[29] J. Canny, A new algebraic method for robot motion planning and real geometry, in: Foundations of Computer Science, 1987, 28th Annual Symposium on, IEEE, 1987, pp. 39–48.
[30] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Chapter 15: Visibility graphs, Comput. Geom. (2000) 307–317.
[31] E.W. Dijkstra, A note on two problems in connexion with graphs, Numer. Math. 1 (1) (1959) 269–271.
[32] M. Garber, M.C. Lin, Constraint-based motion planning using voronoi diagrams, in: Algorithmic Foundations of Robotics V, Springer, 2004, pp. 541–558.
[33] R.A. Brooks, Solving the find-path problem by good representation of free space, IEEE Trans. Syst. Man Cybern. (2) (1983) 190–197.
[34] L. Kayraki, P. Svestka, J. Latombe, M. Overmars, Probabilistic roadmaps for path planning in high-dimensional configurations space, Proc. IEEE Trans. Robot. Autom. 12 (4) (1996) 566–580.
[35] I.A. Şucan, M. Moll, L.E. Kavraki, The open motion planning library, IEEE Robot. Autom. Mag. 19 (4) (2012) 72–82. http://ompl.kavrakilab.org.
[36] S.M. LaValle, Rapidly-exploring random trees: A new tool for path planning, 1998.
[37] J.J. Kuffner, S.M. LaValle, RRT-connect: An efficient approach to single-query path planning, in: Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on, vol. 2, 2000, pp. 995–1001.
[38] D. Hsu, R. Kindel, J.-C. Latombe, S. Rock, Randomized kinodynamic motion planning with moving obstacles, Int. J. Robot. Res. 21 (3) (2002) 233–255.
[39] G. Sánchez, J.-C. Latombe, A single-query bi-directional probabilistic roadmap planner with lazy collision checking, Robot. Res. (2003) 403–417.
[40] B. Gipson, M. Moll, L.E. Kavraki, Resolution independent density estimation for motion planning in high-dimensional spaces, in: Robotics and Automation, ICRA, 2013 IEEE International Conference on, IEEE, 2013, pp. 2437–2443.

[41] S. Karaman, E. Frazzoli, Sampling-based algorithms for optimal motion planning, Int. J. Robot. Res. 30 (7) (2011) 846–894.
[42] J. Pearl, Heuristics: intelligent search strategies for computer problem solving, 1984.
[43] M. Likhachev, D. Ferguson, Planning long dynamically feasible maneuvers for autonomous vehicles, Int. J. Robot. Res. 28 (8) (2009) 933–945.
[44] M. Likhachev, G.J. Gordon, S. Thrun, ARA*: Anytime A* with provable bounds on sub-optimality, in: Advances in Neural Information Processing Systems, 2004, pp. 767–774.
[45] S. Koenig, M. Likhachev, D* lite, in: AAAI/IAAI, 2002, pp. 476–483.
[46] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, S. Thrun, Anytime dynamic A*: An anytime, replanning algorithm, Science (2005) 262–271.
[47] M. Likhachev, Search-Based Planning for Large Dynamic Environments (Ph.D. thesis), 2005.
[48] M. Likhachev, A. Stentz, R* search, 2008.
[49] M. Pivtoraiko, A. Kelly, Efficient constrained path planning via search in state lattices, in: International Symposium on Artificial Intelligence, Robotics, and Automation in Space, 2005, pp. 1–7.
[50] J.M. Phillips, N. Bedrossian, L.E. Kavraki, Guided expansive spaces trees: A search strategy for motion-and cost-constrained state spaces, in: Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, vol. 4, IEEE, 2004, pp. 3968–3973.
[51] N. Ratliff, M. Zucker, J.A. Bagnell, S. Srinivasa, Chomp: Gradient optimization techniques for efficient motion planning, in: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, IEEE, 2009, pp. 489–494.
[52] M. Zucker, N. Ratliff, A.D. Dragan, M. Pivtoraiko, M. Klingensmith, C.M. Dellin, J.A. Bagnell, S.S. Srinivasa, Chomp: Covariant hamiltonian optimization for motion planning, Int. J. Robot. Res. 32 (9–10) (2013) 1164–1193.
[53] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal, Stomp: stochastic trajectory optimization for motion planning, in: Robotics and Automation, ICRA, 2011 IEEE International Conference on, IEEE, 2011, pp. 4569–4574.
[54] S. Duane, A.D. Kennedy, B.J. Pendleton, D. Roweth, Hybrid Monte Carlo, Phys. Lett. B 195 (2) (1987) 216–222.
[55] M. Kallman, M. Mataric, Motion planning using dynamic roadmaps, in: Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, vol. 5, 2004, pp. 4399–4404.
[56] J. Cannon, K. Rose, W. Ruml, Real-Time motion planning with dynamic obstacles, in: SOCS, 2012.
[57] O. Brock, O. Khatib, Elastic strips: A framework for motion genertion in human environments, Int. J. Robot. Res. 21 (12) (2002) 1031–1052.
[58] Y. Yang, O. Brock, Elastic roadmaps–motion generation for autonomous mobile manipulation, Auton. Robots 28 (1) (2010) 113.
[59] J. Kwon, T. Yoshikawa, O. Khatib, Elastic strips: Implementation on a physical humanoid robot, in: Intelligent Robots and Systems, IROS, 2012 IEEE/RSJ International Conference on, 2012, pp. 3369–3376.
[60] S.-Y. Chung, O. Khatib, Contact-consistent elastic strips for multi-contact locomotion planning of humanoid robots, in: Robotics and Automation, ICRA, 2015 IEEE International Conference on, 2015, pp. 6289–6294.
[61] K. Gochev, A. Safonova, M. Likhachev, Planning with adaptive dimensionality for mobile manipulation, in: Robotics and Automation, ICRA, 2012 IEEE International Conference on, 2012, pp. 2944–2951.
[62] D. Ribas, N. Palomeras, P. Ridao, M. Carreras, A. Mallios, Girona 500 AUV: From survey to intervention, IEEE/ASME Trans. Mechatronics 17 (1) (2012) 46–53.
[63] D. Youakim, P. Ridao, N. Palomeras, F. Spadafora, D. Ribas, M. Muzzupappa, Autonomous underwater free-floating manipulation using moveit!, IEEE Robot. Autom. Mag. (2017).
[64] MoveIt! Benchmark tutorials, 2013. http://docs.ros.org/kinetic/api/moveit_tutorials/html/doc/benchmarking_tutorial.html.
[65] A. Dornbush, SMPL library, 2016. https://github.com/aurone/smpl.
[66] D. Youakim, A modified version of MoveIt! Benchmarking, 2017. https://github.com/dyouakim/moveit.
[67] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: An efficient probabilistic 3d mapping framework based on octrees, Auton. Robots (2013). Software available at http://octomap.github.com.
[68] M. Moll, I.a. Sucan, L.E. Kavraki, Benchmarking motion planning algorithms: An extensible infrastructure for analysis and visualization, IEEE Robot. Autom. Mag. 22 (3) (2015) 96–102.

Dina Youakim received her Master degree in Computer Vision and Robotics from the Heriott–Watt& UdG in 2015, where she also received the best Master student award. She joined the Underwater Robotics Research Center (CIRS) as a Ph.D. candidate under the supervision of Dr. Pere Ridao. Her research interests are focused on autonomous motion planning for manipulation. Before studying her master, she has worked on various multinational IT companies, where she gained a wide experience in software development for automotive and medical applications. Moreover, she has worked as a researcher in LITIS, Rouen — France, implementing a SLAM solution for indoor navigation. In addition, as part of her Ph.D., she is currently doing a secondment at the Robotics Institute – Carnnegie Mellon university, in order to evolve her work on motion planning.

Pere Ridao who received the Ms.C. degree in computer science in 1993 from the Technical University of Catalonia, Barcelona, Spain, and the Ph.D. degree in computer engineering in 2001 from the University of Girona, Spain. His research activity is mainly focused on underwater robotics in research topics such as intelligent control architectures, UUV modeling and identification, simulation, navigation and mapping, Mission Control and real-time systems. He joined the Institute of Informatics and Applications, University of Girona, in September 1995. Currently, he is an associate professor with the Department of Computer Engineering of the University of Girona. Currently he is involved in national projects (RAUVI or ARCHROV) and European projects (FP7 STREP TRIDENT, PANDORA, MORPH, EUROFLEETS 2, amongst others) about underwater robotics and some technology transference projects (INSPECSUB). Dr. Ridao is the chair of the IFACs Technical Committee on Marine Systems and member of the editorial board of Springers Intelligent Service Robotics journal.

4

Multi-Representation, Multi-Heuristic A* Motion Planning for a Free-Floating UVMSs in Unknown Environment

In this chapter, we focus on our proposal of a new motion planning algorithm to meet the previously defined requirements (consistency, real-time response, efficiency and safety). The Multi-heuristic, multi-representation A* falls under the search-based planning group. It generates deterministic trajectories that consider the shortest collision-free path to the goal. In addition, it exploits a characteristic of our system, which is its loose coupling nature between the base and the arm. The approach takes profit of this characteristic to efficiently move one or the other keeping the shortest, safe path as the metric. It is worth noting, that decoupling the base and arm motion has been previously introduced, in simulation, in [42], where a control-based approach was proposed to distribute the control input between the arm and the base depending on their nature and on the phase of the mission (approach vs. intervention phase).
The proposed approach has been compared to other common available planning algorithms to validate its efficiency. Moreover, it has been validated in a two-phase intervention mission in a known apriori environment, in simulation. Lastly, it has been demonstrated in a water tank, to turn a valve in an unknown environment. This work has recently been submitted to the following journal and is currently under review:

Title: Multi-Representation, Multi-Heuristic A* Search-based Motion Planning for a Free-floating Underwater Vehicle Manipulator System in Unknown Environment
Authors: **D. Youakim**, P. Cieslak, A. Dornbush, A. Palomer, P. Ridao, and M. Likhachev
Submitted to: Journal of Field Robotics
Quality index: JCR2017 , Journal of Field Robotics, Q1 (6/26)

Multi-Representation, Multi-Heuristic A* Search-based Motion Planning for a Free-floating Underwater Vehicle Manipulator System in Unknown Environment

Dina Youakim
Computer Vision & Robotics Institute
University of Girona
dina.isaac@udg.edu

Patryk Cieslak *
Computer Vision & Robotics Institute
University of Girona
patryk.cieslak@udg.edu

Andrew Dornbush
The Robotics Institute
Carnegie Mellon University
andrewpd@andrew.cmu.edu

Albert Palomer
Computer Vision & Robotics Institute
University of Girona
albert.palomer@udg.edu

Pere Ridao
Computer Vision & Robotics Institute
University of Girona
pere.ridao@udg.edu

Maxim Likhachev
The Robotics Institute
Carnegie Mellon University
maxim@cs.cmu.edu

Abstract

A key challenge in autonomous mobile manipulation is the ability to determine, in real-time, how to safely execute complex tasks when placed in unknown or changing world. Addressing this issue for Intervention Autonomous Underwater Vehicles (I-AUVs), operating in potentially unstructured environment is becoming essential. Our research focuses on using motion planning to increase the I-AUVs autonomy, and on addressing three major challenges: 1) Producing consistent deterministic trajectories, 2) Addressing the high-dimensionality of the system and its impact on the real-time response, and 3) Coordinating the motion between the floating vehicle and the arm. The latter challenge is of high importance to achieve the accuracy required for manipulation, especially considering the floating nature of the AUV and the control challenges that come with it. In this paper, we extend our previous work relying on exploiting the loose coupling between the AUV and the arm, by applying MR-MHA* (Multi-Representation, Multi-Heuristic A*) search-based motion planning to control a simulated I-AUV operating in a known apriori offshore environment. We further developed the previously proposed algorithm to control a real system (GIRONA 500 I-AUV) doing intervention on a sub-sea pipe infrastructure mock-up located in a water tank. Moreover, the experiment is performed in unknown environment, being discovered by the robot during the mission. we relied on an underwater laser scanner to provide real-time point-cloud updates of the sensed world. The results show the success and efficiency of our approach to meet the desired behavior, as well as the ability to adapt to unknown environments.

*Patryk Cieślak has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement no. 750063

5

Results and Discussion

The content of this chapter is split into two parts. Initially, the results previously presented is summarized with key points pointed out with the following structure: First we present the preliminary intervention demonstration using motion planning from Chapter 2 in Section 5.1. Then, the experimental analysis of the state of the art motion planning techniques in Section 5.2. At the end of this summary,in Section 5.3, we present a representative intervention mission in simulation, followed by a water tank experimentation validation, and finally an additional simulation analysis of the proposed algorithm "Multi-Representation, Multi-Heurisitc A*".

Lastly, in Section 5.4, in order to criticize our proposed solution, we expand the previously obtained results by a comparison of our approach against five of the most common motion planners. We used three of the previously defined benchmarks, in addition we introduced two more, in order to show when and why the new approach performs well.

5.1 Motion Planning for an UVMS using *MoveIt!*

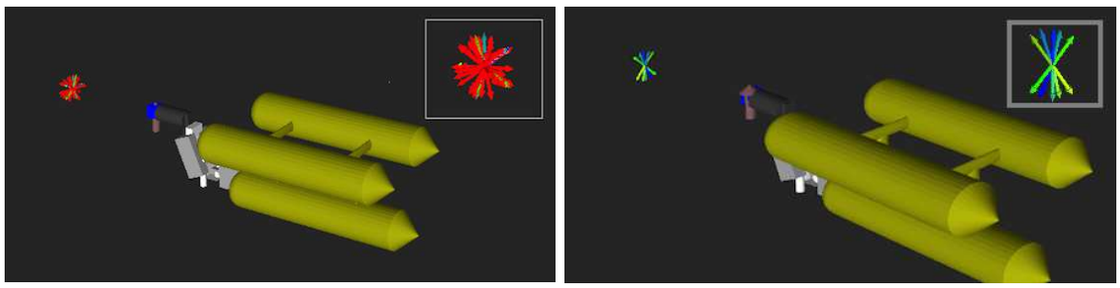
The focus of this initial step was the evaluation of the use of *MoveIt!* motion planning capabilities to control *Girona500 AUV & ECA Arm* and demonstrate beyond-state-of-the-art tasks.
As a first output, we presented the modeling of the 8 DoF AUV-Arm as a single system using the standard Universal Robot Description File (URDF) format, in addition to the Semantic Robot Description File (SRDF) for the collision representation (more details can be found in [43]). At this stage, we wanted to validate the idea of using motion planning, the choice of the planner was not the focus; for that we used the default planner provided by *MoveIt!*, the "RRT-Connect". To experimentally validate the use of motion planning, we demonstrated two free-floating intervention tasks in a water tank: "Valve Turning" and "Connector plug/unplug", using a subsea panel mock-up with four T-shaped valves and a hotstab type of connector. The working space included two types of known apriori obstacles:

1. The walls of the water tank and the panel.
2. Virtual obstacles (in the valve turning scenario).

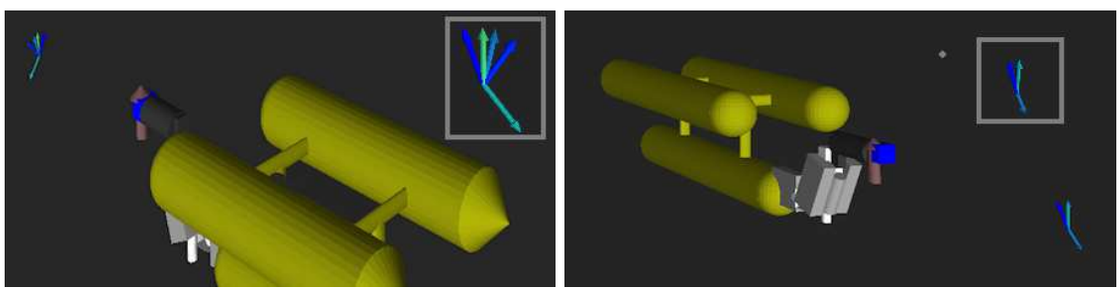
5.1.1 Valve Turning:

In this experiment, a fixed V-shaped gripper matching the valve shape was used. The intervention task was carried out through four phases as follow:

- *Detection:* as a first step, the purpose is to locate the panel and establish a visual contact to enable visual navigation with respect to the panel [44].
- *Inspection*: for homing the AUV to the panel. It enables high-accuracy visually based navigation, providing a good estimate of the valve pose. The EE is guided to a Cartesian-space-defined pose located in front of the panel.
- *Grasp planning:* To perform the intervention, one must specify the desired valve pose. We performed a workspace analysis that involved sampling the end-effector orientation space ψ, θ, ϕ, with a sampling step of 15° (the whole space shown in Fig. 5.1a with the arrow pointing in the +ve X-axis of the end-effector), then a different filters were applied:
    1. Filter out those orientations that do not satisfy the task constraints: respecting the approach direction (filtering out the half sphere pointing on the negative approach direction), and allowing a variability of ±45° in ψ, θ, ϕ with respect to the computed grasp orientation (see Fig. 5.1b).
    2. Filter out those infeasible orientations that violate the system kinematic constraints and joint limits (see Fig. 5.1c).
    3. Filter out those orientations resulting in a colliding configuration (see Fig. 5.1d for the environment in Fig. 5.2, where the best orientation is marked in yellow arrow).



(a) Sampled EE Orientation Space **(b)** Task-Valid EE Orientation



(c) Kinematically-Valid EE Orientation **(d)** Environment-Valid EE Orientation

Figure 5.1: Workspace Analysis Output at Various Stages

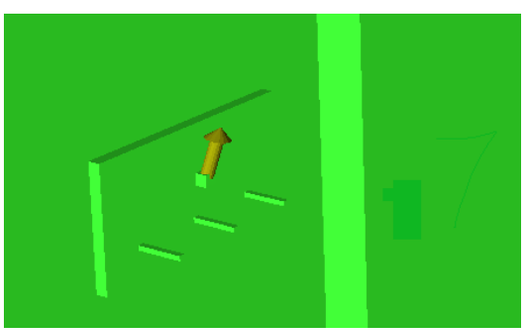


Figure 5.2: The Best (highest MI) End-Effector Orientation for the Valve Turning in the presence of Obstacle close to the Panel.

- *Valve turning:* involving the following two steps:

  1. Approach: aiming at getting close to the panel at a predefined distance, in the direction normal to the panel, guaranteeing the end-effector pose before performing the intervention.

  2. Grasp & Turn: The method is based on the MoveIt! pick pipeline. The frame of reference for the approach is the valve, and the approach direction is normal to the subsea panel Z-axis. Because there is no gripper to open/close, the grasp closure pose is used to perform the turn action. The wrist joint is specified as the posture joint, and the turning degree is provided as the joint value.

5.1.2 Connector Plug/Unplug:

We used a model inspired by a hot-stab connector, similar to the one used in the oil and gas industry. To perform this task, a newly developed three-finger hand described in [45], was used. The task starts with the connector plugged in, then requesting the robot to unplug it, move away, and finally plug it back. The intervention procedure is similar to the procedure previously described but with the valve-turning step replaced by:

- *Unplug:* The hand is requested to grasp the connector by closing the fingers. Similar to the valve turning, the *MoveIt!* pick pipeline is used. First, an approach direction perpendicular to the plane of the connector is specified, and the pre-grasp pose is chosen to open the hand before grasping. The connector pose corresponds to the pick pose, and the grasp closure pose is used to close the hand after holding the connector. Once the grasping is successful, the unplugging is accomplished by performing a retreat backward. The retreat direction is specified along the Z-axis in the connector frame.

- *Plug:* The *MoveIt!* place pipeline is used to plug the connector to its original location. Once the connector reaches its original location, the hand is opened, leaving the connector plugged in.

For each demonstration, we showed the planned versus the executed trajectories for each DoF $x, y, z, \phi, q1, q2, q3, q4$, to demonstrate how the system followed the plan as well as the performed grasping (i.e. 90° valve turn or the hand opening/closing) within the predefined tolerances [0.06 cm, 0.06 cm, 0.02 cm, 0.06 rd, 0.075 rd, 0.075 rd, 0.075 rd, 0.3 rd] for each DoF respectively. The resulting end-effector error corresponding to these tolerances, is variable depending on: tthe AUV and the arm controllers accuracy, as well as the localization accuracy. Given the fact that the intervention was performed facing the panel, that was simultaneously used for localization, the resulting end-effector error was minimal (as can be appreciated in Fig.9(d) in Chapter(2)), allowing the success of the mission. The following states the end-effector pose error for both the inspection and valve turning, the detection was skipped as the goal was to reach a given joint configuration:

- For the inspection point the end-effector pose error is [0.00085 cm, 0.03101 cm, 0.08292 cm, −0.23 rd, 0.01 rd, 0.88 rd]

- For the valve turning point the end-effector pose error is [0.00031 cm, 0.01594 cm, 0.00092 cm, 0.16 rd, 0.00327 rd, 0.18 rd]

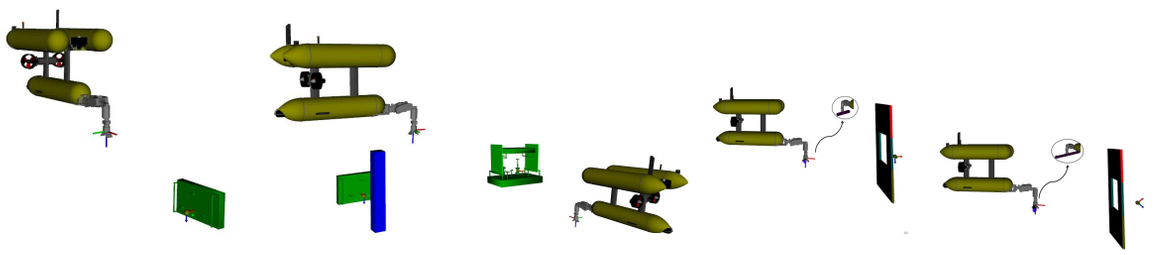
In addition the 3-D end effector trajectory was shown to verify the obstacle avoidance (Fig. 9,10 Chapter 2).

5.2 In-depth Analysis of Motion Planning for Mobile Manipulation

At this stage of our work, we tried to answer the tough question of “which planner to choose” given an application or set of requirements for an autonomous mobile manipulator. We reviewed the state of the art of the most common approaches, and presented a set of benchmarks with the aim to provide not only a theoretical review but also a qualitative/quantitative comparison of the algorithms. Our objective was to provide an insight

of their performance with respect to different metrics. Even though the results are based on our UVMS, they can be extended to terrestrial and aerial robots.

5.2.1 Benchmarks



(a) BM#1: Simple Valve Turn

(b) BM#2:Valve Turn With Obstacle

(c) BM#3: Cluttered Environment

(d) BM#4: 50cm Stick Manipulation

(e) BM#5: 1m Stick Manipulation

Figure 5.3: Benchmarks with the I-AUV Start Configuration & the goal Pose ($x, y, z \rightarrow$ Yellow Sphere, $\phi, \theta, \psi \rightarrow$ RGB Axis)

We have carefully created five benchmarks to demonstrate the strengths and the weaknesses of each approach. Moreover, they reflect representative set-ups/environments in which a mobile manipulator might be placed:

- ***BM#1, BM#2:*** correspond to subsea panel intervention, similar to "table-top" manipulation for terrestrial robots. BM#1 is free of obstacles, while BM#2 includes static obstacle close to the manipulation object.
- ***BM#3:*** represents a cluttered environment, where various obstacles are scattered around the manipulation object.
- ***BM#4, BM#5:*** represent narrow passage manipulation.

5.2.2 Planners

Three motion planning families have been addressed: 1) sampling-based, 2) optimization-based, and 3) search-based. For each family, our choice aims to cover the basic approach in addition to the various features provided by the planning family (e.g. lazy collision checking for both sampling-based and search-based), with a total of 17 planners (listed in Table 5.1). For the search-based planners, the different combinations of the different core components: search, graph and heuristic were compared. In order to reduce the number of results presented, and because for some scenarios a given combination was not showing improvement; we omitted some search-based combinations and ended up with the following set-up: The combination of ARA^* search, CS and WS graph, and heuristics ($EUCLID$ and BFS) was tested for all benchmarks. The lazy-collision checker version was tested for BM#1,BM#2,BM#3, while the multi-frame BFS was tested for BM#4, BM#5.

5.2.3 Metrics

For each benchmark, all the planners were compared along M runs, with respect to four standard metrics:

Categories	Planners	Benchmarks
Sampling-based	RRTConnect	All
	SBL	All
	STRIDE	All
	RRT*	All
	PRM*	All
Optimization-based	STOMP	All
	CHOMP	All
Search-based	ARA*.BFS.CS	All
	ARA*.EUCLID.CS	All
	ARA*.BFS.WS	All
	ARA*.EUCLID.WS	All
	LARA*.BFS.CS	BM#1,BM#2,BM#3
	LARA*.EUCLID.CS	BM#1,BM#2,BM#3
	LARA*.BFS.WS	BM#1,BM#2,BM#3
	LARA*.EUCLID.WS	BM#1,BM#2,BM#3
	ARA*.MFBFS.CS	BM#4 & BM#5
	ARA*.MFBFS.WS	BM#4& BM#5

Table 5.1: Planners Choice & Benchmarks Coverage

- **Planning Time** the time needed by the planner to report a collision-free trajectory.
- **Path Clearance** the minimum distance to obstacles along each way-point of the computed trajectory.
- **Path Smoothness** the measure of smoothness of transitions between successive way-points of the computed trajectory.
- **Success Rate** the percentage of runs (out of M run) where the planner succeeded to report a collision-free trajectory.

Details on the definition and computation of each metric can be found in Chapter 3. In addition, a complementary qualitative assessment graph was introduced, named as "**Distance to objects colour map**", where the purpose was to visually assess the behavior of the planner with respect to obstacles. It also allows perceiving the change of planner behavior with respect to environment complexity (for more details about the plot data see Chapter 3). It is worth noting that, as a difference with respect to the rest of the plots, this plot illustrates the results of only one of the M runs, the one corresponding to the longest path, ignoring all non-collision free paths.

5.2.4 Analysis & Guidelines:

Through the previously defined benchmarks and metrics, we can conclude general outlines for each planning group:

- **Sampling-based Planners:**
  1. *Non-optimal Strategies:* they fit the highly time-constraint applications well, as they demonstrated the fastest planning time across all benchmarks, at the

expense of other quality metrics like path length, consistency, and natural motion generation. Moreover, for the uniform sampling, the planner has tendency to generate more samples in free regions, leading in general to high clearance.

2. *Asymptotically-optimal Strategies:* they compromise a cost function (e.g. path length) with the planning time. In average, they showed higher consistency and more human-like motion than their non-optimal counterpart, with the least performance for the narrow passage scenarios.

An additional advantage of the sampling-based planners provided by the OMPL library, is that they are well tweaked for a wide range of applications, and their default parameters, sampling-strategy and distance metric work quite well. This leaves room for improvement to overcome their weaknesses using application-oriented parameters and sampling strategies.

- **Optimization-based planners:** overall their performance is average among the others, being highly affected by the benchmark complexity. Their strongest advantage, is that they solve the motion planning as an optimization problem, which allows the combining of complex cost functions. On the other hand, they suffer technical difficulties when it comes to implementation due to the complexity of their mathematical background.

- **Search-based planners:** showed an advantage in generating optimal trajectories in terms of path length, with more natural motion and higher consistency, while still meeting average time performance. The planning time is highly affected by the number of motion primitives and the resolution chosen. Increasing the motion primitives would lead to better coverage at the cost of spending more time searching the graph. A major disadvantage is their dependency on parameter tweaking (i.e. resolution & motion primitives). In particular, for the manipulation lattice, designing the motion primitives requires significant effort and a considerable knowledge of the environment. Using convenient heuristics, it would be possible to adapt them to deal with more complex objective functions (e.g. the manipulability index).

Unfortunately, there is no single technique that would perfectly fit, out of the box, all possible applications, systems and environments. Nevertheless, a valid open question would be: "Which motion planning technique to start with?". Table 5.2 summarizes the planners categories versus the various features/requirements for a typical motion planning problem. Whenever a feature is marked possible, it means that it showed average performance against this feature in our tests; or, that simple parameter tuning would enhance its performance against such a feature. A feature is considered as not supported, if the nature of the planner does not (without major changes) support it.

5.2.5 UVMS Requirements

For our next step, we are considering the case of an UVMS for underwater free-floating manipulation with the following requirements:

1. Fast planning response to allow real-time execution.

2. Cluttered obstacles, possibly leading to narrow passages in some cases.

Planners Categories	Real-Time Response	Cluttered	Narrow Passage	Consistency	Human-Like Motion	Fast Re-plan	Clearance	Objective Function
Non-Optimal Sampling	✔	possible	✗	✗	✗	possible	✔	✗
Optimal Sampling	possible	possible	✗	✗	possible	possible	✔	✔
Heuristic anytime A* workspace	possible	possible	✔	✔	✔	possible	possible	possible
Heuristic anytime A* manipulation	possible	possible	✔	✔	✔	possible	possible	possible
Optimization-based	✗	possible	✗	✗	possible	possible	possible	✔

Table 5.2: Planners vs. Environment Features

3. *A priori* unknown environment, this would require real-time re-planning capabilities to deal with the incremental map-building process.

4. Generated trajectories with a combined optimization function: short path, good manipulability index, optimizing the redundant DoFs motion (e.g. during the intervention, it might be preferable to move the arm and not the AUV).

5. Consistency is preferable given the incomplete knowledge of the environment. It is is in particular useful in applications requiring human-robot interaction; and for those autonomous systems that operate under human supervision as the robot behavior would be foreseen for the observer.

5.2.6 Planner Selection

Considering the previously detailed requirements and the outcome of our analysis (as summarized in Table 5.2, we had two options to proceed with; either asymptotically-optimal sampling-based planning or search-based planning. The asymptotically-optimal family showed promising behavior specially in terms of planning time. On the other hand, the search-based had a key feature due to its deterministic nature, "the consistency"; also it demonstrated above average performance in all basic features. Moreover, sampling-based planners have been widely investigated for mobile manipulation, while the search-based remained less explored. For these reasons, we decided to build our solution under the search-based umbrella.

5.3 Exploiting System Loose Coupling in Search-based Motion Planning

Based on the survey results presented in the previous section, both the workspace and manipulation lattices have similar benchmarking results, but only the manipulation lattice can exploit the "Loose Coupling" nature of the UVMS. In this lattice, the search occurs in the configuration space, allowing to decouple to some extent the AUV vs. the

arm motion. Based on this observation, in Chapter 4 we proposed a manipulation lattice search-based motion planning algorithm that exploited this common mobile manipulators characteristic denoted "Loose Coupling". As the results will show later, this characteristic allowed not only to coordinate the AUV and the arm motion, but also to increase the planning efficiency in terms of path length and time in some cases.
The validation of the proposed algorithm followed three stages: first, an interventions similar to those used in the off-shore industry were performed in simulation in a known apriori environment. Later, a water tank demonstration was performed. For this experiment, the environment was assumed to be unknown and a laser scanner was integrated to sense the robot surroundings while moving. Finally, an exhaustive analysis of the planner behavior was carried out in simulation to give further insights about the proposed method.

5.3.1 Representative Intervention Mission Simulation:

This step consisted of a simulation of a representative underwater manipulation mission in a typical sub-sea structure like those used by the Oil&Gas industry. An octomap of a real sub-sea structure is used as a known apriori map. The mission consisted of two common intervention tasks: 1) Pipe Inspection and 2) Connector unplug/plug, with the corresponding goals labeled $B(1)$ *initial,intermediate,final* and $C(1)$ *initial,final* as in Fig. 5.4.1.
Fig. 5.4 shows some of the I-AUV configurations during the execution with the start configuration shown in Fig. 5.4.1. The inspection intervention consisted of three tasks (i.e. three planning queries): First to pick the bar from its initial position (Fig. 5.4.2, Fig. 5.4.3) with planning Time = 4.8 s, second to inspect pipes on the upper part of the structure (Fig. 5.4.4, Fig. 5.4.5) and it took 6.8 s to plan this task. Finally, moving to the side of the structure holding the bar (Figures Fig. 5.4.6, Fig. 5.4.7) with a planning time of 14.3 s. After terminating the inspection, the bar is placed on the structure to start the new intervention.
For the second intervention, the connector had to be picked and un-plugged (Fig. 5.4.8, Fig. 5.4.9), and this phase took 1.6 s to plan. At last, the connector was plugged in one of the destined holes for connectors (as shown in Fig. 5.4.10, Fig. 5.4.11), with 1.7 s planning time. We can conclude that the planner is able to cope with complex tasks, given that the overall mission (i.e. 5 tasks) planning time is almost 30 s, which would enable real-time performance in real environments.

5.3.2 Water Tank Demonstration:

The water tank test environment is shown in Fig. 5.5, where the yellow structure is equipped with multiple valves, and the purpose of the mission was to turn the lower left (blue) valve where initially the environment is unknown and has to be sensed online by the I-AUV (see Fig. 5.5)

In order to carry out the valve turning in unknown environment, four stages were followed:

- *Exploration Phase:* where the purpose is to locate the pipe structure position with respect to the world frame whose origin was defined at the right front corner of the water tank. Initially, the AUV is placed at this origin. Then, different detection points defined in the configuration space are sent to the planner and executed until

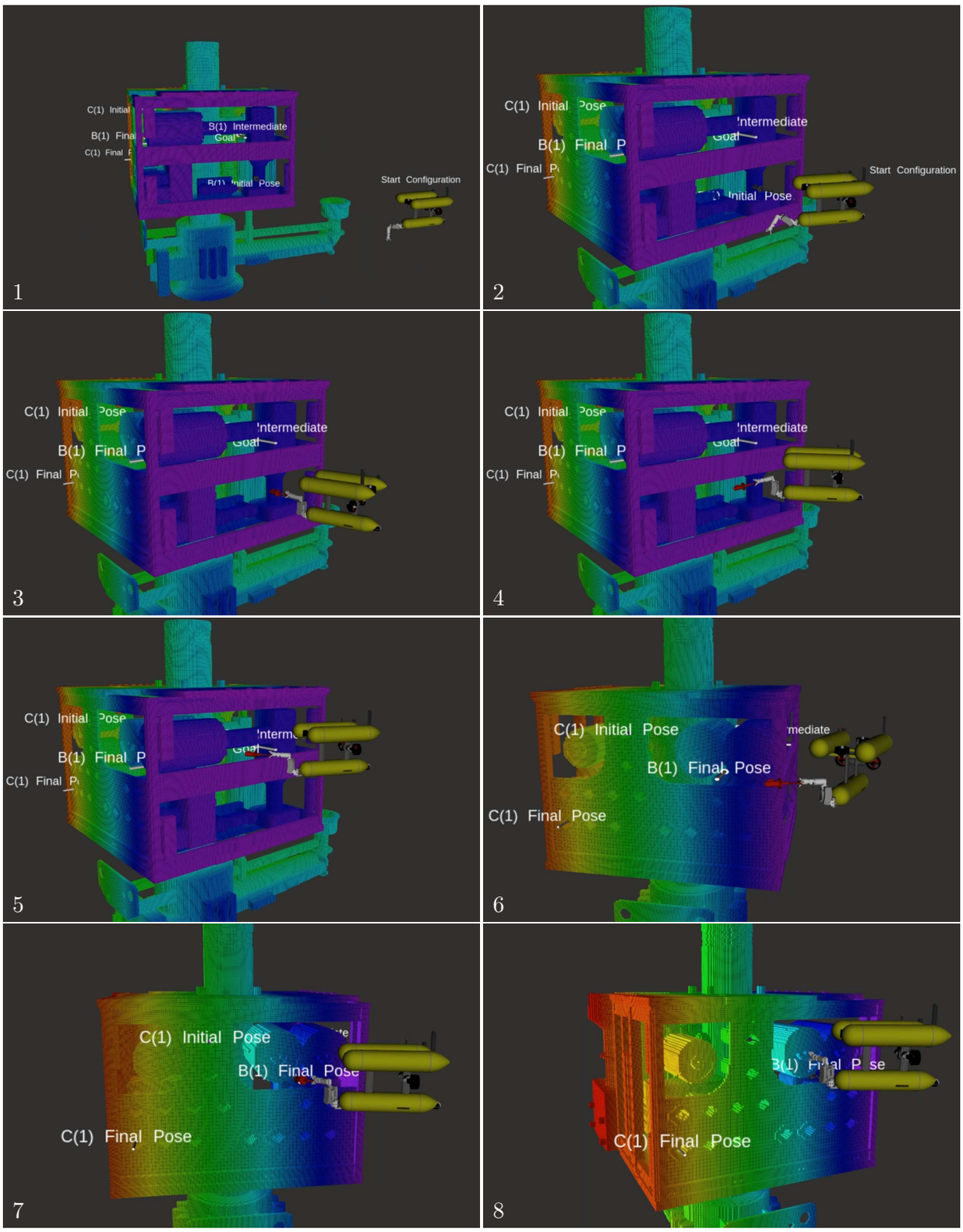


Figure 5.4: Parts of the Different Trajectories during the 5 Phases of the Mission.
$1 \rightarrow 7$: Pipe Inspection Mission
$8 \rightarrow 12$: Connector unplug/plug

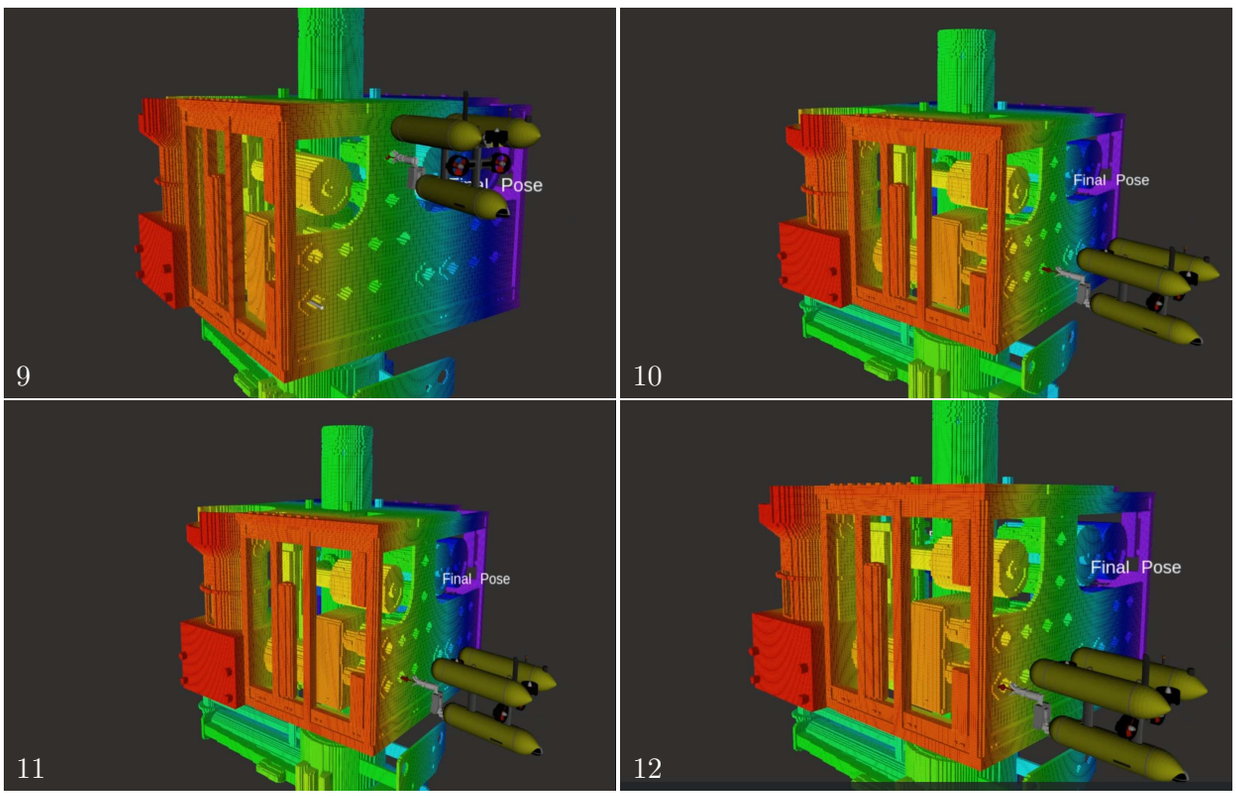


Figure 5.4: Parts of the Different Trajectories during the 5 Phases of the Mission.
$1 \rightarrow 7$: Pipe Inspection Mission
$8 \rightarrow 12$: Connector unplug/plug

the two corner markers are detected (known by their IDs). Those points mainly move the AUV in the X-Y plane. When the markers are observed, their relative poses with respect to the robot can be computed. Then, compounding these poses with the robot pose, the markers poses in the world frame can thus be computed. Once available, the pose of the structure in the world frame can also be computed. Even the pose of the structure is not required to complete the mission, it is useful to compute approximately an approach pose to the vale to turn. Moreover, knowing the pipe pose helps to interpret the results, since it allows us to compare the robot trajectory with the actual position of the pipe.

- *Homing Phase:* the aim of this step is to move the I-AUV in a position that enables it to visualize the pipe well in order to carry out the desired intervention. This phase is similarly carried out by planning the I-AUV to a configuration space, where the goal is computed with respect to the pipe center previously detected.

- *Approach Phase:* given the computed pipe center, and knowing the geometry of the pipe, the position of each valve can be computed. In this phase, the objective is to approach the valve to be turned (lower-left valve in Fig. 5.5). Given the computed valve pose, the end-effector goal is sent to the planner with an offset in the Z-axis, in order to place the end-effector in an approach position on the top of the valve. This offset (= 40 cm in the experiment) was estimated based on the FOV of the end-effector camera to ensure full view of the valve when reaching it from the top, in

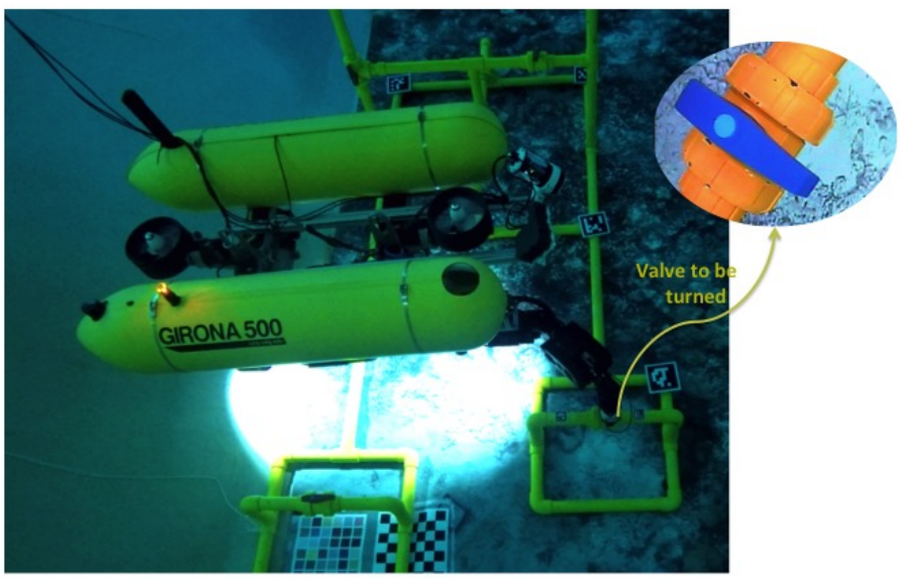


Figure 5.5: Test Environment in a Water Tank with the blue valve to be turned

order to re-compute and correct the estimated valve pose using a vision-based algorithm. During this phase, the planner has no prior knowledge of the pipe existence. After computing the first plan, the laser starts scanning the environment, providing updates to the octomap, then the planner acts accordingly and re-computes alternative collision-free plans each time the occupied octomap voxels obstruct the current plan solution. The process is repeated until safely reaching the top of the valve.

- *Visual Servoing:* the last phase of the mission has the purpose of turning the valve. Due to the various inaccuracies (AUV localization, camera-calibration, pipe model geometry vs. reality), it is not possible to use the initial estimated valve pose and plan directly to it. Thus we rely on a visual servoing mechanism to perform the turning, which is a combination of Task-Priority based control and vision-based valve detection. The vision-based algorithm provides pose updates while the I-AUV is moving towards the valve in a closed-loop feedback. The valve orientation updates are provided as long as the whole valve is seen in the image, while the position is more frequently updated as long as the blight-blue circle in the valve center (Fig. 5.5) is seen, enabling better positioning in the last few centimeters of getting close to the valve. The sequence is performed by sending a series of desired end-effector poses to the Task-Priority. First, a closer approach is computed from the detected valve position with an offset in Z-axis, and same detected orientation. Next, the end-effector lands on the valve by moving to the exact valve detected pose. In the third stage, after the I-AUV is locked on the valve, a pose rotated by the desired angle (70° in the experiment) around the Z-axis is generated. Lastly, an end-effector pose is generated to retreat the I-AUV back to the initial position.

For the motion planning stages, in Chapter 4 we showed the planning vs.actual trajectories for the 8 DoF, as well as for the end-effector trajectories. These plots demonstrated how the loose coupling of the base vs. arm resulted in moving the AUV until close to the goal region, then the arm complemented the motion when close to the goal. Moreover, it showed the sensing and re-planning capabilities of the system once an obstacle invalidates the already computed plan. Additionally, the planning time, system clearance, path length were reported to validate the real-time response of the system, and its efficiency in terms of the distance traveled and safety.
Similarly, for the visual servoing part, the plots demonstrate how the different activated task in the TPRC hierarchy were respected:

1. The arm joint limits were respected.
2. Both the AUV and the arm yaw were restricted to the already planned configurations to ensure safe operation given that the planned configuration is collision-free.
3. The end-effector pose followed the valve detected pose and performed the desired turn.

5.3.3 In-Depth Motion Planning Analysis in Simulation:

At this step, we performed a thorough analysis of two characteristics of the motion planner: 1) The clearance cost function, and 2) The AUV vs. the arm motion loose coupling.
To serve this purpose, we created a simulation scenario using the same pipe structure as in the water tank experiment, but with the target of turning the valves [V#1, V#4, V#2, V#5, V#8] (labeled in Fig. 5.6).

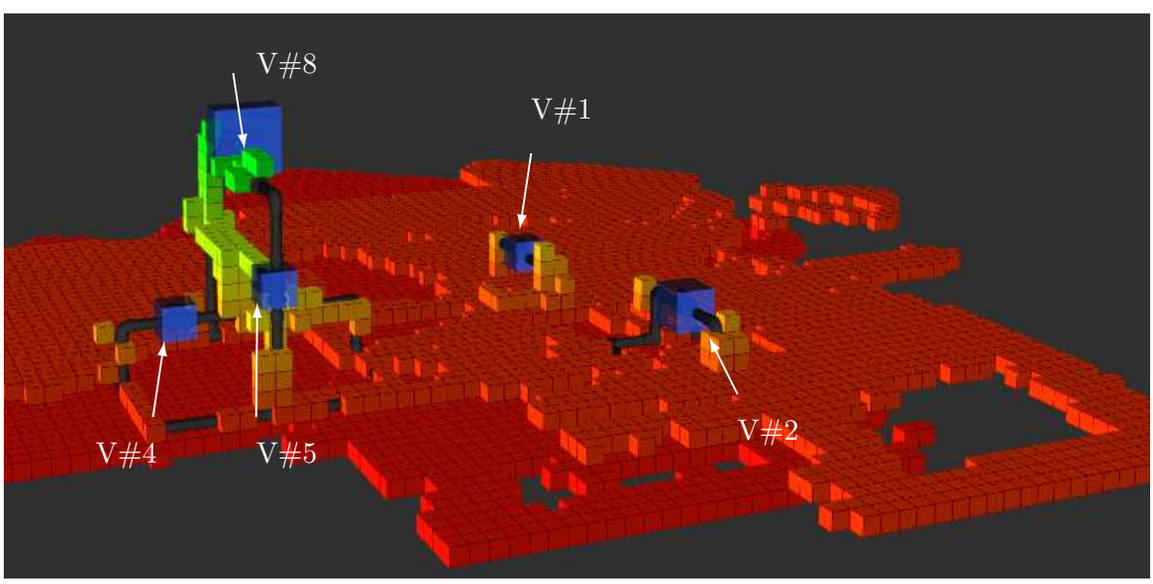


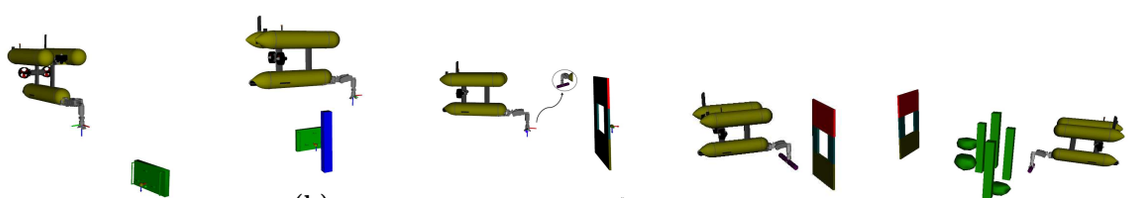
Figure 5.6: Pipe Simulation Environment with the octoamp & Valves to be turned.

Each manipulation consisted of an "Approach" and a "Turn". The scenario has been repeated three times with different clearance threshold $(0, 0.1\text{ m}, 0.5\text{ m})$, and each time the environment was assumed unknown.
The evolution of both the g and the h functions vs. the distance to obstacles, for various

clearance thresholds, were reported in Chapter 4 showing the penalty imposed on the states violating the set clearance threshold. The minimum distance to obstacle at each way-point of the generated trajectories, demonstrated the variability in the plans given the clearance threshold. It also showed the difficulty of respecting the 0.5 m at all moments, specially when moving to/from a valve.
In order to analyse the motion loose coupling, we reported plots showing the percentage of either the arm or the AUV motion with respect to the position of the corresponding way-point in the plan. We could deduce that the arm was used either around the goal region or to maneuver around obstacles, else the base was taking over the motion. Lastly, numerical analysis was reported for the different tested clearance thresholds: graph search time, clearance overhead, overall planning time, path length, and number of expansions. From these numbers it can be shown the effect of increasing the clearance threshold on the search time as well as on the path length, with the worst performance reported for the 0.5 m clearance.

5.4 MR-MHA* Planner Evaluation

In order to evaluate the performance of our algorithm, we used five benchmarks (shown in Fig. 5.7), similar to the ones previously reported in Chapter 3. BM#1, BM#2 and BM#3 are identical to those in Section 5.2. BM#4 simulates the task of moving an elongated object (stick) through a window-like opening. BM#5 performs the same operation but going through a space cluttered with obstacles. The stick length in BM#3 and BM#5 is 50 cm, which is smaller than the window width. For BM#4 the stick is 1m length (> window width), with same initial I-AUV configuration as in BM#3, but with the window closer to the I-AUV, making it more challenging to reach the goal by passing the stick through the window.



(a) BM#1: Simple Valve Turn

(b) BM#2:Valve Turn With Obstacle

(c) BM#3: Stick 50cm Manipulation

(d) BM#4: Stick 1m Manipulation Close-Window

(e) BM#5: Stick 50cm with Scattered Obstacles

Figure 5.7: Benchmarks representing various test environments used for simulation validation

We compared our technique against five other common planners, three of which are sampling-based (*RRT* [46], *RRTConnect* [47], *RRT** [48]), one optimization-based planner (*STOMP* [49]), and the basic search-based planner (*ARA**). The evaluation consisted of comparing four metrics based on 20 runs of each planner in each of the five benchmarks: planning time, path length, success rate, and consistency. The reported numbers are the mean over the 20 runs. Each planner was allowed at most 30 s for planning. In addition, the reported solutions by each planner have been post-processed (e.g. shorcutting), and the reported planning time includes the post-processing step. It is worth noting that for both *RRT** and *ARA** results are reported for the first solution found without further

improvements to ensure fair comparison with other planners.

The results are shown in Tables 5.3, 5.4, and 5.5:
From the first table, our approach results in shorter path length (measured as a combination of the L2 norm of both translational and rotational joints) compared to all other planners, with the exception of the first benchmark where the difference to the shortest path is slight (0.37). A reason for this behavior, is the simplicity of the scenario with wide obstacle-free areas around, where all planners after post-processing result in similar solutions. While for the second benchmark, the difference with respect to $ARA*$ is almost negligible (0.02), not affecting the conclusion that the proposed approach generally performs better.
Concerning the planning time, as expected, the non-optimal sampling based outperformed; while our planner had comparable performance to the asymptotically-optimal sampling, and generally outperformed the ARA^*. A drawback is seen in BM#3 where ARA^* is able to find the direct path through the window faster than the proposed approach, and expands much less nodes (as shown in Table 5.5).
On the other hand, with more challenging benchmarks, the success rate of some sampling-based planners decreases, whereas the ARA^* showed consistent failure in BM#4 & BM#5. This proves the improvement of our proposed approach in the scenarios where the direct path to the goal is blocked by obstacles or not feasible, and moving the base alone is necessary to explore other areas.
Fig. 5.8 shows the trajectories found for BM#4 & BM#5 by our approach. The states in red have been found by searching the base representation and those in blue have been found by searching the arm representation. It can be appreciated the base support to get around the blocked areas and advancing the search forward finding its way to the goal faster.
To perform a consistency test, in each run out of N (= 20), the start configuration per benchmark has been randomly modified in the exact same way for all the planners, and within a range of (20 cm,5°) for the translational and rotational joints respectively, and then the test was performed. Finally the reported numbers are the mean of the *Frechet Distance* between the combination of the M successful computed motion (i.e. the joint-space plan), where M is the number of successful runs (<= 20). Furthermore, the success rate is reported as different failures were encountered due to the start configuration change. In table 5.4, "bold" is used to denote the highest consistency (lowest Frechet distance). As expected, due to their deterministic nature, search based planners demonstrated higher consistency (i.e. lower consistency index) with the $MR-MHA^*$ outperforming. Differently, STOMP shows an incredibly low index for BM#5 but only with a 20% success rate.

Planner	BM#1			BM#2			BM#3			BM#4			BM#5		
	T	L	%	T	L	%	T	L	%	T	L	%	T	L	%
RRT	0.087	5.5	100	0.84	8.2	100	0.068	24.12	100	**0.045**	18.74	95	5.6	11.15	65
RRTConnect	**0.02**	**5.43**	100	**0.036**	13.8	100	**0.03**	19.8	100	0.05	17.94	100	**0.035**	16.4	90
RRT*	0.06	5.68	100	1.8	15.1	100	0.23	18.48	100	0.05	22.7	100	3.34	14.3	100
STOMP	2.68	6	100	3.65	6.25	100	3.38	4.45	25	F	F	0	23.8	11.16	10
ARA*	0.087	7.1	100	0.12	**5.88**	100	0.25	6	100	F	F	0	F	F	0
MR-MHA*	0.12	5.9	100	0.15	5.9	100	2.43	**4.2**	100	0.94	**14.98**	100	0.39	**9.56**	100

Table 5.3: Planning Time, Path Length & Success Rate

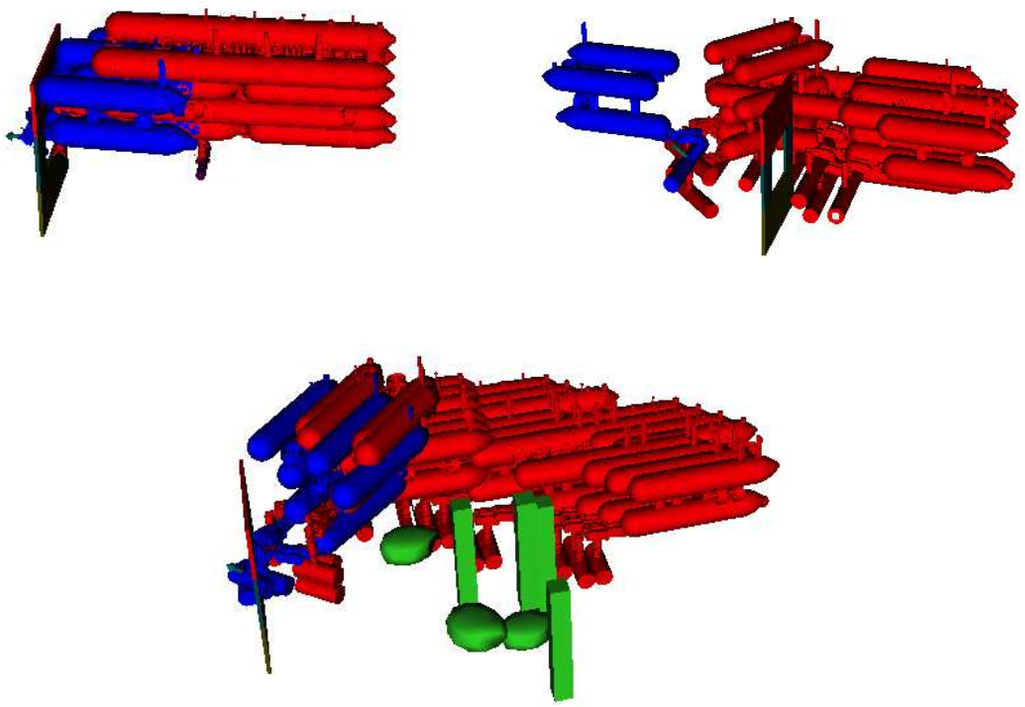


Figure 5.8: Found Solutions for BM#3, BM#4 & BM#5 (base motions in red & arm motions in blue)

Planner	BM#1		BM#2		BM#3		BM#4		BM#5	
	C.I	Success%	C.I	Success%	C.I	Success%	C.I	Success%	C.I	Success%
RRT	1.68	100	2.5	100	6.22	100	5.43	100	4.38	45
RRTConnect	0.78	100	2.3	100	4.62	100	8.72	100	4.36	100
RRT*	1.15	100	1.64	100	4.39	100	5.92	100	3.47	100
STOMP	1.2	100	0.79	95	0.0	5	5.82	15	0.66	20
ARA*	0.61	100	0.9	100	1	100	F	0	2.32	65
MR-MHA*	**0.5**	100	**0.66**	100	**0.59**	100	**4.23**	100	**1.45**	95

Table 5.4: Consistency

Planner	# Expansion				
	BM#1	BM#2	BM#3	BM#4	BM#5
ARA*	7	9	9	F	F
MR-MHA*	12	12	340	80	50

Table 5.5: Average Number of Expansions for search-based Planners

6

Conclusions, discussion and Future Work

In this chapter we first summarize, in Section 6.1, the contribution of this thesis related to the previously proposed objectives (Section 1.2). Later in Section 6.2, potential future work in the same line of research, is discussed.

6.1 Conclusions and Discussion

For the first time, in this thesis, motion planning has been investigated and applied to autonomous underwater intervention. Through this work, we have extended the state of the art of I-AUVs, by demonstrating real-time manipulation in the presence of obstacles in both known a priori and unknown environments.

The objectives previously detailed in Chapter 1, have been successfully covered through the following contributions:

Develop a first solution for an I-AUV using Motion Planning : in an attempt to import the already developed technologies deployed to terrestrial robots, to the underwater field , we developed an initial solution based on *MoveIt!* mobile manipulation framework. We evaluated its motion planning capabilities to control an UVMS and demonstrate beyond-state-of-the-art tasks like valve turning in the presence of virtual obstacles and connector plug/unplug operations using free-floating manipulation, for the first time. Although the work reached its purpose, this early experiment revealed important aspects to be considered in order to achieve higher autonomous intervention capabilities:

- Achieving a real-time response, specially when dealing with unknown environments.
- Keeping a minimum clearance to obstacles to ensure safety of operations, specially with the inherent uncertainty of the robot navigation.
- Generating efficient trajectories with respect to path length.
- Given that the UVMS is a compound system, optimizing the motion of the AUV vs. the arm is a key in order to reach the goal with the shortest safest possible trajectory.
- keeping a sensor field of view to ensure accurate localization through visual servoing, leading to successful manipulation.

Some of these objectives have been already tackled within the task priority framework, but further investigation is needed to introduce them to motion planning for underwater intervention.

Perform an in-depth analysis of state of the art Motion Planning techniques: at this step, we were concerned about choosing the most suitable motion planning approach to reach the overall goal of the thesis. Given the lack of experimental comparison of various approaches, we presented an in-depth analysis of the most common ones.

- A thorough explanation of the state of the art motion planning approaches have been presented.
- A discussion of the challenges introduced to planning for mobile manipulation have been provided.
- Not only a theoretical comparison and categorization of the available approaches have been conducted, but we went the extra mile of performing a qualitative and quantitative comparison. To reach that purpose:

1. Five representative underwater intervention benchmarks have been proposed.
2. Seventeen motion planning algorithms have been compared through the five benchmarks.
3. New quantitative measures have been introduced for comparison.

- A set of guidelines were introduced, based on the comparison results, to help other researchers to base their choice on a scientific methodology.

Based on the results of this analysis, and considering the characteristics desired for the I-AUV: consistency, efficiency in terms of one or more metrics (e.g. path length, clearance...), and above average time response; we decided to adopt the search-based planning strategy for our I-AUV.

Propose a real-time motion planner for an I-AUV: We presented an approach to planning for an UVMS using Multi-Representation, Multi-Heuristic A*. The approach exploits loose coupling between the base and the arm when performing intervention. This is particularly useful given the floating nature of the base (i.e. less execution accuracy) and the limited arm reachability. The approach has been:

- Evaluated by comparing it to other common motion planning approaches using similar benchmarks to those previously developed. The comparison showed the efficiency of the proposed method in terms of consistency, path length, improved planning time and motion compromise between the AUV and the arm.
- Validated in a known apriori environment in simulation, by preforming a two-phases intervening mission in sub-sea structure.
- Validated in a water tank in unknown environment. A significant effort on system integration was required for putting together:
  1. A recently developed real-time laser scanner.
  2. A real-time vision based navigation method.
  3. A visual servoing task priority based method to grasp the valve.

  For that last demonstration, the safety has been considered by adding a clearance cost function to the proposed algorithm.
- Analysed in-depth through additional simulation tests that focused on validating:
  1. The clearance cost function: by performing several tests with various clearance thresholds. The tests demonstrated the success of incorporating the clearance in most of the cases. In addition, it showed an increase in the planning time and the path length with the increase of the clearance threshold.
  2. The loose coupling: the performed tests showed the ability of the algorithm to choose between the AUV and the arm based on the ability of each. Thus, the AUV was used until close to the goal region, while the arm was used only inside the goal region or to manoeuvre an obstacle on the way.

Validate experimentally the proposed approach: at each of the previously stated objectives, and whenever suitable, we focused on a real system demonstration in a

water tank [1, 3, 5], in order to face the challenges of the full stack integration and support our approach - at each step - with a strong validation.

Based on the technical development done during this thesis, we would like to further discuss the benefits and issues that came out from building our solution within an existing wide framework like *MoveIt!*. Having such a generic framework, that covers a wide variety of functionality for mobile manipulation, is like a two-sided weapon. The same features that can be considered benefits in one situation, can turn out as an obstacle in another.

Benefits: The major advantage came from the quick and ease of introducing a new mobile manipulator to *MoveIt!* and the availability of all needed features as configurable plug-ins:

- Easy definition of the UVMS kinematics: The use of a URDF file and the integration with the KDL allowed for an easy definition of our model, avoiding the tedious definition of Inverse Kinematics (IK) solvers or even the integration with available libraries.
- Easy generation of collision-free path: The ability to define the collision geometry of the model, in combination with the Flexible Collision Library (FCL) library, and the group of path-planning algorithms available at the Open Motion Planning Library (OMPL) library makes it easy to compute collision-safe trajectories.
- Reduced development time: Having the arm and AUV low-level control already available, it was possible to implement the coordination layer in charge of the joint AUV/arm control, including the experimental demonstration, in a relatively short time.

Challenges: With further progression of our own approach and the specific needs of our system, the generality of the framework turned out to be a challenge. It became costly to maintain our solution within the framework in terms of development time and limiting features. The major challenges included:

- Non-actuated DoFs: Underactuated AUVs may experience a degree of perturbation on their non-actuated DoFs, for instance, with the roll DoF of our AUV. It is not possible to control the roll, which is passively stable, but it may experience a few degrees of variation due to a perturbation. Because our arm is almost neutrally buoyant and does not induce significant roll disturbances, a compromise solution was adopted. The non-actuated DoFs were included in the URDF model but with a fixed value equivalent to their values at the beginning of each experiment. This solution suffers from a loss of accuracy of the end-effector position in case the value of the non-actuated DoF changes significantly during the manipulation. This is due to a limitation in the URDF modeling as it does not allow read-only joints, where the joint would be skipped during planning but still taken into account during the forward and inverse kinematics computations.
- Rise of solution-based functionalities: on the way of developing our solution, specific needs were arising and extending the existing framework was not a straight forward process:

1. During the survey development, using *MoveIt! benchmark plug-in*, handling end-effector goals, multiple start states, or relying on a scene with objects attached to the end-effector were missing functionalities.
2. Because the OMPL is more commonly used, its employment within *MoveIt!* was more intuitive than any other library. For instance, for the motion planning development, maintaining an internal environment representation was mandatory, while *MoveIt!* had its own representation. Thus, various changes to *MoveIt!* internal pipeline was necessary to keep the framework working as expected with the new approach.
3. The notion of interleaving execution and planning, or replanning, while maintaining the old environment representation (without re-creation of various instances from scratch) is not the *MoveIt!* default behavior.

6.2 Future work

As this thesis puts the first milestone in the usage of motion planning for underwater intervention, it opens doors for further extensions in several directions. A step forward would be to replace the 4 DoF arm with a more dexterous, higher dimensional one (e.g. 6 DoF). This would be more interesting as once the target is reachable, the 6 DoF pose of the end-effector will only depend on the arm, which is faster and far more accurate than the floating base.
Another aspect that has not been tackled in this work, is incorporating the Field of View (FOV) as a constraint during planning, in order to give less preference to the motion that would limit the sensor visibility (e.g. backward motion of the base).
In addition, in the future, it would be desirable to perform tests at sea, and this will require further developments in other areas like the vision-based navigation that would need to be based on Visual SLAM approach, and it is worth considering incorporating the uncertainties in the robot localisation and sensor reading within the planning strategy.
As discussed earlier, the use of *MoveIt!* has been helpful initially, but more an obstacle as our own strategy started to develop. For that, it is worthy to invest effort on building an underwater tailored framework, integrating: modeling, perception, control and motion planning, and simulating the underwater environment to facilitate moving to water tank demonstrations.
In addition, developing further the proposed benchmarks for underwater intervention would be a great contribution to the community. Specially, if a focus is given to analyzing collision-checking, which is a major bottleneck for any planning strategy.
Moreover, further developing the framework is required to allow for interleaving planning and execution, in order to extend the proposed approach to work in an *anytime* fashion, allowing improvement of the computed trajectories as much time as it is given to the planner.
The proposed solution, relies on optimizing the planner strategy to achieve real-time reaction in unknown environment. A drawback of our solution, is that everytime a plan is invalidated due to a sensor update, a plan from scratch is computed. An alternative would be to take profit of the previously constructed search graph, to further reduce the planning time and optimize the reaction to environment changes. A factor we did not analyze, is the planner behavior in more complex and dynamic environment (e.g. in the presence of moving obstacles).

Another direction to consider for intervention in unknown environments, is to have a two-level reaction framework: 1) a local control-based level (e.g. reactive obstacle avoidance in a task priority framework); 2) a global planning-based level similar to our proposed solution. Such integration would take profit of both strategies, and ensures fast and safe reaction, while giving time to the global planning to generate an efficient, optimized trajectory in terms of system-specific criteria.
An interesting next step would be to move to dual-arm manipulation and/or multiple I-AUVs. Depending on how the problem is formulated (e.g. coordinated vs. non-coordinated manipulation, how the goal for each arm is defined...), the complexity and challenges might vary. For example, the overall problem might need to be divided into phases, like for humanoid planning, and various planners at each phase might be better. In order, to gain a better insight on such a different problem, extending the previously developed benchmarks and analyzing the various planning approaches would be beneficial. Moreover, investigating the extension of the proposed solution to multiple-arm/I-AUVs, is worth it, as the loose coupling concept would still hold.

Bibliography

[1] **D. Youakim**, P. Ridao, N. Palomeras, F. Spadafora, D. Ribas, and M. Muzzupappa. "MoveIt!: Autonomous Underwater Free-Floating Manipulation". In: *IEEE Robotics & Automation Magazine* 24.3 (2017), pp. 41–51 (cit. on pp. vii, 13, 104).

[2] **D. Youakim** and P. Ridao. "Motion planning survey for autonomous mobile manipulators underwater manipulator case study". In: *Robotics and Autonomous Systems* 107 (2018), pp. 20–44 (cit. on pp. vii, 25).

[3] **D. Youakim**, P. Cieslak, A. Dornbush, A. Palomer, P. Ridao, and M. Likhachev. "Multi-Representation, Multi-Heuristic A* Search-based Motion Planning for a Free-floating Underwater Vehicle Manipulator System in Unknown Environment". submitted on December 2018 (cit. on pp. vii, 52, 104).

[4] **D. Youakim**, A. Dornbush, M. Likhachev, and P. Ridao. "Motion Planning for an Underwater Mobile Manipulator by Exploiting Loose Coupling". In: *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*. IEEE. 2018, pp. 23–30 (cit. on p. vii).

[5] A. Palomer, P. Ridao, **D. Youakim**, D. Ribas, J. Forest, and Y. Petillot. "3D Laser Scanner for Underwater Manipulation". In: *Sensors* 18.4 (2018), p. 1086 (cit. on pp. vii, 104).

[6] R. Simoni, P. Ridao, P. Cieslak, and **D. Youakim**. "A Novel Obstacle Avoidance Approach for an I-AUV: Preliminary Simulation Results". In: *New Horizon For Underwater Intervention Missions: From Current Technologies to Future Applications IROS Workshop*. IEEE. 2018 (cit. on p. vii).

[7] R. Simoni, P. Ridao, P. Cieslak, and **D. Youakim**. "A Novel Obstacle Avoidance Approach for an I-AUV". In: *Autonomous Underwater Vehicle (AUV), 2018 IEEE*. IEEE. 2018 (cit. on p. viii).

[8] P. Ridao, M. Carreras, D. Ribas, P. J. Sanz, and G. Oliver. "Intervention AUVs: The next challenge". In: *Annual Reviews in Control* 40 (2015), pp. 227–241. ISSN: 1367-5788. DOI: http://dx.doi.org/10.1016/j.arcontrol.2015.09.015 (cit. on p. 8).

[9] H. H. Wang, S. M. Rock, and M. J. Lee. "OTTER: The design and development of an intelligent underwater robot". In: *Autonomous Robots* 3.2-3 (1996), pp. 297–320 (cit. on p. 8).

[10] H.-T. Choi, A. Hanai, S. K. Choi, and J. Yuh. "Development of an underwater robot, ODIN-III". In: *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*. Vol. 1. IEEE. 2003, pp. 836–841 (cit. on p. 8).

[11] M. Ishitsuka and K. Ishii. "Development of an underwater manipulator mounted for an AUV". In: *OCEANS, 2005. Proceedings of MTS/IEEE*. IEEE. 2005, pp. 1811–1816 (cit. on p. 8).

[12] J. Yuh, S. Choi, C. Ikehara, G. Kim, G. McMurty, M. Ghasemi-Nejhad, N. Sarkar, and K. Sugihara. "Design of a semi-autonomous underwater vehicle for intervention missions (SAUVIM)". In: *Underwater Technology, 1998. Proceedings of the 1998 International Symposium on*. IEEE. 1998, pp. 63–68 (cit. on p. 8).

[13] G. Marani, S. K. Choi, and J. Yuh. "Underwater autonomous manipulation for intervention missions AUVs". In: *Ocean Engineering* 36.1 (2009), pp. 15–23 (cit. on p. 8).

[14] P. J. Sanz, M. Prats, P. Ridao, D. Ribas, G. Oliver, and A. Ortiz. "Recent progress in the RAUVI project: A reconfigurable autonomous underwater vehicle for intervention". In: *Proceedings ELMAR-2010*. IEEE. 2010, pp. 471–474 (cit. on p. 8).

[15] P. J. Sanz, P. Ridao, G. Oliver, C. Melchiorri, G. Casalino, C. Silvestre, Y. Petillot, and A. Turetta. "TRIDENT: A framework for autonomous underwater intervention missions with dexterous manipulation capabilities". In: *IFAC Proceedings Volumes* 43.16 (2010), pp. 187–192 (cit. on p. 8).

[16] O. Khatib. *Ocean One*. `https://cs.stanford.edu/groups/manips/ocean-one.html`. 2016 (cit. on p. 8).

[17] P. Marty. "ALIVE: An autonomous light intervention vehicle, Advances in Technology for Underwater Vehicles Conference". In: *Oceanology Int* (2004) (cit. on p. 8).

[18] N. Palomeras, A. Peñalver, M. Massot-Campos, P. L. Negre, J. J. Fernández, P. Ridao, P. J. Sanz, and G. Oliver-Codina. "I-AUV docking and panel intervention at sea". In: *Sensors* 16.10 (2016), p. 1673 (cit. on p. 9).

[19] D. M. Lane, F. Maurelli, P. Kormushev, M. Carreras, M. Fox, and K. Kyriakopoulos. "PANDORA-Persistent Autonomy through Learning, Adaptation, Observation and Replanning". In: (2012) (cit. on p. 9).

[20] A. Carrera, N. Palomeras, N. Hurtos, P. Kormushev, and M. Carreras. "Learning by demonstration applied to underwater intervention". In: 17th International Conference of the Catalan Association of Artificial Intelligence. 2014 (cit. on pp. 9, 11).

[21] Y. Nakamura, H. Hanafusa, and T. Yoshikawa. "Task-priority based redundancy control of robot manipulators". In: *The International Journal of Robotics Research* 6.2 (1987), pp. 3–15 (cit. on p. 9).

[22] P. Cieslak and P. Ridao. "Adaptive Admittance Control in Task-Priority Framework for Contact Force Control in Autonomous Underwater Floating Manipulation". In: *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*. IEEE. 2018, pp. 30–36 (cit. on pp. 9, 11).

[23] S. M. LaValle. *Planning algorithms*. Cambridge university press, 2006 (cit. on p. 9).

[24] H. Choset, K. Lynch, H. Seth, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun. *Principles of Robot Motion-Theory , Algorithms , and Implementation*. 2005. ISBN: 0262033275 (cit. on p. 9).

[25] J.-C. Latombe. *Robot motion planning*. Vol. 124. Springer Science & Business Media, 2012 (cit. on p. 9).

[26] D. Katz, J. Kenney, and O. Brock. "How can robots succeed in unstructured environments?" In: *Robotics Science and Systems (RSS) Workshop on Robot Manipulation* (2008) (cit. on p. 9).

[27] B. Cohen, S. Chitta, and M. Likhachev. "Single- and Dual-Arm Motion Planning with Heuristic Search". In: *The International Journal of Robotics Research* (2013) (cit. on pp. 9, 11).

[28] N. Palomeras, A. El-Fakdi, M. Carreras, and P. Ridao. "COLA2: A control architecture for AUVs". In: *IEEE Journal of Oceanic Engineering* 37.4 (2012), pp. 695–716 (cit. on p. 10).

[29] M. Carreras, J. D. Hernández, E. Vidal, N. Palomeras, D. Ribas, and P. Ridao. "Sparus II AUV-A Hovering Vehicle for Seabed Inspection". In: *IEEE Journal of Oceanic Engineering* (2018) (cit. on p. 11).

[30] D. Ribas, N. Palomeras, P. Ridao, M. Carreras, and A. Mallios. "Girona 500 AUV: From Survey to Intervention". In: *Mechatronics, IEEE/ASME Transactions on* 17.1 (2012), pp. 46–53. ISSN: 1083-4435. DOI: 10.1109/TMECH.2011.2174065 (cit. on p. 11).

[31] A. Carrera, S. Ahmadzadeh, A. Ajoudani, P. Kormushev, M. Carreras, and D. G. Caldwell. "Towards autonomous robotic valve turning". In: *Cybernetics and Information Technologies* 12.3 (2012), pp. 17–26 (cit. on p. 11).

[32] A. Carrera, N. Palomeras, N. Hurtós, P. Kormushev, and M. Carreras. "Cognitive system for autonomous underwater intervention". In: *Pattern Recognition Letters* 67 (2015). Cognitive Systems for Knowledge Discovery, pp. 91–99. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2015.06.010 (cit. on p. 11).

[33] P. Cieslak, P. Ridao, and M. Giergiel. "Autonomous underwater panel operation by GIRONA500 UVMS: A practical approach to autonomous underwater manipulation". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. May 2015, pp. 529–536. DOI: 10.1109/ICRA.2015.7139230 (cit. on p. 11).

[34] E. Hernández, M. Carreras, J. Antich, P. Ridao, and A. Ortiz. "A topologically guided path planner for an auv using homotopy classes". In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2011, pp. 2337–2343 (cit. on p. 11).

[35] E. Galceran Yebenes. "Coverage path planning for autonomous underwater vehicles". PhD thesis. Universitat de Girona, 2014 (cit. on p. 11).

[36] J. D. Hernández, K. Istenič, N. Gracias, N. Palomeras, R. Campos, E. Vidal, R. García, and M. Carreras. "Autonomous Underwater Navigation and Optical Mapping in Unknown Natural Environments". In: *Sensors* 16.8 (2016), p. 1174 (cit. on p. 11).

[37] E. Vidal, J. D. Hernández, K. Istenic, and M. Carreras. "Online View Planning for Inspecting Unexplored Underwater Structures." In: *IEEE Robotics and Automation Letters* 2.3 (2017), pp. 1436–1443 (cit. on p. 11).

[38] N. Palomeras, N. Hurtós, M. Carreras, and P. Ridao. "Autonomous Mapping of Underwater 3-D Structures: From View Planning To Execution". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 1965–1971 (cit. on p. 11).

[39] M. Likhachev, G. Gordon, and S. Thrun. "ARA*: Anytime A* with provable bounds on sub-optimality". In: *Advances in Neural Information Processing Systems* 16 (2004), p. 12. ISSN: 10495258. DOI: 10.1.1.3.9449 (cit. on p. 11).

[40] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun. "Anytime search in dynamic graphs". In: *Artificial Intelligence* 172.14 (2008), pp. 1613–1643 (cit. on p. 11).

[41] M. Likhachev and A. Stentz. "Path clearance". In: *IEEE robotics & automation magazine* 16.2 (2009) (cit. on p. 11).

[42] O. Kermorgant, Y. Pétillot, and M. Dunnigan. "A global control scheme for free-floating vehicle-manipulators". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 5015–5020 (cit. on p. 52).

[43] **D. Youakim**, P. Ridao, and N. Palomeras. "MoveIt! Based Implementation of An I-AUV". MA thesis. University Of Girona - Vibot Master, 2015 (cit. on p. 86).

[44] N. Palomeras, S. Nagappa, D. Ribas, N. Gracias, and M. Carreras. "Vision-based localization and mapping system for AUV intervention". In: *OCEANS'13 MTS/IEEE*. 2013 (cit. on p. 86).

[45] F. Spadafora, M. Muzzupappa, F. Bruno, D. Ribas, and P. Ridao. "Design and construction of a robot hand prototype for underwater applications". In: *IFAC-PapersOnLine* 48.2 (2015), pp. 294–299 (cit. on p. 88).

[46] S. M. LaValle. "Rapidly-exploring random trees: A new tool for path planning". In: (1998) (cit. on p. 98).

[47] J. Kuffner and S. LaValle. "RRT-connect: An efficient approach to single-query path planning". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)* 2 (2000), pp. 995–1001. DOI: 10.1109/ROBOT.2000.844730 (cit. on p. 98).

[48] S. Karaman and E. Frazzoli. "Sampling-based algorithms for optimal motion planning". In: *The international journal of robotics research* 30.7 (2011), pp. 846–894 (cit. on p. 98).

[49] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. "STOMP: Stochastic trajectory optimization for motion planning". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE. 2011, pp. 4569–4574 (cit. on p. 98).

Acronyms

AHRS	Attitude and Heading Reference System.
ARA*	Anytime Repairing A*.
AUV	Autonomous Underwater Vehicle.
BFS	Breadth First Search.
CHOMP	Covariant Hamiltonian Optimization for Motion Planning.
CIRS	Underwater Robotics and Vision Research Center.
cola2	Component Oriented Layer-based Architecture v.2.
CS	Configuration Space.
DoF	Degree of Freedom.
DVL	Doppler Velocity Log.
EUCLID	Euclidean.
FCL	Flexible Collision Library.
FOV	Field of View.
GPS	Global Positioning System.
I-AUV	Intervention Autonomous Underwater Vehicle.
IK	Inverse Kinematics.
IMR	Inspection Maintance Repair.
LARA*	Lazy Anytime Repairing A*.
MI	Manipulability Index.

MP	Motion Primitives.
OMPL	Open Motion Planning Library.
PRM	Probabilistic Roadmap Method.
ROV	Remotely Operated Vehicle.
RRT	Rapidly Exploring Random Trees.
SBL	Single-query Bi-directional Lazy collision checking.
SMPL	Search Motion Planning Library.
SRDF	Semantic Robot Description File.
STOMP	Stochastic Trajectory Optimization for Motion Planning.
STRIDE	Search Tree with Resolution Independent Density Estimation.
TP	Task Priority.
TPRC	Task-Priority Redundancy Control.
UdG	Universitat de Girona.
URDF	Universal Robot Description File.
UUV	Unmanned Underwater Vehicle.
UVMS	Underwater Vehicle Manipulation System.
WS	Workspace.