THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National Polytechnique de Toulouse (INP Toulouse)

Discipline ou spécialité :

Intelligence Artificielle

Présentée et soutenue par :
Assia BELBACHIR
le : jeudi 10 février 2011

Titre :

A Cooperative Architecture For Target Localization Using Underwater Vehicles - Une Architecture Coopérative pour la Localisation de Cibles Marines avec des Véhicules Sous-Marins -

---

JURY
Rachid Alami (LAAS) - Président du jury
Amal El Fallah Seghrouchni (Université Paris 6) - Rapporteur
David Andreu (Université Montpellier 2) - Rapporteur
Lionel Lapierre (Université Montpellier 2) - Membre
Jan Opderbecke (IFREMER Centre Méditerranée) - Membre

Ecole doctorale :
Mathématiques Informatique Télécommunications (MITT)
Unité de recherche :
LAAS-CNRS
Directeur(s) de Thèse :
Simon Lacroix (LAAS) - Directeur
Félix Ingrand (LAAS) - Co-directeur
Rapporteurs :
Amal El Fallah Seghrouchni
David Andreu

INPT
LAAS-CNRS

THÈSE

présente en première version en vu d'obtenir le grade de Docteur, spécialité
"Intelligence Artificielle"

par

Assia Belbachir

A COOPERATIVE ARCHITECTURE FOR TARGET LOCALIZATION USING UNDERWATER VEHICLES

– UNE ARCHITECTURE COOPÉRATIVE POUR LA LOCALISATION DE CIBLES MARINES AVEC DES VÉHICULES SOUS-MARINS –

Thèse soutenue le 10/02/2011 devant le jury composé de :

M^{me}	AMAL EL FALLAH SEGHROUCHNI	LIP6	(Rapporteur)
M.	DAVID ANDREU	LIRMM	(Rapporteur)
M.	LIONEL LAPIERRE	LIRMM	(Jury)
M.	JAN OPDERBECKE	IFREMER	(Jury)
M.	RACHID ALAMI	LAAS-CNRS	(Jury)
M.	FÉLIX INGRAND	LAAS-CNRS	(Directeur)
M.	SIMON LACROIX	LAAS-CNRS	(Directeur)

A ma grand mère pour son humeur,
a mes parents pour leurs encouragements
et a ma sœur pour sa présence…

REMERCIEMENTS

BUDDHA, SAYS: « Let us rise up and be thankful, for if we didn't learn a lot today, at least we learned a little, and if we didn't learn a little, at least we didn't get sick, and if we got sick, at least we didn't die; so, let us all be thankful.»

Je tiens à remercier et à exprimer toute ma reconnaissance auprès de mes deux encadrants Félix Ingrand ainsi que Simon Lacroix. Ils m'ont initiée à la recherche dans le domaine de la robotique qui m'a toujours motivé. Ils ont su me diriger afin que j'aboutisse à la clôture de ma thèse. Je suis très fière de la formation de chercheuse acquise sous leur encadrement. Je remercie Félix Ingrand de m'avoir donné la possibilité de découvrir différents travaux comme ceux du MBARI et ceux de l'IFREMER.

Je remercie mon rapporteur Amal El Fallah Seghrouchni pour l'honneur qu'elle m'a accordée en acceptant d'évaluer ma thèse. Merci pour son humour dans les emails et ses félicitations à la fin de ma thèse, j'espère un jour recroiser son chemin.

Je remercie mon rapporteur David Aandreu d'avoir accepté de rapporter ma thèse. Je le remercie également pour les commentaires très constructifs qu'il a formulés. J'espère que le vent m'amènera sur des terres non explorées.

Je remercie Jan Opderbecke d'avoir accepté de faire partie de mon jury. Au delà de cela, Jan m'a beaucoup aidé lors de mon séjour à Toulon afin que j'exploite au mieux les travaux existants à l'IFREMER pour l'aboutissement de ma thèse.

Je remercie Lionel Lapierre d'avoir accepté d'être membre de mon jury de thèse et de ses différentes questions lors de ma présentation, qui m'ont permise de prendre du recul sur ce qui a été fait dans ma thèse et ce qui reste à faire.

Je remercie Rachid Alami d'avoir présidé ma thèse. Je le remercie aussi pour ses encouragements durant ma deuxième année de thèse.

J'ai eu le plaisir durant ma thèse de travailler avec l'équipe PRAO (Positionnement, Robotique, Acoustique et Optique) ainsi qu'avec Michel Perrier et Vincent Rigaud à l'IFREMER qui ont été très accueillants avec moi durant mon séjour à Toulon et qui ont mis à ma disposition tous les outils dont j'avais besoin pour ma thèse.

J'ai aussi eu le plaisir de travailler avec Kanna Rajan, McGann Conor et Fréderic Py durant mon séjour au MBARI (Monterrey Bay Aquarium Research Institute). Je les remercie pour leur disponibilité á me présenter leurs travaux et me faire visiter la Californie.

J'ai aussi apprécié mon séjour au sein du LAAS-CNRS et plus particulièrement dans le groupe RIA qui m'a donné l'envie de travailler.

Je suis reconnaissante à Joan de l'aide qu'il m'a apporté à des moments clés de ma thèse, cela m'a permis de m'améliorer au fil des années.

Je remercie Moky et Samir d'avoir répondu présent lors de ma présentation et je les remercie des différents conseils qu'ils m'ont donnés.

Je remercie Luis de m'avoir encouragé lors de mes premières années de thèse pour continuer le travail. Je remercie Akin et Raquel de m'avoir encouragé par des mots et emails durant ma thèse, même s'ils étaient loin de Toulouse.

Je remercie mes amis, non roboticiens pour tous ce qu'ils m'ont offert comme agréable moments en Algérie (Nawel, Aicha, Mourad, Féthi, Sihem, Nabila, Nesrine, etc.) mais aussi en France (Assia, Nawar, Ines, Hamida, Toufik, Aamir, Kostas, Farah, Mehdi, Hamada, Nedia, Amina, Nizar, Solene, Sofie, Matthieu, Oussama, Prayjoth, Rami, Ixbalank, Juan, Giuliat, Gilberto, Marlene, David, Karina, Saad, Cyril, Redouane, Mathieu, Diego, Carlos, Husnein, Aisha, Zinnedine, Ali, Rym, etc).

Je remercie Önder de m'avoir encouragé vers la fin de ma thèse. Je remercie Ali, Naveed et Lavindra, avec qui j'ai partagé la plupart de mon temps en discutant de plein de choses de la vie. J'ai passé d'inoubliables moments en leur compagnie et merci pour le discours enregistré.

Je remercie le magnifique couple Pandey et Sabita, qui m'ont invité plusieurs fois chez eux afin de décompresser durant les durs moments de ma thèse.

Je remercie les filles de mon groupe Viorela, Imen et Layale d'avoir été là à un moment de la thèse pour discuter, faire du sport, rigoler et surtout partager des moments inoubliables.

Je remercie ma grande famille : mes oncles, tantes, cousines et grandes mères (Abdellah, Krimo, Fadela, Fatiha, Mehdi, Kheira, Kamel, Amel, Malika, etc. La liste est très longue !) et mes deux grandes mères Manena et Lela pour leur soutien spirituel indéfectible. Je n'oublirai pas de remercier Tata Saida et tonton Hassane pour leur soutien moral tout au long de mes études en France.

Tout le mérite revient à mes parents et ma sœur qui m'ont encouragée, aidé moralement pour que je continue ma thèse. Sans leur soutien moral, cette thèse n'aurait pu commencer.

CONTENTS

CONTENTS vii

ABSTRACT 1

1 INTRODUCTION 3
  1.1 MOTIVATIONS AND CONTRIBUTIONS 3
  1.2 THESIS OVERVIEW 4
  1.3 PUBLICATIONS 5

2 CONTEXT, STATE OF THE ART AND OVERALL APPROACH 7
  2.1 INTRODUCTION 9
  2.2 UNDERWATER VEHICLES 9
    2.2.1 Types of underwater vehicles 9
    2.2.2 Types of mission 12
    2.2.3 Main constraints associated to underwater environments 13
    2.2.4 Levels of autonomy 15
    2.2.5 Overview of AUV related work 15
    2.2.6 Synthesis 17
  2.3 PARADIGMS FOR DISTRIBUTED INTELLIGENCE 17
    2.3.1 Bio-inspired paradigms 17
    2.3.2 Organizational and social paradigms 17
    2.3.3 Knowledge-based paradigms 19
    2.3.4 Synthesis and problematic 19
  2.4 MULTIPLE UNDERWATER VEHICLES EXPLORATION 19
    2.4.1 Exploration strategies with task control 20
    2.4.2 Exploration strategies with navigation control 21
    2.4.3 Adaptive exploration strategies 21
  2.5 OVERVIEW OF OUR APPROACH 22

3 BUILDING TARGET MAPS 25
  3.1 INTRODUCTION 27
  3.2 REALISTIC TARGET MODEL 27
  3.3 SENSORS 29
    3.3.1 Existent sensors for target localization 29

3.3.2 Classification of existent sensors 29
3.4 TARGET DETECTION MODELS 30
3.4.1 Target detection model with known maximal temperature 31
3.4.2 Target detection model with unknown maximal temperature 32
3.5 MAP BUILDING 34
3.5.1 Range-only sensor update 35
3.5.2 Bearing-only sensor update 37
3.5.3 Illustration of map building with a single robot 37
3.6 MULTI ROBOT MAP BUILDING 38
3.6.1 Communication model 38
3.6.2 Fusing maps among underwater vehicles 39
3.6.3 Communication bandwidth and amount of exchanged data 39
3.6.4 Illustration 42
CONCLUSION 42

4 ADAPTIVE EXPLORATION STRATEGIES **43**
4.1 INTRODUCTION 45
4.2 MOTION MODEL 45
4.3 EXPLORATION STRATEGIES 45
4.3.1 Non adaptive exploration strategy 45
4.3.2 Adaptive strategies 46
4.3.3 Adaptive Cooperative Exploration Strategy (ACES): Algorithm 47
4.4 STATISTICAL RESULTS USING DIFFERENT COOPERATIVE SCENARIOS 51
4.4.1 Adaptive vs. Non-adaptive Exploration Strategies 51
4.4.2 Importance of the Coverage 53
CONCLUSION 54

5 PROPOSED ARCHITECTURE AND DEPLOYED SYSTEM **55**
5.1 INTRODUCTION 57
5.2 ROBOT CONTROL ARCHITECTURES 57
5.2.1 Control architecture requirements 57
5.2.2 Different control architectures 58
5.2.3 Summary 62
5.3 THE T-REX ARCHITECTURE 63
5.3.1 Planning system in the T-ReX Architecture 63
5.3.2 Planning problem and plan resolution in the T-ReX architecture 65
5.3.3 The deployed T-ReX architecture 67
5.3.4 A database 68
5.3.5 A deliberative (planner) 68
5.3.6 A dispatcher and a synchronizer 70
5.3.7 The developed reactors in the T-ReX architecture 70
5.3.8 The T-ReX architecture for PR2 71

5.4 Description of the implemented model and the algorithms in the CoT-ReX architecture . . . 72
    5.4.1 The timelines in the Mission Manager . . . 73
    5.4.2 The timelines in the Executer . . . 74
    5.4.3 The architecture of the MapReactor . . . 75
    5.4.4 The architecture of the CoopReactor . . . 77
Conclusion . . . 77

6 Simulation results 79
6.1 Introduction . . . 81
6.2 Simulation results for target detection . . . 81
    6.2.1 Simulation environment . . . 81
    6.2.2 Mission definition and role of each underwater vehicle . . . 82
    6.2.3 The phases of case studies . . . 82
    6.2.4 Case 1: Evaluation of cooperative predefined exploration strategies . . . 82
    6.2.5 Case 2: Evaluation of cooperative exploration strategies at different depths . . . 86
    6.2.6 Case 3: Exploration strategies at three different depths . . . 88
6.3 Simulation results using IFREMER simulator . . . 89
    6.3.1 Simulation environment . . . 89
6.4 Implementation at IFREMER . . . 90
    6.4.1 Mission definition and simulation . . . 90
    6.4.2 Mission experimentation . . . 90
Discussion . . . 92

Conclusion and Perspectives 93

A Résumé des travaux 95
A.1 Introduction . . . 97
A.2 Contributions . . . 97
A.3 Etat de l'art . . . 98
A.4 Contexte de travail . . . 99
    A.4.1 Scénario considéré . . . 99
    A.4.2 Proposition . . . 100
A.5 L'architecture mono-robot . . . 100
A.6 Proposition d'architecture . . . 102
A.7 Implementations et résultats . . . 106
    A.7.1 Scenario I . . . 106
    A.7.2 Scenario II . . . 107
Conclusion et perspectives . . . 109

Bibliography 111

ABSTRACT

THERE is a growing research interest in Autonomous Underwater Vehicles (AUV), due to the need for increasing our knowledge about the deep sea and understanding the effects the human way of life has on it. This need has pushed the development of new technologies to design more efficient and more autonomous underwater vehicles. Autonomy refers, in the context of this thesis, to the "decisional autonomy", i.e. the capability of taking decisions, in uncertain, varying and unknown environments.

A more recent concern in AUV area is to consider a fleet of vehicles (AUV, ASV, etc). Indeed, multiple vehicles with heterogeneous capabilities have several advantages over a single vehicle system, and in particular the potential to accomplish tasks faster and better than a single vehicle.

Underwater target localization using several AUVs (Autonomous Underwater Vehicles) is a challenging issue. A systematic and exhaustive coverage strategy is not efficient in term of exploration time: it can be improved by making the AUVs share their information and cooperate to optimize their motions. The contribution of this thesis is the definition of an architecture that integrates such a strategy that adapts each vehicle motions according to its and others' sensory information. Communication points are required to make underwater vehicles exchange information: for that purpose the system involves one ASV (Autonomous Surface Vehicle), that helps the AUVs re-localize and exchange data, and two AUVs that adapt their strategy according to gathered information, while satisfying the associated communication constraints. Each AUV is endowed with a sensor that estimates its distance with respect to targets, and cooperates with others to explore an area with the help of an ASV. To provide the required autonomy to these vehicles, we build upon an existing system (`T-REX`) with additional components, which provides an embedded planning and execution control framework. Simulation results are carried out to evaluate the proposed architecture and adaptive exploration strategy.

1

INTRODUCTION

1.1 MOTIVATIONS AND CONTRIBUTIONS

In general, most of robotic systems such as spacecraft, rovers, or underwater vehicles execute predefined commands that are sequenced and monitored by remote operators. But communications between the surface and a spacecraft or underwater vehicles, are highly constrained, and affect the efficiency of the mission if the operators are in the control loop: to improve this efficiency, researchers embed decisional autonomy to plan the robot activities, control the execution of goals and monitor the state of the system.

In this thesis, we focussed on underwater vehicles, whose objective is to explore large undersea areas. The problem of exploring an unknown area is a central issue in mobile robotics. In general, the coverage of the entire terrain is required, but this is not practical due to resource costs (e.g. energy, execution time of the mission). This resource constraint creates an important issue in designing an exploration strategy for an underwater vehicle; how does the underwater vehicle decide where to explore next? In this thesis we present an adaptive exploration strategy based on sensed data. In contrast to several exploration strategies that are non adaptive [Rahimi04], our approach tries to modify the trajectory of the vehicle by maximizing the information gain to localize the maximal number of marine targets.

To improve the capabilities of one underwater vehicle, we choose to use several underwater vehicles. Nowadays, there is an increase in the use of multiple autonomous vehicles: several vehicles bring robustness, allow for faster and more efficient missions, and achieve missions that intrinsically require the use of a fleet of vehicles. When it comes to information gathering missions (e.g. exploration, surveillance, target detection and localization) synergies occur when robots effectively communicate to merge information gathered on the environment and to coordinate their observation plans. In the robotics community, several studies have been done in this direction and there is an extensive literature on communicating robot fleets, *e.g.* [Stenz04].

In our context, we are interested in exploration scenarios where a fleet of vehicles survey an unknown area and try to localize targets (plume, hot-spot, mapping, etc) as fast as possible. The acoustic nature of the water greatly restricts the communication range and bandwidth [Meyer06]: communication between two vehicles can only be reliably established when both are inside a vertical acoustic cone with respect to each other. Using an ASV (Autonomous Surface Vehicle) as a communication hub reduces the AUVs energy consumption and the overall mission duration, as the AUVs do not go to the surface to exchange data. Furthermore, the ASV can correct the AUV position drift that eventually occurs.

So each AUV gathers information to locate these targets, while the ASV acts as a communication hub between all AUVs and refines their localization estimate. Every AUV has autonomous control abilities that enables it to achieve a given pre-planned sequence of tasks and motions. *Rendezvous* with the ASV are required to exchange data between the AUVs – and for the operator to supervise the mission. From an operational point of view, an AUV cannot be let freewheeling under the surface, depending entirely on the data it gathers to define its motions. In this context, information-driven strategies have to be integrated within a pre-planned scheme that contains communication *rendezvous*, *i.e.* spatio-temporal constraints.

Our thesis aims at defining an architecture that allows each vehicle to direct itself according to the collected information and satisfy pre-defined spatio-temporal constraints. Our contributions can be summarized as follows :

- Introduction of a data driven approach that allows each vehicle to take into account the measured data during the exploration and to reason about this data for more precise localization of the target.
- Integration of this approach with a task planner and an execution controller that takes into account temporal constraints for each vehicle.

1.2 THESIS OVERVIEW

This document is organized along the following chapters:
Chapter 2 is an introduction to the different types of underwater vehicles, the types of mission that they can achieve and their various levels of autonomy. We discuss different constraints related to the water environment, present existing work on exploration strategies, and end the chapter by depicting the objectives of our work and our overall approach.

Chapter 3 introduces the models of the target and sensors we consider, and the way they are exploited to build probabilistic maps of the target locations. Our models lead to the definition of two types of sensors: range-only and bearing-only sensors. Illustrations of map building using one or two communicating AUVs are provided.

Chapter 4 introduces the adaptive strategies: according to the built map, the vehicles choose the next cell to explore. Two adaptive strategies are defined, and illustrated with a single AUV and with two communicating AUVs.

Chapter 5 presents the implemented system. Our algorithms are integrated within the `T-ReX` architecture that deals with time constraints, to which we have added two components to manage the map building and the communications between the AUVs.

Chapter 6 provides various simulation results for different configurations of target mapping missions. The integration of our overall architecture with the hardware-in-the-loop simulator developed at Ifremer is also presented.

Finally, we make a general conclusion that shows the panorama of our work and possible future extensions, and appendix A summarises our work in french.

1.3 PUBLICATIONS

The following publications are related to this work:

- **Cooperative-Adaptive Algorithms for Targets Localization in Underwater Environment** - A. Belbachir, F. Ingrand, S. Lacroix, M. Perrier. International Conference on Autonomous Underwater Vehicle IEEE / Oceanic Engineering Society (AUV2010), Monterey, CA, USA, September 1-3, 2010.

- **Localizing Underwater Targets using a Cooperative AUV Architecture** - A. Belbachir, F. Ingrand, S.Lacroix. International Conference on Machine and Web Intelligence (ICMWI'2010), USTHB University, Algiers, October 3 - 5, 2010.

- **A Cooperative Architecture for Target Localization with Underwater Vehicles** - A. Belbachir, F. Ingrand, S.Lacroix. International Symposium on Unmanned Untethered Submersible Technology (UUST), NH, USA, August 23 - 26, 2009.

- **Architecture pour la planification et l'exécution d'une flotte de véhicules marins et sous-marins.** A.Belbachir, F.Ingrand and S.Lacroix. 4th National Conference on "Control Architectures of Robots" ONERA – Toulouse, France, April 23 - 24, 2009.

2

CONTEXT, STATE OF THE ART AND OVERALL APPROACH

A lot of research has dealt with ground and underwater vehicles for exploration missions. In this chapter, we present the existent work in the literature and the specific constraints associated to underwater vehicles. We end the chapter by presenting the general idea of the proposed approach.

ANTHONY ROBBINS, SAYS: «To effectively communicate, we must realize that we are all different in the way we perceive the world and use this understanding as a guide to our communication with others.» [Unlimited power: The New science of Personal Achievement (1987) p.237]

2.1 INTRODUCTION

There is a growing research interest in using Autonomous Underwater Vehicles (AUV) to better understand the deep sea biological and chemical phenomena as well as the effects the human way of life has on them. This need has pushed the development of new technologies to design more efficient and more autonomous underwater vehicles. A recent concern in this area is to consider not only one vehicle, but a fleet of cooperating vehicles (Autonomous Underwater Vehicles (AUV), Autonomous Surface Vehicles (ASV), etc). Indeed, multiple robots (with heterogeneous capabilities) have several advantages over a single robot system [Burgard05]. Cooperative robots, when properly managed, have the potential to accomplish tasks faster and better than robots evolving independently. Numerous studies propose interesting architectures for robots cooperation, but most of them are focused on terrestrial robots for which the communication among the vehicles is reliable and permanent, whereas underwater environment is very limiting with respect to communication.

This chapter presents an overview of the literature using cooperative vehicles and the different problems encountered for underwater vehicles exploration. Section 2.2 presents the different types of existing underwater vehicles and the associated constraints. In Section 2.3 we briefly describe existing work on cooperating terrestrial vehicles and show the limitations of these approaches in the context of underwater vehicles. In Section 2.4 we show the associated constraints for cooperative underwater vehicles and present the main approaches for exploration. Finally, section 2.5 outlines the main characteristics of our approach.

2.2 UNDERWATER VEHICLES

2.2.1 Types of underwater vehicles

There are two classes of underwater vehicles, depending if the human is on-board the vehicle or not. The two classes are presented as follow, with a surface underwater vehicle:

a. Populated submarines The vehicle is controlled by an onboard human. An example of these vehicles is the `Nautile`, developed at IFREMER for marine exploration, with a maximal depth of 6000m.

b. Unmanned Underwater Vehicles (UUV) These vehicles do not have humans on-board, and are classified according to a variation of their degree of autonomy:

b.1. Remotely Operated Vehicles (ROV) A ROV is a tele-operated engine, linked with the surface by a tether. This tether (or umbilical) is a set of cables that carry to the vehicle electrical power, control data from the surface, and bring back to the surface various data. `Victor6000` [Michel03] is a ROV equipped with cameras and different sensors dedicated to

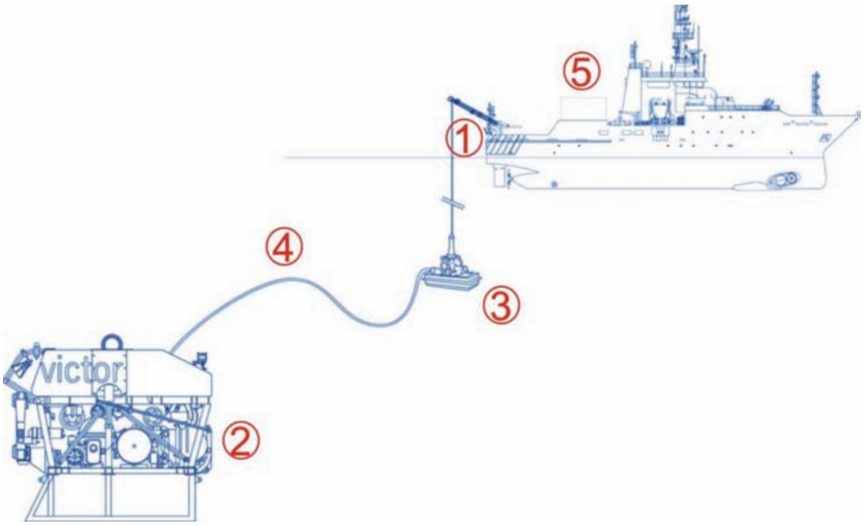


Figure 2.1 – *Victor6000, a Remotely Operated Vehicle used at IFREMER. (1) 8500 meters long umbilical of 20 mm diameter a 30 tons weight, (2) the 4 tons vehicle, (3) and (4) a jetsam located within 100 to 300 meters depth, and (5) the 20 feet container that fuels the vehicle with energy.*

oceanography (Figure 2.1). `VORTEX` [Rigaud94] is also an experimental ROV used at IFREMER, that can be used as an AUV when it is on automatic mode. The main drawback of ROVs is the umbilical, that restricts its progress.

b.2. Autonomous Underwater Vehicles (AUV) To get rid of the umbilical, new kinds of vehicles called "Autonomous Underwater vehicles" (AUVs) have been designed. Contrary to a ROV, and AUV embeds all its energy source and its mission controller.

For a specific mission, an AUV communicates with the surface ship to correct its localization, send collected data or receive new commands, etc. Two types of communication can be used: radio or acoustic. Radio communication are only effective when the vehicle is at the surface level, whereas acoustic communications can be used when the vehicle is diving, in a volume roughly restricted to a cone beneath the surface communication point, and with a small bandwidth.

Figure 2.2 shows the AUV Asterix used at IFREMER. The main devices that equip this vehicle are:

- Sound sediments sensor and Obstacle sounders sensor are respectively used to measure the proximity of the ground or obstacles from the vehicle by using an acoustic signal[1].
- GAPS (Global Acoustic Positioning System) and DGPS are used to estimate the vehicle position.
- Acoustic communication device (MATS: Multimodulation Acoustic Telemetry System), that allows the vehicle to communicate with the surface.
- Radio antenna to communicate with the surface ship.

The central unit (Vehicle Control Computer or VCC) embeds the control architecture of the vehicle.

---

[1] The distance is computed by the time taken to the acoustic to propagate until it returns back to the vehicle.

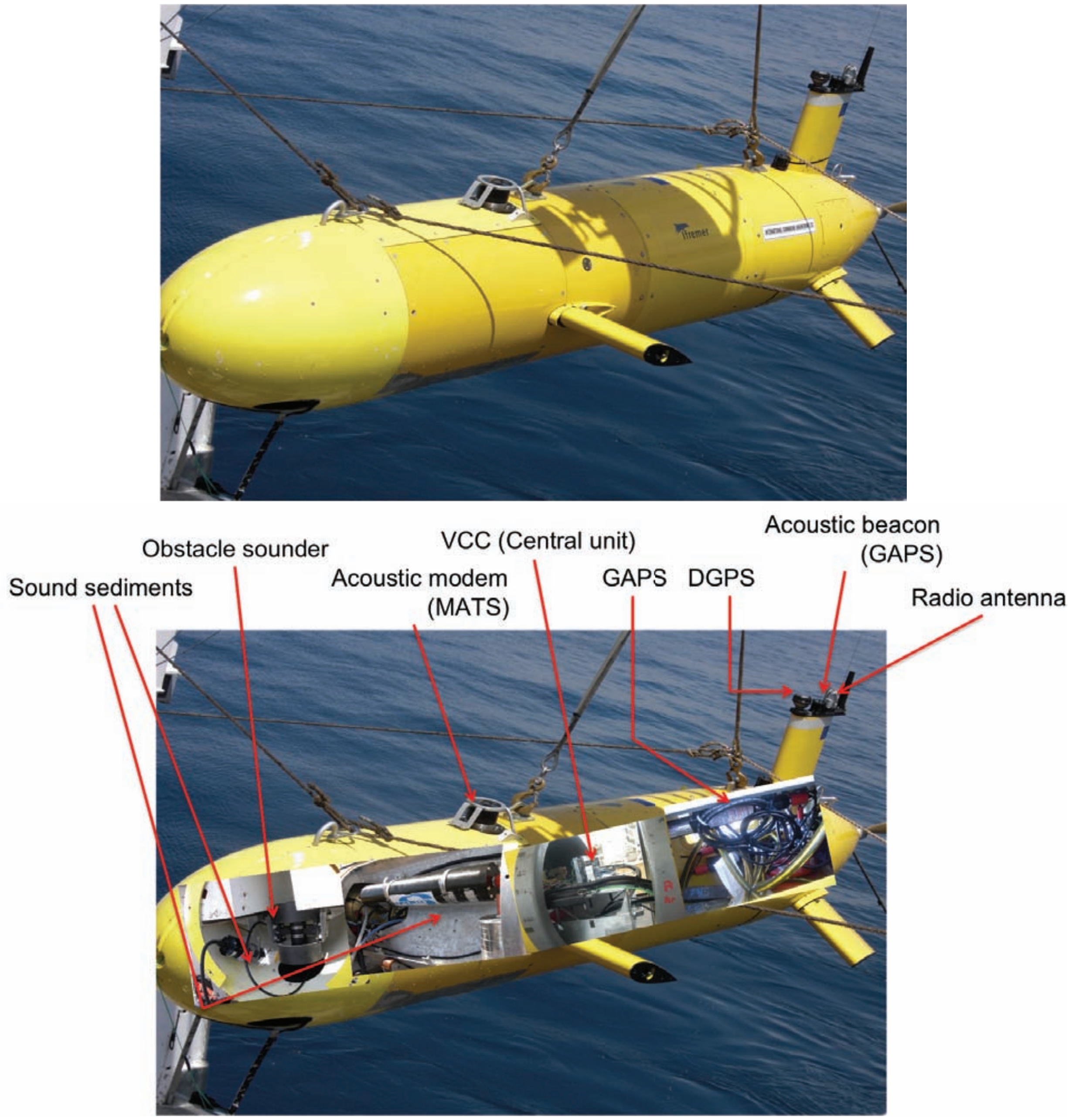

Obstacle sounder
Sound sediments
VCC (Central unit)
Acoustic modem
(MATS)
GAPS
DGPS
Acoustic beacon
(GAPS)
Radio antenna

Figure 2.2 – *The AUV* `Asterix` *used at IFREMER and its components.*

c. Autonomous Surface Vehicles (ASV). These boat vehicles are autonomous vehicles in term of energy and control, but remain at the surface.

For an ASV the localization and communication with the ship is less constrained than for an AUV – the acoustic communications with a diving AUV are however similar to ship/AUV communications.

Figure 2.3 shows the ASV `ROAZ II`, mainly designed for bathymetry, security, search and rescue support operations at the Autonomous System Laboratory in Porto (Portugal).

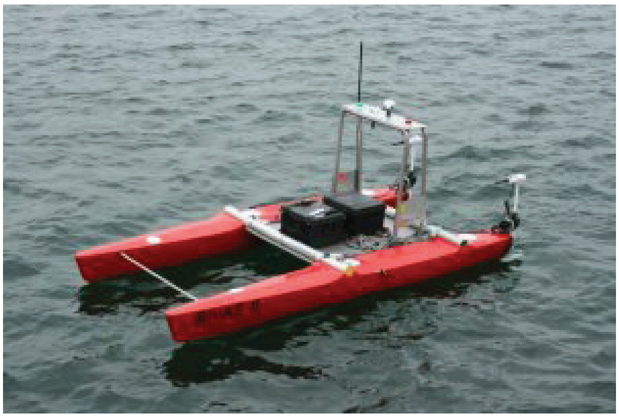


Figure 2.3 – *`ROAZ II`: an autonomous surface vehicle used at LSA's lab*

2.2.2 Types of mission

Underwater vehicles can be used for various missions, which can be gathered into the following three types.

- Exploration missions. Exploration consists in surveying the environment and transmit data back to the surface (e.g. for scientific analyzes). A fundamental challenge in exploration is the limited bandwidth available to transmit data, thus motivating the development of novel techniques for analyzing data on board the underwater vehicles and only transmitting necessary information. The exploration has as objective: targets localization, sea-floor mapping or water sampling.

    1. Target localization consists in localizing particular features on the sea floor, like algal blooms, hydrothermal vents [German08], wrecks, flight recorder systems, etc.
    2. Mapping is used to gather knowledge about the ocean-floor. In general, scientists first use sonar from surface ships to map areas of potential interest. They plant transponders along the ocean floor to allow pinpoint navigation. Using the signals from the transponders a camera is towed along the bottom to automatically take photographs. Finally, scientists study the photos to choose a few places of further observations. Even with all this preparation, scientists do not always find what they are looking for in the darkness of the deep sea. Another kind of mapping is the bathymetry referring to the measurement of ocean depth through depth sounding.

3. Water sampling is used to detect and analyze various phenomena that impact some physico-chemical parameters of water (e.g. vents on the sea-floor can be detected and localized by detecting the associated plume). In general, the AUV has a restricted number of gulpers. This makes the choice of the sampling location important.

- Intervention missions. ROVs are generally used for this kind of missions, that often come to object manipulation. The operators tele-operate the ROV manipulators from the ship and try, for example, to contain oil outflows.
- Formation missions. These types of missions are used to survey a region with each vehicle exploring the area at a certain distance to each other. The formation can be used to re-localize vehicles [Alexander2009], and the redundant collected data can enhance the overall information gathered.

2.2.3 Main constraints associated to underwater environments

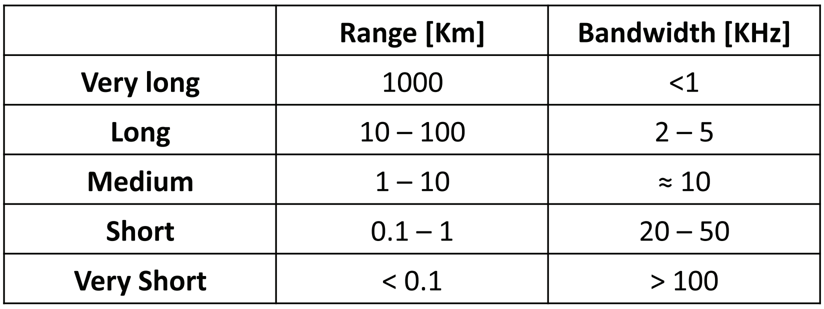
Up to now, most robotics vehicles have been terrestrial vehicles designed to move on land, the various kinds of sensors deployed on these vehicles being adapted to such environments. The environment naturally has an important impact a vehicle design: in underwater environments, besides the vehicle itself, the sensors and communication devices must be adapted to specific constraints that we briefly review here.

Communications The constraints and limitations associated to underwater acoustic communications are essential. They are due to the slow propagation velocity of sound through sea water (around 1500 m/s) and to various phenomena such as scattering that deviates the signal propagation, or shadow zones that can hinder the signal propagation. As a consequence, acoustic communications have limited bandwidth (few tens of bits/s up to few kbits/s). They are also very sensitive to the presence of noise (ambient and generated by the vehicles motors and propellers), to the vehicles relative positions, and to the environmental conditions (pressure, temperature, etc.).

An acoustic communication channel can only be established under particular conditions, and especially when the two communicating nodes locations satisfy some geometric constraints. Akyildiz et al [Akyildiz04] have studied the relation between the range of the acoustic signal and the bandwidth in underwater vehicles, summarized in Figure 2.4.

In multiple AUVs applications, these considerations imply that the communication network management must take into account the relative configuration of the fleet members when attempting to transfer data, in order to maximize the potential of success.

Positioning While the absolute position of an ASV is easily known thanks to GPS, AUV positioning remains a difficult issue. Inertial navigation systems have witnessed vast improvements, but with time the localization error eventually increases. Absolute positioning can be obtained thanks to dedicated acoustic means. The two main methods include Long Base Line (LBL) systems, or Ultra-Short Base Line systems that enable the estimation of the absolute position of an AUV, ASV or ship. Figure 2.5 illustrates how a ROV can estimate its position using LBL.



	Range [Km]	Bandwidth [KHz]
Very long	1000	<1
Long	10 – 100	2 – 5
Medium	1 – 10	≈ 10
Short	0.1 – 1	20 – 50
Very Short	< 0.1	> 100

Figure 2.4 – *Available bandwidth for different ranges in Underwater Acoustic Channels*

Four transponders are deployed at known locations on the seabed. The positioning technique employed is trilateration, a method that determines the relative positions using the geometry of triangles. As all observations are prone to errors, a positioning system should be designed to minimize the effect of those errors and to eliminate error where possible. Here, three reference points is the minimum number required to compute a unique solution, but the introduction of a additional observations enables the detection of inconsistencies in observed values and the minimization of the computed pose error.

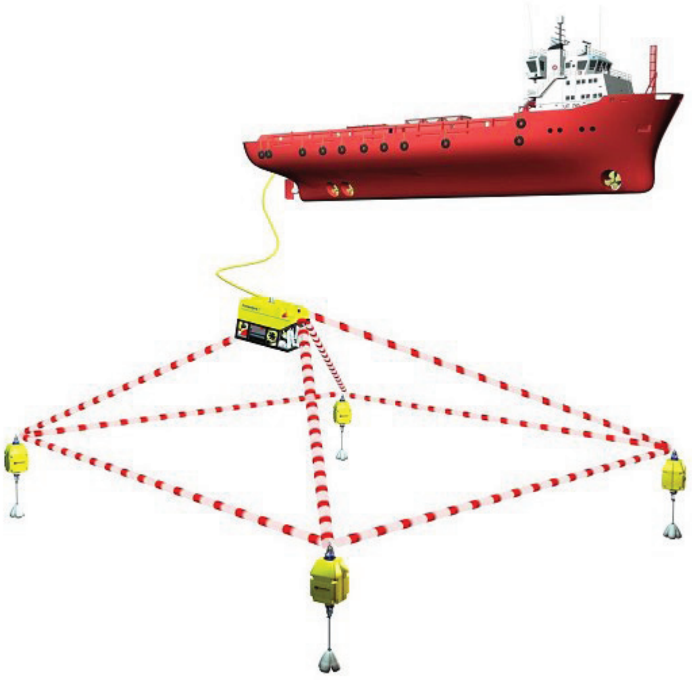


Figure 2.5 – *Illustration of the use of LBL with a ROV.*

Energy limitation Finally, the last stringent constraint for AUVs is energy, which is naturally limited and has to be taken into account while planning and executing the mission. AUVs are equipped with on-board energy monitoring devices.

2.2.4 Levels of autonomy

AUVs are fitted with a control software architecture, that defines their level of autonomy. Classically, these levels are set according to a hierarchy of controls (figure 2.6):

- Mission control: The vehicle is provided with a mission plan and controls the execution of the pre-defined mission goals. In general, AUVs have the ability to control the tasks execution.
- Navigation control: generation of high-level commands (waypoints) and equipment commands (for example data acquisition triggering). This corresponds to external closed-loop control.
- Motion control: generation of low-level commands (heading, speed, depth, etc.) according to the current navigation goal (go from a waypoint to another, go-to a point, etc.). This corresponds to internal closed-loop control. In the major case, ROVs use just this type of control to execute the given commands from the operators.
- Finally the lowest control level is the closed-loop control of the actuators.

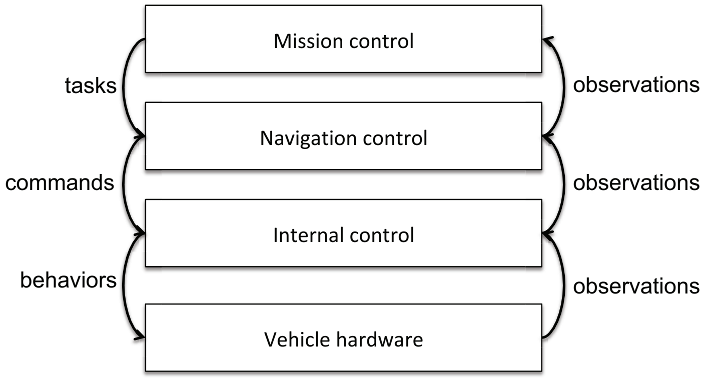


Figure 2.6 – *Illustration of different levels of underwater vehicles' autonomy.*

2.2.5 Overview of AUV related work

In [Ferri08] the authors localize hydrothermal vents using an autonomous underwater vehicle called the ABE (Autonomous Benthic Explorer vehicle, from Woods Hole Oceanographic Institution). The mission of the vehicle is split in two phases: (1) ABE has to move to predefined places (e.g. waypoints), (2) when some conditions are met the vehicle begin to do a spiral trajectory. The authors consider the *Eh* potential[2] anomalies as a signal to trigger spiral movements.

---

[2] *Eh* is the reduction potential, whose variations provide information on the variation of the chemicals in the water

This exploration strategy is interesting as it mixes a pre-defined plan with reactively defined trajectories. But this kind of strategy does not deal with time constraints and does not exploit a realistic vent model.

There are other robotic applications for chemical plume tracing pollution and environmental monitoring, search and rescue and deep sea hydrothermal vent prospecting. In [Sinha09], the authors estimate the shape of the dispersion of a chemicals pollution. The pollution represents a transparent gas of nuclear, biological or chemical contaminants. Some contributions use other plume parameters to localize existent plumes [Christopoulos05] or construct a maps of probable target location [Pang06]. These solutions do not address the problem when the number of targets is unknown.

Farrell et al. [Farrell03] experiment a tracing plume algorithm implemented on a REMUS AUV. The proposed algorithm is defined on the basis of six interchangeable behaviors according to chemical detection events and timeouts.

A different approach to plume source localization followed by some authors requires the estimation of the parameters of a model of plume formation, which is exploited to assess the source location. The results quality depend strongly on the realism of the model that represents the environment.

Ishida et al. [Ishida01] developed a terrestrial system that estimates the parameters (including source location) of a time-averaged model of plume dispersal in a uniform propagation field. Their robot is able to successfully locate an ethanol source a few meters away from its starting location in several minutes. The slow convergence time and limited range are a consequence of the time required to average concentrations in the actual plume to converge to those predicted by the model. Christopoulos et al. [Christopoulos05] describe an algorithm for optimally adapting the robot trajectory to estimate the parameters of a diffusion model of plume evolution and present simulation results.

Several approaches to chemical source localization specific to hydrothermal plumes have been proposed. Veirs et al. [Veirs99] propose a method based on CTD measures. This method successfully identifies the locations of several known vent fields and suggested locations that might contain undiscovered sites.

The authors in [Dearden07] localize a vent according to several measures and a predefined model. First, the vehicle explores the area using its predefined model and POMDP (Partially Observable Markov Decision Process) approach to choose the next cell to explore. Second, the scientists analyze the collected data and evaluate the target location. In the case where the scientists do not localize the target, the vehicle can do another mission.

Jakuba et al. [Jakuba08] localize a hydrothermal vent by inferring the location of the sea floor vent from the water column measurement. This includes an exploration task and at the same time estimate the location of the target. The authors use a grid to represent the environment. Rather than localize one target, they try to localize a maximal number of target using different map updates.

Patron et al. [Patron08] consider that the vehicle can learn from its measure: according to a measure the vehicle can evaluate a plan (more precisely an action). The authors try to find a balance between a data driven and a task driven approaches: they use the collected data to evaluate an action. This approach is an adaptive mission planning that deals with time.

2.2.6 Synthesis

Missions using underwater vehicles have focused on exploration, target localization, water sampling, etc. In most of the cases, the exploration strategies are predefined (pre-planed) and the vehicle tries to execute the pre-defined mission as closely as possible. In other cases, the plan adaptation is kept simple.

As compared to a single robot, multi-robot fleets bring robustness with respect to robot failures, allow to achieve more efficient missions (more rapidly, or with a greater spatial extent), and to fulfill missions or tasks that intrinsically require the use of several robots. When it comes to information gathering missions, *e.g.* exploration, surveillance, target detection and localization, synergies naturally occur if the robots effectively communicate to merge information gathered on the environment and to coordinate their observation plans. A large amount of work has been done in the robotics community, and the literature abounds with contributions on communicating robot fleets – see *e.g.* [Konolige03, Burgard05, Howard06]. In the next section we review the main used approaches for multiple ground or underwater vehicles.

2.3 PARADIGMS FOR DISTRIBUTED INTELLIGENCE

Various types of distributed intelligent interacting systems have been proposed in the literature. Distributed intelligent systems are inherently multi-agent systems [Woolridge01] although not always described as such. Some interactions schemes are inspired from insect societies, human behaviors, etc. On the basis of the classification proposed by [Parker08], we outline here three paradigms for distributed intelligence.

2.3.1 Bio-inspired paradigms

Bio-inspired paradigms are related with social behavior and emergence. They often involve specifying a set of simple rules that are iteratively applied and that implicitly specify an organization. As an example, ants search for food by first randomly exploring the environment. Once an ant found food, it returns to its colony while leaving pheromones on its way. Repeatedly, other ants follows the trail and also leave pheromones when returning back. Pheromones evaporate over time: the longer it takes for an ant to travel a path, the more the pheromones evaporate. As a result, a shortest path from the colony to the food place is eventually defined.

Parunak et al. [Parunak01] use a technique based on the quantity of sensed pheromone in the environment to track targets. Here, each vehicle puts and/or evaporates pheromone and choose the next cell to explore according to this quantity of pheromone. Such approaches are called “sigmergy”, and have been adapted for robotics, e.g. in [Kube93] or [Masao94].

2.3.2 Organizational and social paradigms

These paradigms are derived from the way humans organize themselves, as studied in the fields of sociology or economics. Several organizational theories are applied to robotics:

Organization by assigning roles. Each robot has a specific role in the robot community, where each role defines a specific task to accomplish. With each assigned role, each robot has specific tasks to achieve. For example, in multi-robot soccer [Veloso98] each robot has a predefined assigned role as left defender or right defender.

In another work, [Kim08] defines optimal groups of mobile robots (ad hoc) that can cooperate. The approach has to define groups and select their leader. They use the notion of entropy to define the relation between the robots, whose organization depends on transmission range [3]. The more the robot is in the communication range of another robot, the higher the probability that it belongs to its group and the smaller the entropy between vehicles inside the group is. At the end, a group leader is selected based on optimizing power consumption, where the role of the leader is to to give order to the other vehicles that has to execute the task. Burgard et al. [Burgard05, BurgardJ05, Burgard02] propose an algorithm where each vehicle computes a utility and a cost to go to a target. This utility is obtained by the distance to that target. At the end, each vehicle will be assigned to a nearest target. These approaches can be considered as a role assignment, when a target is assigned to a robot, the role of the robot is to localize it.

Market economies. In general, Marked-based approach (*e.g.* in [Stentz04] or in [Goldberg02]) are used to allocate tasks for each robot by using measures. This measure can be the contribution of a robot to a specific task, or can be represented as a cost function. For instance, the use of a market-based approach specifically for multi-robot task allocation was developed in the M+ architecture [Botelho00]. In the M+ approach, robots plan their own individual plans for the task they have been assigned. Then, they negotiate with other teammates to incrementally adapt their actions to suit the team as a whole, through the use of social rules that facilitate the merging of plans.

Moorehead et al. [Moorehead01], to explore unknown areas, use also an approach that optimize a utility/cost ratio. Ogren et al. [Ogren04] present a control strategy for a team of vehicles to move and rely on a gradient descent (potential field).

In [Tambe02], the authors consider the problem of disaster mitigation in the RoboCup Rescue Simulation [Kitano99] to allocate tasks. The authors use a greedy search method to explore the world, where each robot has to visit nodes the least number of times.

Meier et al. [Meier06] use a polygon to simplify the information transmission to the other vehicles. The main used algorithm assigns a target to a robot according to its distance (the closer the robot is near to the target, the more it can be assigned to go to this target). The authors reduce communication between vehicles by using a geometrical reasoning. Similarly, the authors in [Konolige06] were interested by the unknown region mapping problem. The authors divided the area into static [4] clusters, where each robot belongs to a cluster and can exchange information only with vehicles that belong to the same cluster.

---

[3] The entropy is computed on the basis of the distance from one vehicle to the other. The higher the distance is, the less the transmission range is, than the higher the entropy is between two vehicles

[4] The cluster is fixed and over time cannot be rebuild.

2.3.3 Knowledge-based paradigms

The knowledge-based approach aims at sharing the knowledge of each robot to develop the system evolution. There has been considerable work done on target localization by robot teams. In [Stroupe05], each robot motion is defined on the basis of the current knowledge of the targets locations, where regularly each robot broadcasts its observations.

The knowledge-based approach is also used for task allocation in multi-robot teams, such as in the ALLIANCE approach [Parker98], in which robots model the ability of team members to perform the tasks of the system by perceiving team member functioning and collecting relevant task quality, such as time to task completion. Robots then use these models to choose tasks to achieve having as objective to benefit the group of robots.

In general, these approaches need permanent, long range and large bandwidth communications, three properties that are not satisfied with underwater vehicles.

Stone et al. [Stone99] introduce Periodic Team Synchronization (PTS). In robotic soccer, teams can plan strategies before the game, at halftime, or at other breakpoints, but during the course of the game, communication is limited. This synchronization for the authors [Stone99] is called PTS. Their work consists of a general team member agent architecture suitable for creating teams of agent in PTS domains. This architecture define pre-determined multi-agent protocols accessible to the entire team, called "locker-room agreements". They define a flexible teamwork structure that allows for task decomposition and dynamic role assignment in PTS domains. When there is no communication the agent acts autonomously.

2.3.4 Synthesis and problematic

Numerous studies propose interesting architectures for robot cooperation. For bio-inspired paradigms the coordination between vehicles is implicit. The communication bandwidth is not taken into account by organizational and social paradigms. The same thing for knowledge based paradigms, where broadcasting each new collected data can affect the effectiveness of AUVs mission.

Nevertheless most of them are focused on terrestrial robots for which the communication among vehicles is, in most cases, reliable and permanent. For underwater vehicles, it is a different story. The water acoustic greatly restricts the communication range and bandwidth. As a result, the general robot cooperation problem has to be reconsidered to take this particular constraint into account.

2.4 MULTIPLE UNDERWATER VEHICLES EXPLORATION

Fleets of AUVs have the potential to explore the ocean and increase the spatial and temporal coverage of a single research team. The use of various AUVs allows new types of missions but also raises new challenges for communication, coordination, navigation and localization.

Additionally to the constraints mentioned in section 2.2.3, the communication with the surface, energy limitation and the embedded control, there is a constraint of communicating between underwater vehicles. It may be required that each AUV has information about each fleet member position. This can be the absolute position of each vehicle, or the relative position between all.

For instance, when using acoustic communication devices, the time of signal flight measurement can give range and possibly bearing for navigation purposes during the AUVs coordination. Considering the fact that in an heterogeneous fleet of AUVs, the navigation accuracy of each individual member may vary from others, the information exchange between the vehicles may improve the positioning of vehicles featuring less performing navigation systems by benefiting from the ones able to estimate more accurately their position.

Additionally to that, the communication allows each vehicle to exchange collected data for a better mission execution (e.g. target localization).

2.4.1 Exploration strategies with task control

The project MAUV (Multiple Autonomous Underwater Vehicles [Herman88]) controls several vehicles. The architecture is divided hierarchically into several layers: the mission layer, the group layer, the vehicle state layer, the e-move layer, the primitive layer and the servo layer. The highest layer is the mission layer, that decomposes the mission into a set of commands for a fleet of vehicles. The other layer is the group layer that assigns for each vehicle the associated task. The vehicle state, the e-move, primitive and servo are also other layers dedicated each one to do one task in a hierarchical way. In this project the communication between vehicles is intensive, when each vehicle measures a modification of its environment, it has to communicate it to others. But in general, the communication between vehicles is hard and cannot occur at any time. This is why the authors propose to make a leader team for each fleet of vehicles and communications occur just between leaders.

The objective of CoDA (Cooperative Distributed Autonomous oceanographic sampling networks control Turner [TurnerJ01]) is to make a cooperative protocol for intelligent control. Several autonomous underwater vehicles (AUVs) and other instrument platforms are gathering data over long term in an area. In this article two levels are designed, called: MLO (meta-level organization) and TLO (task-level organization). The MLO is a loose organization of multiple AUVs that self-organizes to analyze the mission and the resources available and the TLO is focused on efficient organization to carry out the mission. So the MLO will most of the time looks like a consensus-based group of vehicles, while the TLO might be a hierarchy, a committee, a team, a market, or any other organizational type that seems to fit the situation. This protocol design is interesting in term of the freedom to choose the desired protocol of cooperation. In fact, the proposed protocol is scalable but each vehicle has to broadcast its organization when it enters in the MLO. This is why this protocol can induce a bottleneck that can affect the waiting time of each vehicle, then for the whole mission execution.

Rahimi et al. [Rahimi04] propose an approach that randomly samples the environment, in a systematic way which does not take into account any measured data.

The authors in [You05] investigate their efforts on developing a market based framework for multiple underwater vehicles. In this framework, the tasks are allocated to each vehicle for the winner of the cautions.

Finally, much research deals with the use of wireless sensor networks for target detection, localization and tracking [Liu02, Biao08]: the nature of the considered sensors (often bearings-only or range-only) turns the problem into an estimation problem and a deployment problem where the quality of the localization depends strongly on the sensor placement. Such systems are suited to monitor dynamic events, *i.e.* that evolve overtime.

2.4.2 Exploration strategies with navigation control

In [Maurizio07], each vehicle controls its trajectory in a cooperative manner, with the objective to maintain the desired formation and drive the fleet to a desired average value. Two approaches are developed, the first needs a common virtual leader, while the second needs decentralized estimation of the virtual leader by each vehicle. The control is here a two-level consensus problem. Each agent reach agreement on the virtual leader state at one level and at the other level reach formation about the virtual leader. The decentralized approach is effective even when communication among agents is limited.

Contributions that deal with multiple AUVs (sometimes involving ASVs) deal with various tasks. [Haraksim09, Brignone09] deal with *formations*, in which the objective is to make the vehicles follow each other, while maintaining permanent communications.

A leader-follower algorithm is used for AUV formation by Edwards et al. [Edwards04]. The leader is the one that broadcasts the collected information and makes decisions. The mission is a navigation in formation, where the leader controls the distance between vehicles. In the case where the leader is not available each vehicle acts autonomously using the inertial baseline information. Also in the same research area, [Fiorelli06] or [Bhatta05] describe a methodology for cooperative control.

The authors in [Sotzing08] define a control architecture for a fleet of underwater vehicles. The coordination between vehicles is given by broadcasting all the collected data to all the vehicles. As a result, this architecture does not deal with restricted bandwidth.

In the project GREX [Ghabchello09], the objective is to coordinate the trajectory of each vehicle with the other. The mission is the exploration by formation, where a minimal and maximal distance between vehicles has to be respected.

2.4.3 Adaptive exploration strategies

The authors in [Zhang08] propose an adaptive sampling strategy for a team of ASVs. The approach relies on the partition of ”equal gain” areas, that are then explored by individuals.

Popa et al. [Popa04] developed an adaptive sampling algorithms, that uses information measured to direct the vehicle to the most likely information about the target. The authors use a routing algorithm to minimize the motion cost of the vehicle. At the same time, the vehicle has to compare its entropy, before and after moving. The adaptive sampling algorithm will then seek to sample at a new location such to minimize the cost function.

The authors in [Meliou07] present a new non myopic algorithm, that uses the collected information. In general, the authors take a greedy algorithm and they transform it into a non myopic by adding values on the path of each vehicle.

Low et al. [Low09] drive the AUVs according to the sensed measures, directing the vehicles to the detecting targets: the approach yields finer localization of targets than systematic sampling.

The authors in [Low08] describes an adaptive multi-robot exploration strategy for performing both wide-area coverage and hotspot sampling using non-myopic path planning, based on a dynamic programming formulation called MASP (Multi-robot Adaptive Sampling Problem). They apply Gaussian and log-Gaussian processes, and analyze if the resulting strategies are adaptive and maximize wide-area coverage and hostspot sampling. The robot chooses the next cell to explore by maximizing the information gain. Only one robot can choose to sample a new location at each stage while the rest of the robots remain still, a sequential approach.

2.5 OVERVIEW OF OUR APPROACH

In this chapter we have reviewed different types of underwater vehicles and their main characteristics with respect to terrestrial vehicles. Underwater vehicles are subject to several constraints, the most important being on the communications. Thus, the task planner of these vehicles needs to take into account these communication constraints. But in general if there are several tasks to achieve the planner can take a long time to plan and re-plan for the whole mission. In this sense, dividing the system into subcomponents can be a solution for this temporal reactivity problem.

Another objective is to improve the efficiency of the mission, i.e. is the number of found targets in our scenarios. This efficiency depends on the way the vehicle explores the area. For target localization, the location of targets are unknown, predefined missions can sometimes be inefficient. Our proposed solution is to make underwater vehicles *adapt their exploration mission according to their perception*. This kind of approaches are called reactive approaches, while most of the proposed approaches in the AUV literature does not take into account the measured data during the mission execution.

The trade-off is then to find an architecture that uses a task planner and at the same time remains reactive. We propose to divide the system into several subcomponents with different deliberation times, the higher the deliberation time is in a subcomponent the lower the reactivity is.

To improve the mission execution time, we choose to use several underwater vehicles that share their data to localize targets. The data can be exchanged only at communication points with low rate.

This is why the general embedded system for underwater vehicles requires :

- An embedded control architecture that controls at any time the mission execution.

- An embedded mission planner that can deal with time. It is important to control the execution of the mission by using a temporal delay, that helps when we deal with a cooperation approach. For example, a communication between two vehicles, using an acoustic channel, can only occur if the vehicles are at a certain distance to each other and at the same time. It is also interesting to have different deliberative time for each subcomponent, that makes the subcomponent more reactive or more deliberative.

- A hierarchical architecture. To improve the reactivity of the system, we choose to divide the system into subcomponents that are able to work separately with different role. This division is interesting to improve the vehicle reactivity.

- A reactive approach. Each vehicle can be able to react with its environment at any time during the mission execution.

To validate our cooperative architecture, we applied it to the target localization problem. In our case, the target is a hot spot characterized by a hot temperature: the model of the target and the way to use the collected data to build a map of target presence and to localize them are presented in the next chapter, and the way to adapt the trajectories to the the built map in order to maximize the number of mapped targets is presented in chapter 4.

3

BUILDING TARGET MAPS

This chapter presents the way the perceived information are gathered to build a map in which the probabilities of target presence at given locations is encoded. The model of the considered targets and sensors are given, and the approach to fuse the data is presented. Illustrations are provided, and the mapping process with several AUVs that sporadically communicate is considered.

BROOKS SAYS: «The world is its own best model.»

3.1 INTRODUCTION

In an exploration and mapping mission, the representation of the environment is naturally of essential importance. On the one hand it is the goal of the mission, and in the other hand it is the mean to define the exploration strategies (figure 3.1). To build such maps, models of the observed phenomenon (the targets) and of the sensors are required : these models are respectively presented in sections 3.2 and 3.3. Section 3.4 presents the process that integrates the gathered data into a target map, that represents for each position of the environment the probability that it contains a target. Finally, setions 3.5 and 3.6 illustrate the proposed approach in the single robot and multiple robot cases.

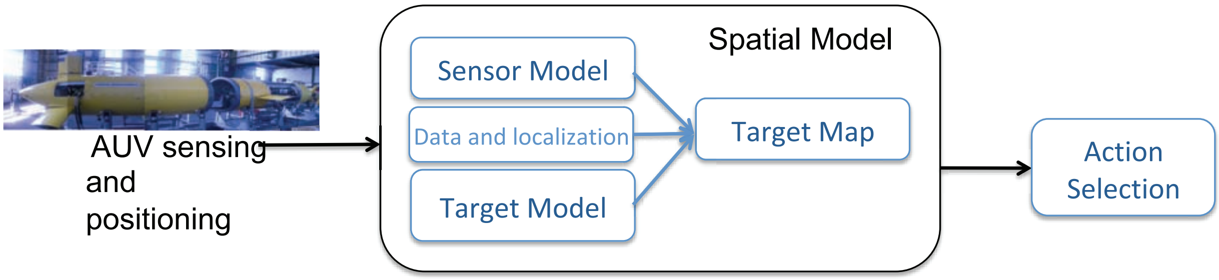


Figure 3.1 – *Illustration of the mapping process.*

3.2 REALISTIC TARGET MODEL

Targets can be static (e.g. flight recorder, hotspot without marine current), mobile (e.g. fishes) or static with mobile properties (e.g. hydrothermal vents). The model of the target is important for defining how to choose the next action. In our case, the target is a vent on the seabed, emitting either clearwater, hot water, chemicals, etc. The emission of the vent expands as it raises higher from the sea floor. Hydrothermal vents are the result of thermal and chemical input from hot spring systems in the oceans, due to magma flows. When cold sea-water contacts with the magma, it dissolves minerals and metals from the nearby rocks and forms an hydrothermal vent.

The venting fluid is combined with cold sea water, and continues rising until it reaches a height where it has the same buoyancy as the nearby water (see figure 3.2).

The intrinsic properties of hydrothermal vents have interested researchers for several reasons: (1) the movement of hydrothermal at the seawater through the oceanic crust influences many geological and oceanographic processes such as loss of heat from the earth, geochemical cycling of the elements, biogeochemistry of deep ocean waters [Lonsdale77] and (2) the biology of hydrothermal vents gives special features [Ferri08] some of them are the presence of manganese, iron and other metals.

Hydrothermal vents are hard to localize for the following reasons:

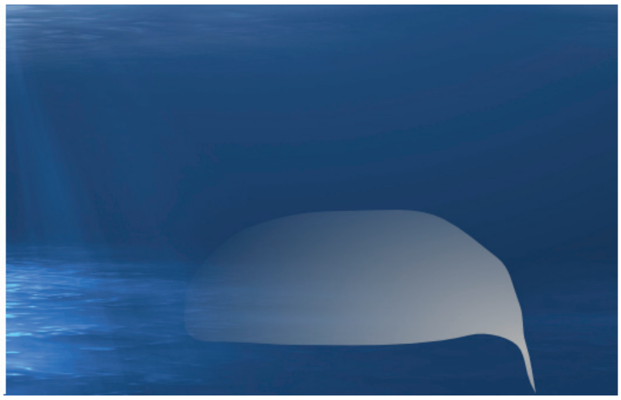


Figure 3.2 – *Illustration of the diffusion model of hydrothermal vent with marine current.*

1. Propagation properties: The vent is mixed with sea water and the temperature, which smoothes the difference from the adjacent water.

2. Current: The current flow is not constant, and can change its direction over time and depth.

3. Water density: Sea water often has different densities for different vertical layers and hydrothermal fluid can not easily cross these layer boundaries. Hydrothermal vents can sometimes become trapped at a depth depending on the water density.

In general, searching for vents is made as follows:

- First, the location of the target is detected with low resolution (Global exploration strategy). In order to do this, bathymetry data is required, because it can be used to identify likely locations for vents based on the topology of the area, and can guide the deployment of sensors. Bathymetry data is captured on the boat using bathymetry sensors that are able to map a region several kilometers wide.

- Vents are then precisely located by using underwater vehicles ROVs/AUVs. These vehicles use adequate sensors to track down vents (see section 3.3).

We consider the general case, where the property of the molecule's concentration at the hotspot is very dense, and where the propagation of these molecules depends on the water density, the location and the instant. The equation of the fluid diffusion relies on the diffusion coefficient and the water density as follows [Philibert06]:

$$\frac{\partial T(r,t)}{\partial t} = \nabla[D(T,r)\nabla T(r,t)] \tag{3.1}$$

where, $T(r,t)$ is the diffusion density of the temperature at location r and time t. $D(T,r)$ is the coefficient of the diffusion at the location r. In the case where this coefficient depends on the density, the equation (3.1) is nonlinear.

Several studies aimed at modeling hydrothermal vents. For example, Stracey et al. [Stracey00] express the plume concentration as a function of radial distance from the source and the coefficient of dispersion.

3.3 SENSORS

3.3.1 Existent sensors for target localization

There are various kinds of sensors that can be used for vent localization [Baker95]:

1. Optical backscattering: When the hydrothermal fluid meets cold sea water, many of the minerals precipitate into particle form, making the vent water very cloudy. The particle concentration in the water can be estimated by measuring the amount of light reflected back using a Light Scattering Sensor (e.g. LSS developed at IFREMER [Vangriesheim92]).

2. Methane: Dissolved methane is an unambiguous indicator of vent activity and is easily measured.

3. Reduction potential (often referred as Eh): Redox potential measures the chemical reactivity of the water; hydrothermal water has low redox potential, as it is low in oxygen. Eh is useful as strong changes in it are only observed close to a vent, within a few hundred meters.

4. Manganese, iron, and other metals: Manganese concentration in particular is enriched by several orders of magnitude in hydrothermal fluid compared to normal seawater.

5. Potential temperature versus salinity anomalies: Conductivity, Temperature, and Depth is a sensor that measures water temperature, salinity, and density. While the temperature of hydrothermal fluid far from the vent is indistinguishable from that of the surrounding water, it is identifiable by a change in the linear relationship between potential temperature and salinity in a given region of ocean. Such anomalies can be found by examining a series of CTD data, as salinity and potential temperature are just functions of conductivity, temperature and density.

3.3.2 Classification of existent sensors

For external sensors, we can define three family of sensors: range-only sensors, bearing-only sensors and the range-and-bearing sensors.

1. Range-only sensors: This type of sensors perceive the distance to the target. As an example, we can take the scenario of target localization (e.g. vent), where we know the maximal temperature of the target. Using the vent model and the measured value, the vehicle can compute the distance to the target.

2. Bearing-only sensors: This type of sensor indicates the direction of the target without knowing its distance. Several types of bearing-only sensor exists in the literature (e.g. vision), in our case we are going to use a CTD sensor in the case where the maximum temperature of the vent is unknown. This sensor can measure the temperature of the place and according to the past and present value of the temperature, the vehicle can infer the target's direction (see section 3.4.2 for more details).

3. Range and bearing sensors: For example sonar (sound navigation and ranging) provides to the vehicle, in one measurement, the distance and the direction to the target. This kind of sensors are used for bathymetry. Sonars propagate the sound in the water to detect the contents of the water, for vehicle's localization using LongBaseLine sonar or for fishing. According to the transmitted signal and the taken time to get back the signal, the sonar allows the vehicle to compute the distance and bearing between its position and the target/ground/fishes.

3.4 TARGET DETECTION MODELS

We defined two target models: in the first model, the maximal temperature is known, whereas in the second one the maximal temperature is not known.

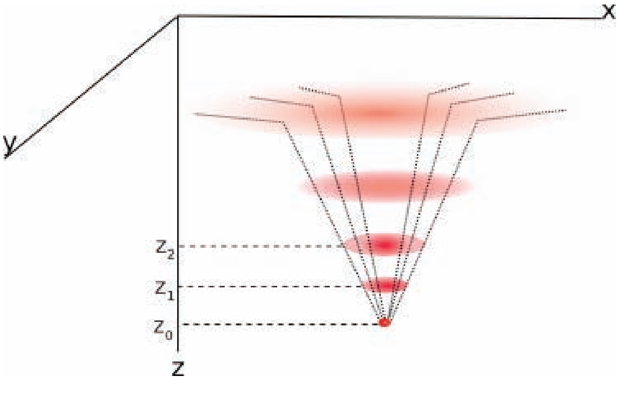


Figure 3.3 – *Illustration of the temperature evolution within a thermal plume in stationary waters: the temperature, here represented in red (the redder the hotter), decreases with the elevation and with the distance to the vertical of the emitting vent.*

3.4.1 Target detection model with known maximal temperature

We consider the case of hot spring localization according only to the temperature properties. The density of the temperature T within the plume is a decreasing function of the horizontal distance ρ with respect to the plume center and of the elevation z above the seabed. This function is the model of the plume, which is an approximation of the actual diffusion phenomenon: the model is a probabilistic one, that expresses the probability density function (*pdf*) of the temperature T as a function of the distance ρ and the elevation z:

$$P(T = t|\rho, z) \tag{3.2}$$

Figure 3.4 shows the behavior of the model for two different elevations: the dispersion of the temperature is also an increasing function of the distance and the elevation. Note that multiple sources do not interfere if they intersect, the water temperature being a function of the closest source (this assumption would not hold in the case of the emission of chemicals, whose concentration augments within the intersection of plumes: such cases call for a different modeling of the *pdf*).

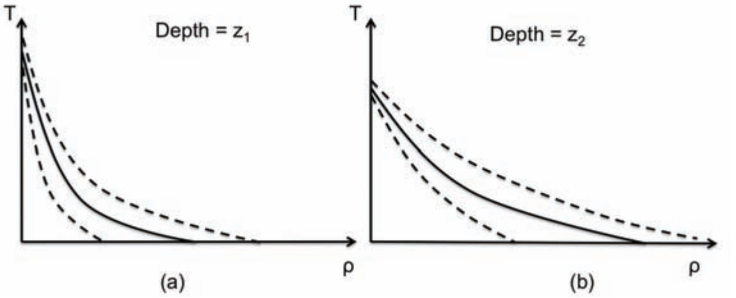


Figure 3.4 – *Evolution of the temperature as a function of the horizontal distance ρ for the two depths z_1 and z_2 defined in figure 3.3. Dashed line represent the dispersion of the temperature.*

$$T_{mean} = \begin{cases} T_{max}(z) - \rho \frac{T_{max}(z) - T_0}{\rho_{max}(z)} & \text{if} \quad \rho \leq \rho_{max}(z) \\ T_0 & \text{if} \quad \rho > \rho_{max}(z) \end{cases} \tag{3.3}$$

where

$$\begin{aligned} \rho_{max}(z) &= \alpha z \\ T_{max}(z) &= \begin{cases} T_{MAX} - z \frac{T_{MAX} - T_0}{z_{MAX}} & \text{if} \quad z \leq z_{MAX} \\ T_0 & \text{if} \quad z > z_{MAX} \end{cases} \end{aligned} \tag{3.4}$$

These equations model the plume as a cone, the parameters Z_{MAX} and α respectively defining its height and aperture. This is certainly a simplification of the actual diffusion phenomenon. In particular, it makes the assumption that there are no current: a stationary current independent of the depth would simply generate an oblique cone, and the consideration of dynamic currents that are a function of depth would require a more complex parametrization.

In the case were there is a marine current. For a permanent and constant speed of the vent at all target altitudes, the target model is represented in the figure 3.5. The distance “G_1” represents the gap between the real location of the target and the actual target properties.

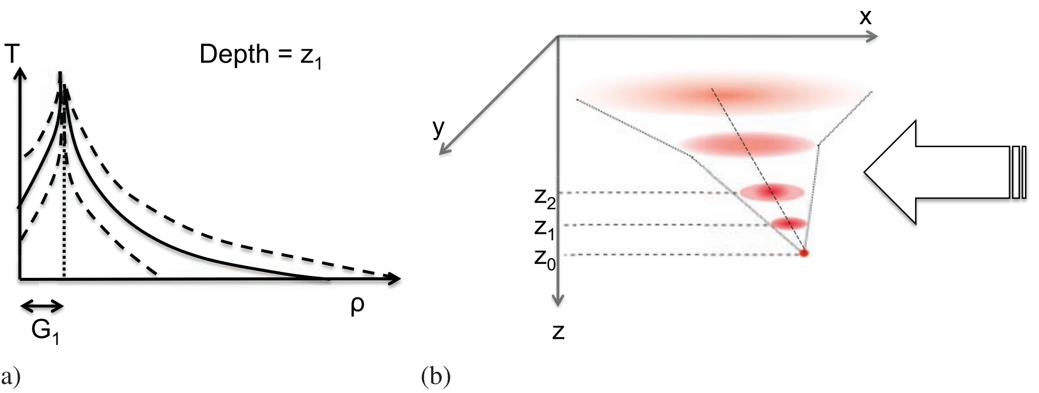


Figure 3.5 – *(a). Evolution of the temperature as a function of the horizontal distance* ρ *for one depth* z_1. G_1 *represents the gap between the hottest temperature and the real position of the target. (b) Illustration of the real target properties propagation at different depths with a steady current.*

When the marine current is not known (that is, in general, the case) by vehicles, a fleet of underwater vehicles can be used. At different exploration depths and sharing the collected data, underwater vehicles can find the corresponding gap (we give more details on the chapter 6).

Finally, for a given position (ρ, z) in the plume, the probabilistic variations of the temperature is modeled by a Gaussian: $T = \mathcal{N}(T_{mean}, \sigma(\rho, z))$, where $\sigma(\rho, z)$ is an increasing function of ρ and z.

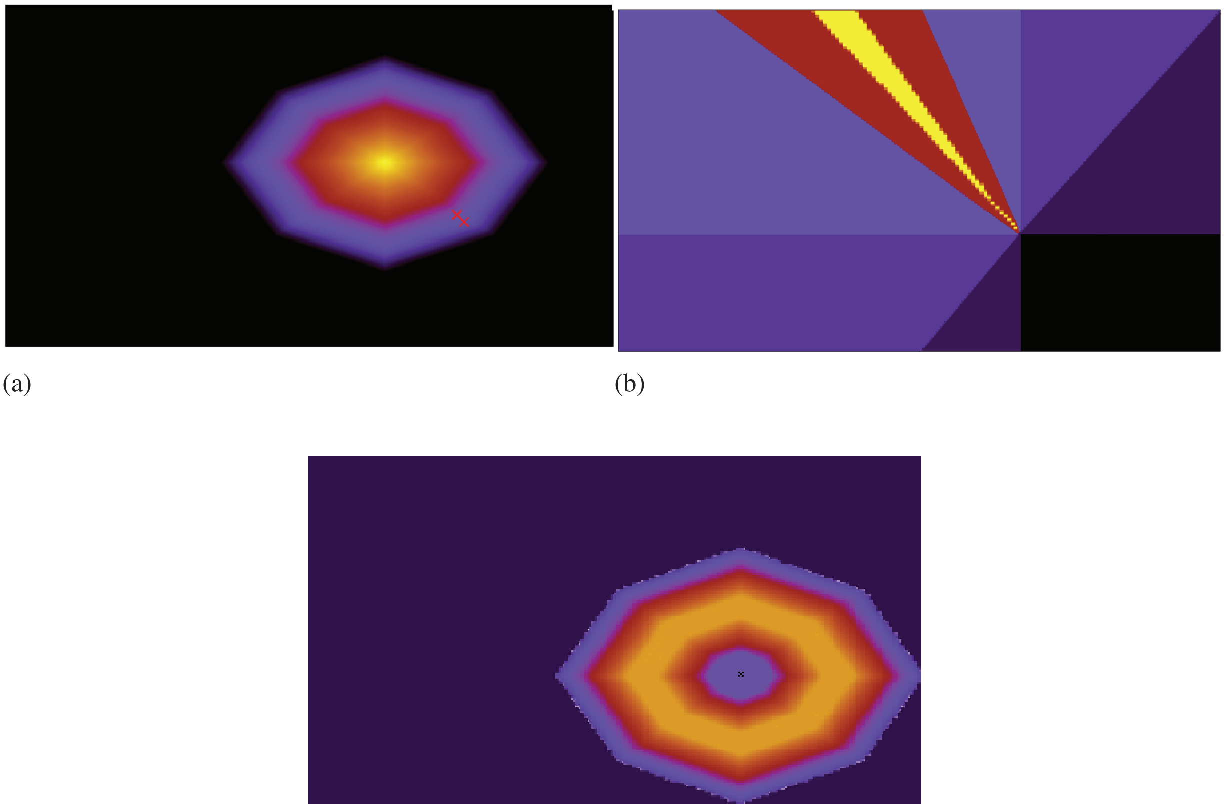
We can model temperature sensors with a probability density function $P(T_{sensor}|T)$ that models its errors (*e.g.* a Gaussian). The overall source perception model is a convolution of the source and sensor model, which results in a *pdf* akin to (3.2), the associated variations being "blurred" by the sensor model– we therefore neglect the sensor errors (since most of the uncertainty comes from the model of the observed phenomenon, not from the sensor errors).
Knowing the maximal temperature of the target is equivalent to the use of a range-only sensor: on the basis of temperature measure the vehicle can compute the distance to the target.

3.4.2 Target detection model with unknown maximal temperature

In this case, T_{MAX} is not known: according to two temperature measures a direction of the target can be computed (we consider this example as if the vehicle has a bearing-only sensor). The direction of the target is computed by the algorithm 1, where the difference between the precedent and the current measure yields an estimate of the target direction.

The variable $Direction$ contains the direction of the target. The more the distance to this direction is, the lower the probability to contains the target is (Figure 3.6).

The used temperature model of the target is represented in figure 3.7. The direction has an error that augment with the distance from the target direction. The model function is expressed by the probability density function (pdf) of the temperature T as a function of a direction (*dir*) and the elevation z:



(a) (b)

(c)

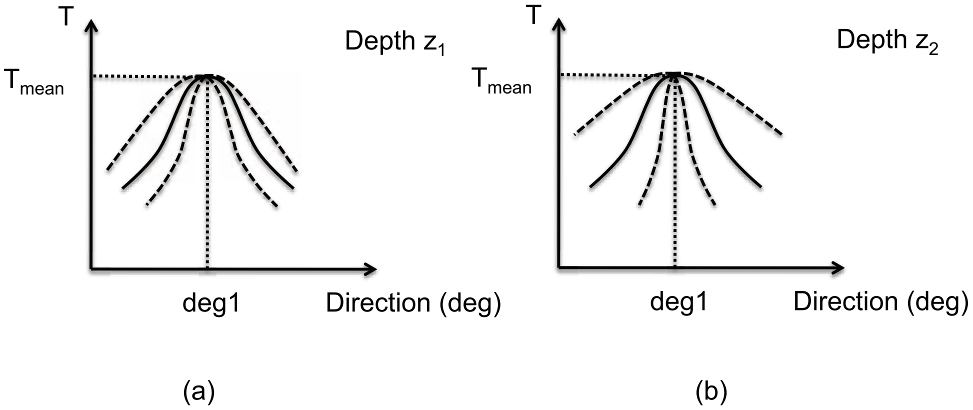
Figure 3.6 – *(a). Illustration of the target model at a given depth. The colors here indicate the temperature. (b) and (c): application of the bearing only and the range only target detection models. The colors represent here the probability of the target presence (The detection models are represented in a grid structure, see section 3.4).*

Algorithm 1: Algorithm to compute the target direction.

```
Require: $T^k$: the actual measured temperature;
    $T^{k-1}$: the precedent measured temperature;
    $Direction$: the direction of the target;
    $Pos^k$: the actual position of the vehicle;
    $Pos^{k-1}$: the precedent position of the vehicle.
$\delta T \leftarrow T^k - T^{k-1}$;
if ($\delta T > 0$) then
    $Direction \leftarrow \overrightarrow{Pos^{k-1}Pos^k}$
else
    if ($\delta T < 0$) then
        $Direction \leftarrow \overrightarrow{Pos^kPos^{k-1}}$
    else
        $Direction \leftarrow Pos^{k-1} \perp Pos^k$
    end if
end if
return Direction
```

$$P(T = t|dir, z) \tag{3.5}$$

This model is specified by a simple Gaussian where $T = (T_{mean}, \rho(dir, z))$.



(a) (b)

Figure 3.7 – *Evolution of the temperature as a function of degrees "deg" for two depths z_1 and z_2 (defined in the figure 3.3).*

3.5 MAP BUILDING

A common representation to map the environment is the occupancy grids [Elfes89]. The environment is subdivided into a grid. To each grid square, a probability that the cell contains an obstacle [S.Moorehead01] or a target [Low09] is associated. The probability meaning depends on the mission type.

In our case, the AUVs gather temperature measures, they are fused into a map that represents the probability of the target locations. The map is a $N \times M$ grid, that discretizes the environment into a set of cells $\{x_{i,j}\}$, $i \in [0,N[$, $j \in [0,M[$. We do not use a volumetric voxel representation because on one hand this would be too costly to memorize, and on the other hand the AUVs can be constrained to evolve within a small set of depths P, P 2D grids are therefore defined. Each cell $x_{i,j}$ contains two types of information:

- A boolean value, equal to 1 if it has been visited by one AUV,
- The estimated probability $P^k(x_{i,j})$ at time k that the cell is at the vertical of a vent.

The probability $P^k(x_{i,j})$ integrates the various measurements made so far ($P^k(x_{i,j}) = P(x_{i,j} = \text{vent}|T^0,\ldots,T^k)$), and is computed incrementally according to a classical bayesian paradigm under a markovian assumption:

$$P^k(x_{i,j}) = \frac{P(T^k|x_{i,j} = \text{vent})P^{k-1}(x_{i,j})}{P(T^k)} \tag{3.6}$$

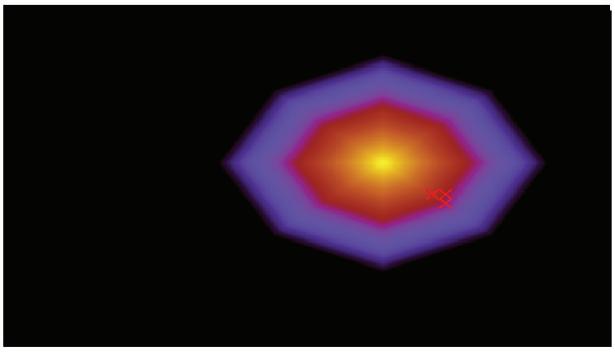


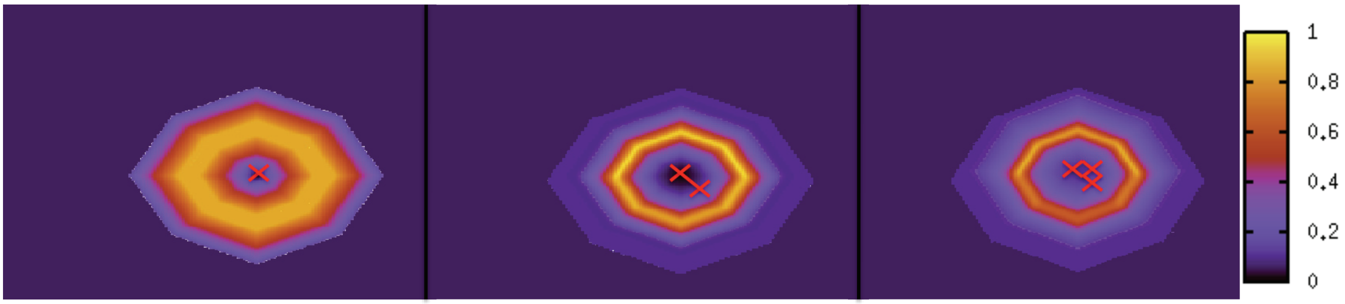
Figure 3.8 – *Real world temperature map of the target and three different location of the vehicle.*

3.5.1 Range-only sensor update

If the sensor is a range-only one then the formulation is equivalent to the update of an obstacle uncertainty grid [Elfes89] using a range-only sensor that returns the distance to the closest obstacle.

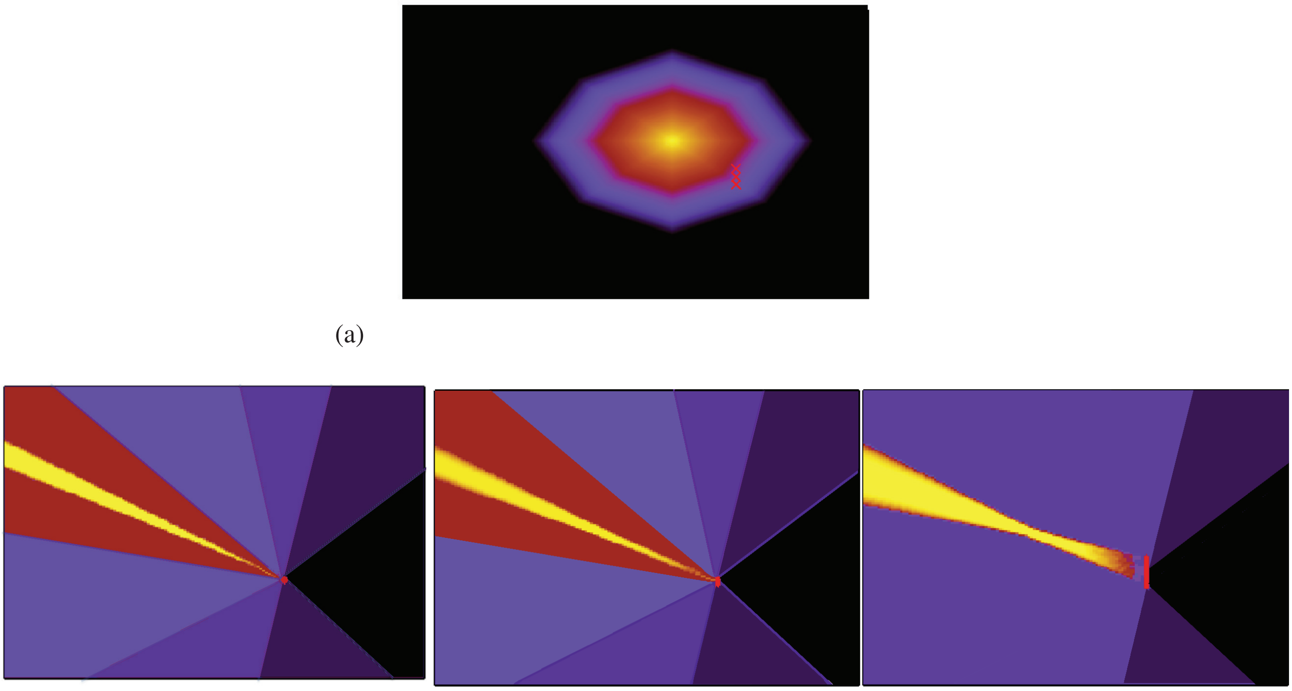
Figure 3.9 illustrates the update of the source map in an environment containing a single source, considering three measures taken at three different positions.

The map implicitly contains two information: the *probability* $P^k(x_{i,j})$ that the cell $x_{i,j}$ is located above a source, the probability value implicitly representing the precision of the source location (a probability equal to 1 meaning that the source is perfectly localized).



(a) (b) (c)

Figure 3.9 – *Evolution of the cell probability to contain a source, derived from 3 measures made at the same depth in the environment shown in figure 3.8 – the positions where the measures are taken are denoted by a cross. (a) is the map using one measure and (b) and (c) are the resulting map using two and three measures in order.*



(a)

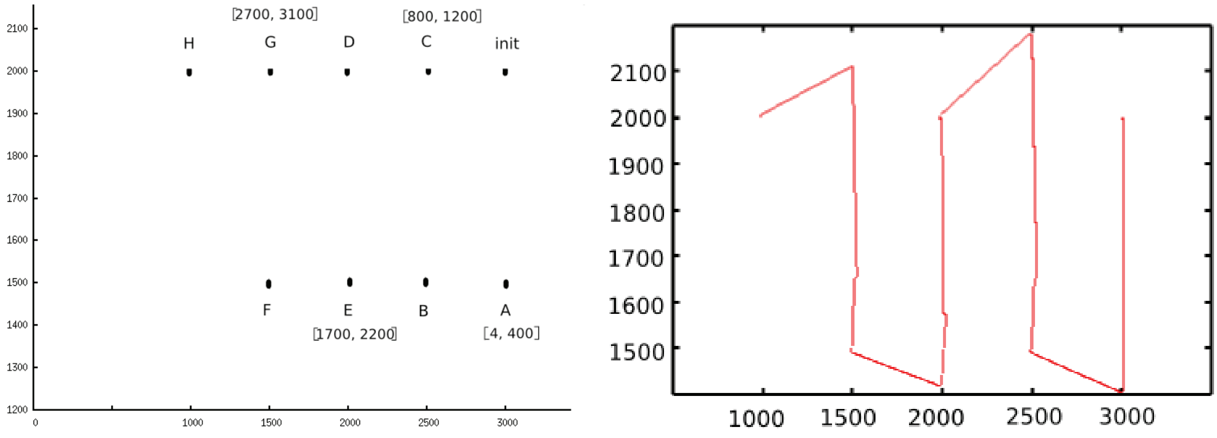
(b) (c) (d)

Figure 3.10 – *(a) Illustration of the initial target model and the fourth different location of the vehicle path. (b), (c) and (d) Illustrate the obtained map using a bearing-only sensor with two, three and fourth measures.*

3.5.2 Bearing-only sensor update

If the vehicle uses a bearing-only sensor, the probability $P^k(x_{i,j} = vent)$ is computed as a distance to the computed direction. The closer the cell $x_{i,j}$ is to the direction axis, the higher the probability is (see figure 3.10b). The direction of the target is computed using the algorithm 1. Figure 3.10 illustrates a couple of measures taken at different location for the same depth. Figure 3.10b is the map using one measure and the figure 3.10c and 3.10d are the resulting map using two and three measures in order.

3.5.3 Illustration of map building with a single robot



(a) (b)

Figure 3.11 – *(a). Illustration of predefined waypoints for one underwater vehicle. (b). Illustration of the obtained path. The time intervals to reach the waypoints (e.g. The vehicle can reach the point A between 4 until 400 ut) are large enough to allow the AUV explores a bit further, hence the resulting saw path.*

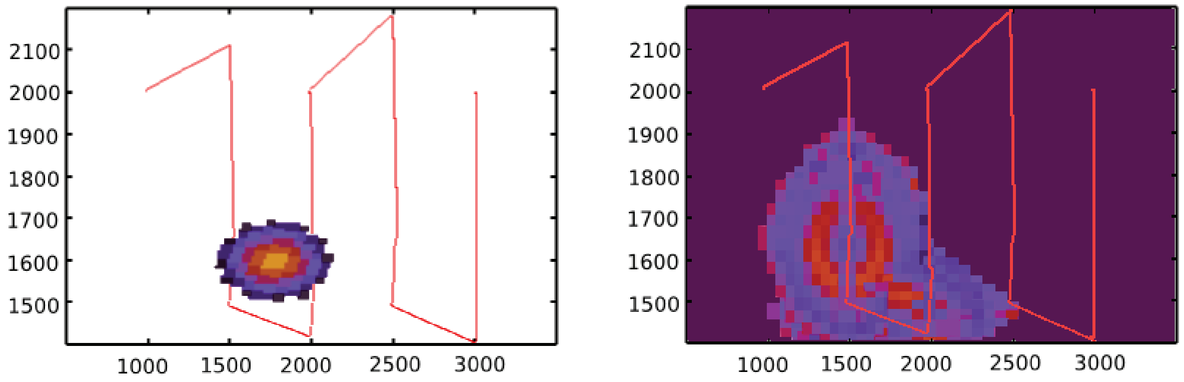
We defined eight points (A, B, C, D, E, F, G and H) that represent a trajectory for the vehicle (figure 3.11). Each point is defined along the three dimensions (x, y, z). In our case, all points are in the same plane and have the same depth (z = 10 meters), except the initial point where the vehicle is at surface (z = 0 meters). There are some waypoints that the vehicle has to visit within a particular time interval. For example for the point A, the vehicle has to be in A at the earliest at 4 *ut* [1] and at the latest at 400 *ut*.

Figure 3.11 shows the executed path of the AUV. All predefined points are achieved on time. Some of waypoints are achieved before the end time of the interval. Rather than staying at the same waypoint (that is impossible for underwater vehicles), we choose to let the vehicle explore the area with limited range, until a new waypoint has to be reached.

The existing target in this area and the obtained map at the end of the execution are represented in figure 3.12. In the next chapter we are going to make the vehicle adapt its predefined exploration strategy according to its measured data. This adaptive exploration strategy has to take into account predefined points and at the same time get more information about the target location.

---

[1] *ut* stands for unit of time.



(a) (b)

Figure 3.12 – *(a). Actual temperatures in an environment containing one target. (b) Illustration of the obtained map using a range-only sensor.*

3.6 MULTI ROBOT MAP BUILDING

In this section we discuss the interest of communication with other vehicles. First, we present the communication model between vehicles. After, we discuss how vehicles can fuse their maps. Then, we explain what is the communication bandwidth and the amount of exchanged data. Finally, we illustrate the map building process with two vehicles.

3.6.1 Communication model

Communications are established between the ASV and the AUVs for the proper monitoring of the mission execution, to exchange data (target maps) between the AUVs and positioning the AUV. Underwater communications are very constrained and quite complex to model [Meyer06]: we choose a conservative model, that states that an AUV and the ASV can communicate if and only if they are located within the same grid cell in the horizontal plane and with a vertical distance of 10 meters. [Akyildiz04] gives all different range and distance between vehicles that can be used to allow the communication. In our case, we choose the very short type of communication, where the “bandwidth” is equal to 100 bps and the communication distance between two vehicles varies from 10 to 100 meters.

At each communication point, the AUV uploads the information gathered since the last communication. Should the maps from various AUVs be fused, this fusion is performed on-board the AUVs, where the ASV acts as centralized information hub. Also, these communication points plays an important role on re-localizing the AUV.

3.6.2 Fusing maps among underwater vehicles

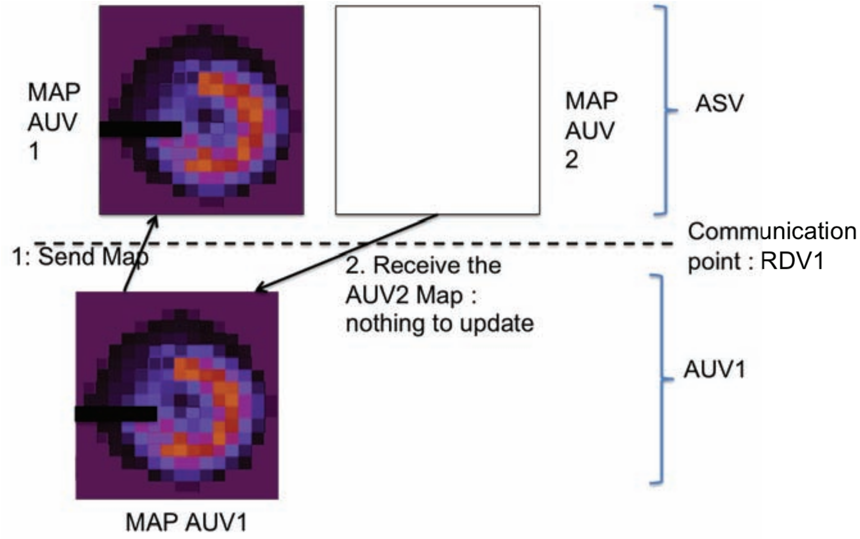


Figure 3.13 – *Evolution of the cell probability to contain a source, derived from a map fusion between the AUV1 and the ASV. At the first communication point with the AUV1, AUV1 sends its new map to the ASV, receives the latest map of the AUV2 and updates its map.*

At each communication point, the AUV sends its new observed probability grid and receives the observed probability grid from the other AUVs by means of the ASV. To avoid data redundancy, the vehicle sends its observed grid from the latest communication point to this new communication point. Figure 3.13 represents the first communication point (RDV1) between the AUV1 and the ASV. The AUV1 sends its data to the ASV. The data here is a couple of the vehicle location and the sensed measure (the data is represented by d_t, where t is the instant). The ASV keeps the latest map of the AUV2 and sends it to the AUV1. At the second communication point (see figure 3.14), the AUV2 is communicating with the ASV. At the top left and right the maps sent from the AUV2 and AUV1 respectively to the ASV. The ASV keeps the latest map of each vehicle. In this example, AUV2 is sending its map to the ASV and then receives AUV1 map which can then be merged. The new map has some relevant target locations that appear in fair color. To update the AUV map, the probabilities are computed incrementally using equation (3.6).

3.6.3 Communication bandwidth and amount of exchanged data

The effectiveness of a cooperative approach depends on the way the sensed data is treated and forwarded. One way to send the strict sufficient minimum data to other vehicles to let the vehicle take more time for exploration.

We want to know what is the communication time duration between two vehicles. Knowing that the size of the sensed data ($|d_0|$) for each cell is equal to $40 bits$ and the bandwidth L is equal to 100 bps. n_d represents the number of explored cells. The communication time (com_t) is

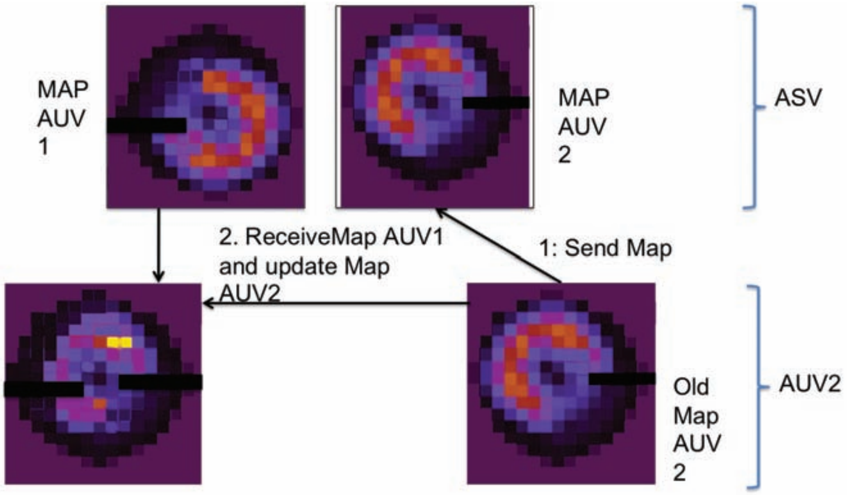


Figure 3.14 – *Evolution of the cell probability to contain a source, derived from a map fusion between the AUV2 and the ASV. At the first communication point with the AUV2, AUV2 sends its new map to the ASV, receives the latest map of the AUV1 and updates its map.*

computed as follow:

$$com_t = \frac{n_d * |d_0|}{L}.$$

Where the data size is calculated as follows:

$$|d_0| = |d_1| = \cdots = |d_i| = | < x_0 ; z_{x_0} > | = |x_0| + |z_{x_0}| = 2 * 16_{bits} + 8_{bits} = 40_{bits}.$$

Replacing the data size by $40bits$ and the used bandwidth by $100bps$, the communication time is equal to $com_t = 0.4 * n_d$. If the communication time is bounded by 2 minutes, it means that the vehicle can explore less than 150 cells (each cell represents 100m^2). In general, because of the localization error of the vehicle, the vehicle can explore at maximum a distance of 1.2 km (12 cells) with a speed of 1.5m/s. This limit indicates that the vehicle will communicate less than 2 minutes, which is a reasonable communication time.

Algorithm complexity To avoid data redundancy, the vehicle sends its observed grid from the latest communication point to this new communication point. The ASV keeps the latest map of each vehicle. To update the ASV map a simple algorithm is used. The complexity of the algorithms for the AUVs and the ASV at each communication point com_i is the following:

- The ASV get the map of one underwater vehicle: the maximal number of cells that can be exchanged is $n_d = 150$ cells. After getting all the new cells information, the ASV updates the global map with at most $N * M * N_{robots}$ operations. Once this map is updated, the ASV sends to each AUV all the maps of other AUVs ($N_{robots} * n_d$ cells). The total operations for the ASV are $N * M * N_{robots} + N_{robots} * n_d$.
- Each AUV sends its new measured data (n_d) and get from the AUV $N_{robots} * n_d$ new cells. The AUV updates its map by using at most $N * M * N_{robots}$ operations.

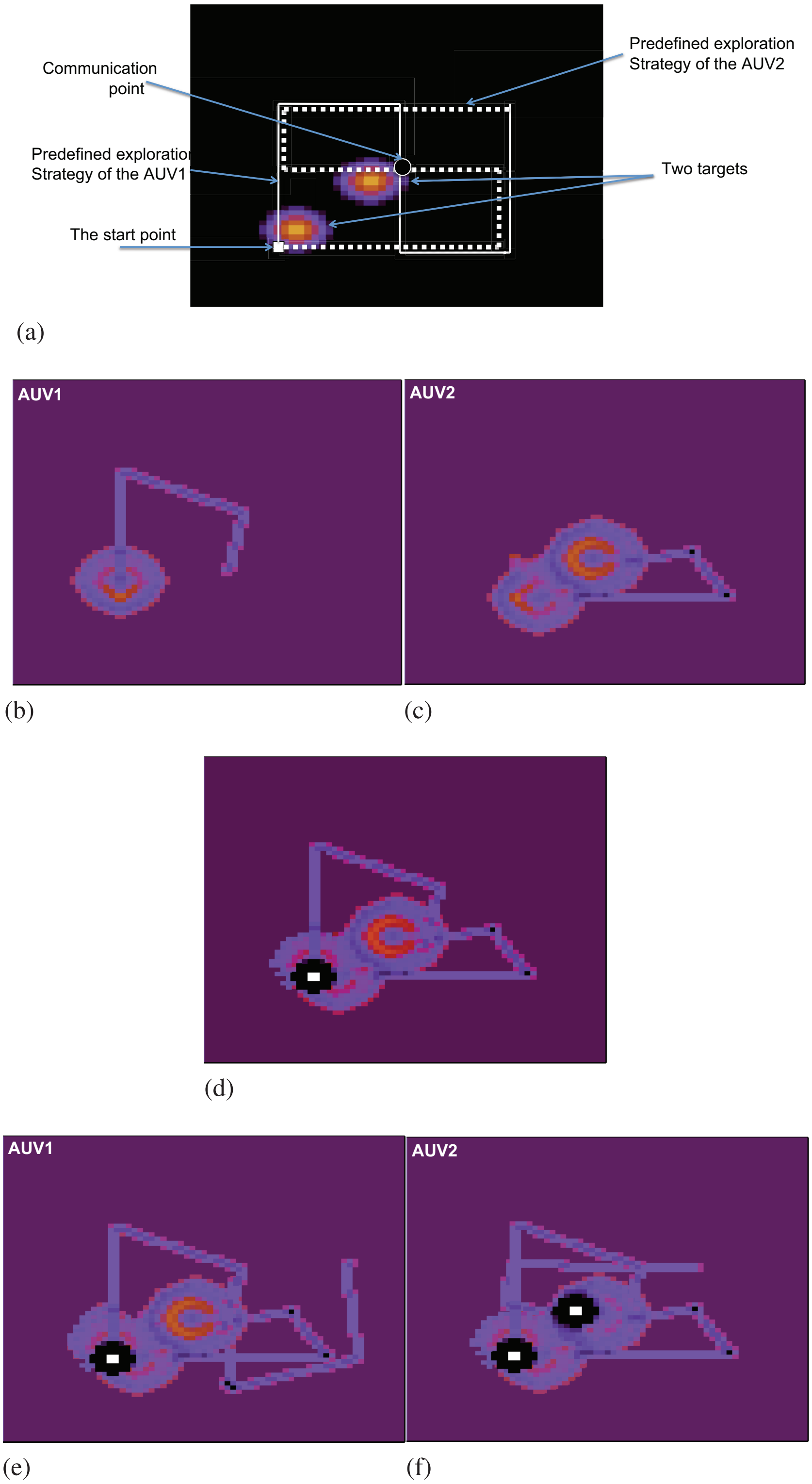

Communication point
Predefined exploration Strategy of the AUV2
Predefined exploration Strategy of the AUV1
Two targets
The start point
(a)
AUV1
AUV2
(b)
(c)
(d)
AUV1
AUV2
(e)
(f)

Figure 3.15 – *(a): Illustration of the real world and the predefined exploration strategy for each AUV. (b), (c): represent the obtained map for each AUV1 and AUV2 before communication. (d): represents the obtained map of each AUV1 and AUV2 after communicating. (e), (f): represent the obtained map of AUV1 and AUV2 at the end of the mission execution.*

3.6.4 Illustration

We designed an example to illustrate how vehicles exchange collected data to localize targets. In this example, we are using two AUVs that explore an area that contains two targets. Each AUV has a predefined exploration strategy and there is a communication point defined between these AUVs as shown in figure 3.15a.

Before they begin communicating, the obtained map of AUV1 and AUV2 are as shown in figure 3.15b and figure 3.15c respectively. They then exchange data to synchronize theirs maps and finish their communication. The resulting map for each AUV after this process is shown in figure 3.15d. The obtained map at the end of the mission of each underwater vehicle is shown in figure 3.15e and figure 3.15f.

CONCLUSION

Modelling real phenomenon is a difficult task, especially when several criterions are to be taken into consideration. We have given a simple model of a marine hotspot. We take into account temperature criteria only and leave other important criterion like marine currents etc., which require rigorous mathematical analysis and are out of scope of our work. Our chosen model yields the definition of range sensors and bearing sensors. The map updating of each under water vehicle is done differently based on the type of sensor used.

The communication mechanism used in underwater vehicles plays an important role for mission execution. We have shown our communication model and the amount of exchanged data between underwater vehicles, and illustrated the cooperative mapping process with two communicating AUVs.

4

ADAPTIVE EXPLORATION STRATEGIES

In this chapter we define an adaptive exploration strategy, and compare it to a non adaptive strategy in term of the number of found targets. Other experiments illustrate the influence of the sensor range and the transect width on the number of found targets.

CARL GUSTAV JUNG, SAYS: «The meeting of two personalities is like the contact of two chemical substances. If there is any reaction, both are transformed.»

4.1 INTRODUCTION

In our context, the cooperation is related to the way the vehicles communicate and the way the sensed data are treated. This chapter defines the used motion model of the vehicle (section 4.2), and defines exploration strategies that adapts the vehicle path according to the built map (section 3.4). To evaluate the cooperative approach, some statistical results are presented in section 4.4.

4.2 MOTION MODEL

The motion model is simply defined on the map grid: an elementary motion shifts the vehicle from one cell to one of the 8 neighboring cells. A time proportional to the distance is associated to the motions [1]. Each motion also yields a drift on the AUV position estimate. This drift is defined as a maximum distance below which the resulting position uncertainty is bounded by the map cell size. This distance is exploited to define the rendez-vous with the ASV, to ensure that the rendez-vous will actually occur, so that the drift is then corrected thanks to the ASV that corrects the AUV drift.

4.3 EXPLORATION STRATEGIES

With an adaptive strategy, the AUVs select the next motion in order to confirm the presence of a target. Given a source presence hypothesis (a local maxima of $P(x_{i,j})$ in the mapped vicinity of the current AUV position), the motion that maximizes the source detection is straightforwardly the one that drives the AUV towards the direction of this maxima. This is true only in the case where we use the target model with known maximal temperature, *i.e.* with a range only sensor. The vehicle follows the maxima to reduce the target location error (see section 3.4.1). In the case where the maximal temperature of the target is unknown, the strategy is the one that maximize the information gain, as presented in section 4.3.3.

Two thresholds can be defined to consider the source as “treated”. A high value denoted P_{loc} leads to a confirmed well localized source, whereas a lower value P_{conf}, leads to the confirmation of a source presence, but not precisely localized. Two types of strategies can be considered with respect to the more global mission: a non adaptive strategy simples leads the AUVs towards their waypoints regardless of the information they gather, and an adaptive strategy drives the AUVs in order to enhance their information, according to what they gathered so far.

4.3.1 Non adaptive exploration strategy

This strategy has predefined waypoints and does not adapt the AUV motion to build the source map. There is a time window to reach each waypoint. The order of these waypoints implicitly defines exploration transects. Even for this simple strategy, one can consider the two thresholds defined above: P_{conf} and P_{loc} to analyze the built map in order to assess the presence or the position of target.

---

[1] Diagonal motions cost $\sqrt{2}$ times the cost of motions along the grid axes.

4.3.2 Adaptive strategies

This strategy has also predefined waypoints with the possibility to redirect the vehicle according to the acquired information. We define two adaptive exploration strategies:

a. Greedy Information Driven (G.I.D)

The vehicle heads toward the path that collects the more information about the source hypothesis until its probability exceeds P_{loc}. This greedy strategy focuses on the nearest target until it is considered localized. Figure 4.2 shows the result of the G.I.D exploration strategy applied in the environment shown in figure 4.1.

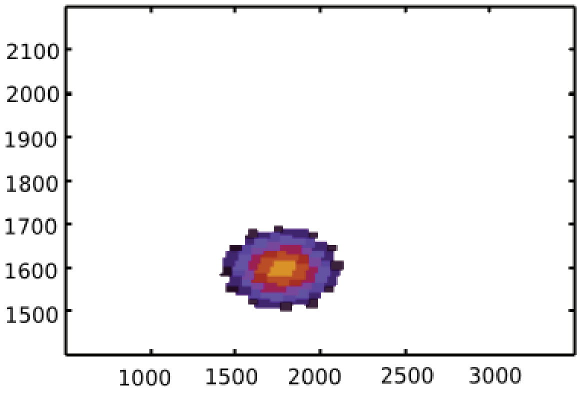
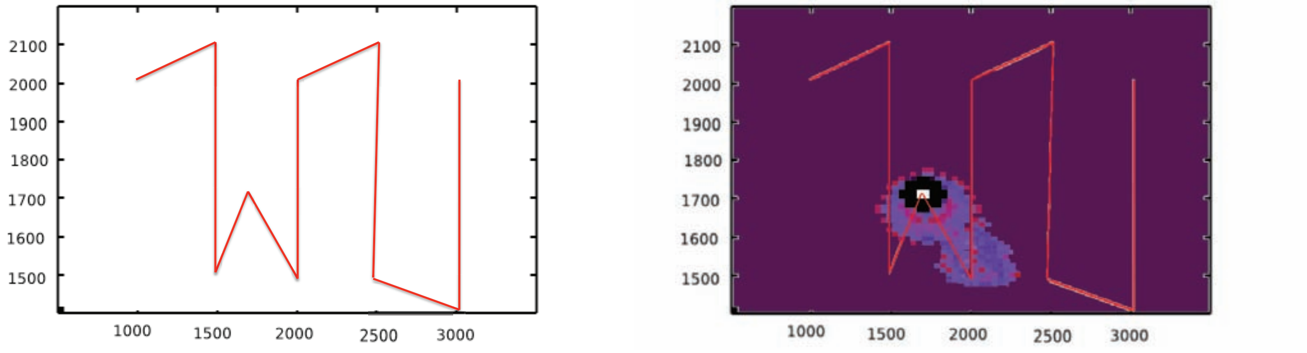


Figure 4.1 – *Illustration of the real world.*

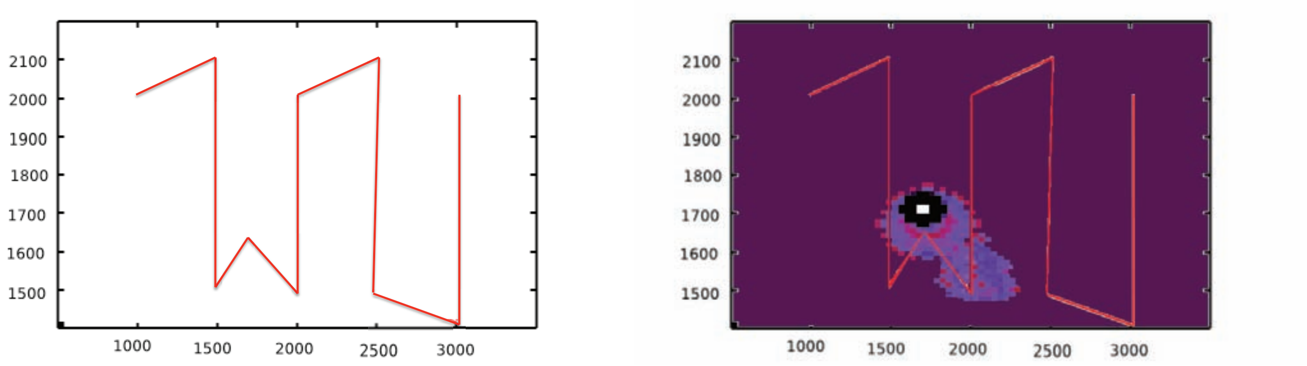


(a) (b)

Figure 4.2 – *(a) Illustration (on the environment with one target shown in figure 4.1) of the obtained map using G.I.D strategy for one underwater vehicle. (b) Illustrates the obtained map at the end of the mission execution.*

b. Global Information Gain driven (G.I.G)

The vehicle heads toward the path that collects the more information about the source hypothesis until its probability exceeds P_{conf}: with respect to the G.I.D strategy, this strategy favors the reduction in the number of vehicle actions, at the cost of less precisely localized sources. Figure 4.3 shows the result of the G.I.G exploration strategy.



(a) (b)

Figure 4.3 – *(a) Illustration of the obtained path using the G.I.G exploration strategy. (b) Illustration of the obtained map at the mission execution end.*

Figure 4.4 shows another example comparing the non-adaptive and the adaptive exploration strategy. The dark circles upper dark/fair lines, represent the communication points with the ASV. The dark line represents the adaptive G.I.G strategy and the white one represents the non-adaptive. In this experiment, using the non-adaptive strategy the number of localized targets is 6 out of 23 (26%). However, the G.I.G strategy localize 14 (60%): unsurprisingly, G.I.G detects more targets.

4.3.3 Adaptive Cooperative Exploration Strategy (ACES): Algorithm

We summarize the general procedure that is used to generate and choose the next cell to explore as follows:

1. *Representation of the environment:* To reduce the amount of collected data, we have represented the environment as a map that discretizes the environment into a collection of ordered cells in a regular pattern $N*M$ (see section 3.5).
2. *Updating the grid:* At each new measure the vehicle has to update its grid. The update is computed incrementally according to the equation (3.6).
3. *Making decisions (the strategies of exploration):* This allows to define adaptive strategies in which the AUVs select the next motions in order to confirm the presence of a target. Given a source presence hypothesis (a local maxima of $P(x_{i,j})$ in the mapped vicinity of the current AUV position), two types of motions are defined depending on the sensor model.

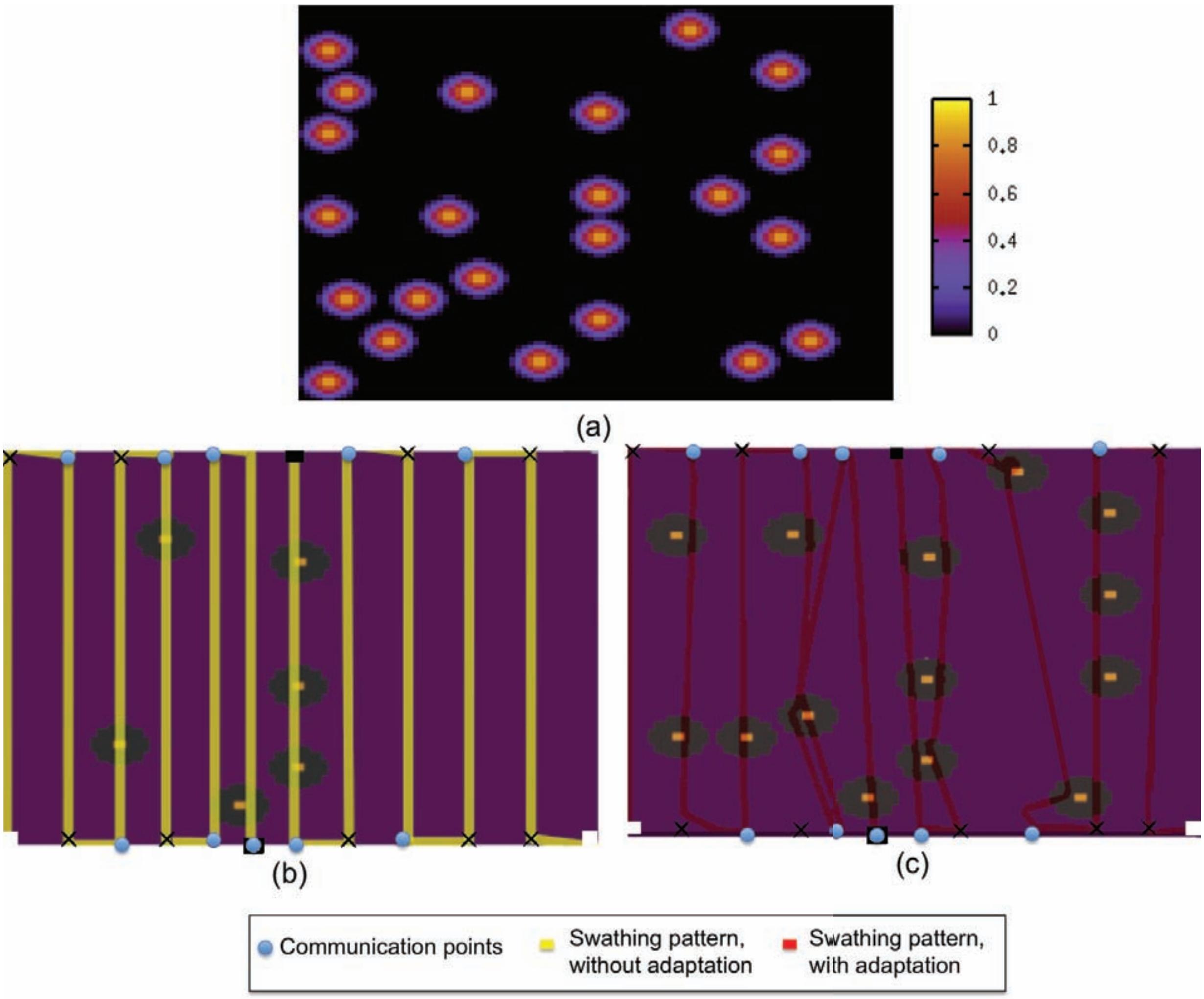

1
0.8
0.6
0.4
0.2
0
(a)
(b)
(c)
Communication points
Swathing pattern, without adaptation
Swathing pattern, with adaptation

Figure 4.4 – *Illustration of the two strategies. Fig. (a) represents the real target positions. Predefined waypoints (black X) and communication points (blue dots) are represented in fig. (b) and (c), the black and white squares respectively represent the starting and ending point of the mission. Fig. (b) shows the vehicle swathing pattern of the vehicle as initially set, without any adaptation, and Fig. (c) shows the trajectory resulting from the adaptive strategy: 6 targets out of 23 were localized in the first case, and 14 targets are localized in the second case.*

Algorithm 2: Algorithm that generates opportunistic goal.

```
Require: $D$: the diameter of the target ;
    $Target_{x,y}$: the coordinate of the target ;
    $P(x_{i,j})$: the probability that $x_{i,j}$ is the target;
    $a$: an action ;
    $\tau$: a transition function. From an action and a cell it can generate the next cell.
Update grid using equation (A.1).
if ($\exists i, j : i \in ]0,N], j \in ]0,M], P(x_{i,j}) \geq P_{conf}$) then
    <Target Found>: Clear around
    for ($\forall i, j : i \in ]0,N], j \in ]0,M]$) do
        if ($0 < |x_{i,j} - Target_{x,y}| < D$) then
            $P(x_{i,j}) \leftarrow 0$;
            $P(Target_{x,y}) \leftarrow 1$;
        end if
    end for
end if
if ($\exists i, j : i \in ]0,N], j \in ]0,M], P(x_{i,j}) \geq P_{loc}$) then
    <Target Found>: Clear around when there is not another target around.
    for ($\forall i, j : i \in ]0,N], j \in ]0,M]$) do
        if (($0 < |x_{i,j} - Target_{x,y}| < D$) and
        ($\forall k, l : k \in ]0,N], l \in ]0,M] : if\ \nexists x_{l,k} | P(x_{l,k}) = 1 \wedge |Target_{x,y} - x_{l,k}| > D$)) then
            $P(x_{i,j}) \leftarrow 0$;
            $P(Target_{x,y}) \leftarrow 1$;
        end if
    end for
end if
<Choose Next Cell>
$a \leftarrow$ using the equation (4.1) or (4.2).
$x'_{i,j} \leftarrow \tau(a, x_{i,j})$
return $x'_{i,j}$
```

(a) Using the range-only sensor, the motion that maximizes the source detection is the one that drives the AUV towards the direction of this maxima. A threshold can be defined to consider the source as "treated". A value P_{conf}, leads to the confirmation of a source presence, but not precisely localized. The algorithm 2 explains for each measured data of one vehicle, what is the next chosen cell $x'_{i,j}$. To choose the next cell the vehicle follows the maximal value of the target. This is why in the algorithm 2 line 21 the next cell is chosen as follows:

$$\begin{aligned}&Max_{a_i\in\{actions\}}P(x_{i,i}/_{x_{i,j}=NotExplored}) \\ &actions \in \{left, right, behind, front, behind_{left}, behind_{right}, front_{left}, front_{right}\}\end{aligned} \tag{4.1}$$

where a_i is the action that allows the vehicle to move at different grid cell. This equation shows that the vehicle is following the direction of the maximal probability value in the map.

(b) Using the bearing-only sensor defines another exploration strategy where to localize a target is not to drive the AUV towards the direction of the maximum but is to go in another direction that can collect more information for the exact target location.

In the algorithm 2 line 21, the vehicle uses this function to choose the next cell to explore:

$$\begin{aligned}&Max_{a_i\in\{actions\}}f(z_{\tau(x_i,a_i)}) \\ &actions \in \{left, right, behind, front, behind_{left}, behind_{right}, front_{left}, front_{right}\}\end{aligned} \tag{4.2}$$

where $f(z_{\tau(x_i,a_i)})$ is a function that predicts the future values in the Map, when the chosen action is a_i. For example, we suppose that the target is on the left side of the vehicle. If the $a_i = left$ that implies that the next values of the cells have the same values as precedent, but if $a_i = front$ the probability for next cells differs. Based on this reasoning the vehicle chooses its next explored cells.

4. *Adaptive Cooperative Exploration Strategy (ACES):* In the case where the vehicles exchange data, the function of exploration (see Algorithm 2 line 21) depends on three criteria:
    (a) Predefined waypoints represent a passage point that the vehicle has to achieve (with priority equal to 1).
    (b) Opportunistic goals are generated according to the vehicle measurement. The point is chosen according to the maximal value of the probability in the grid (with priority equal to 2).
    (c) Predefined communication points are important for the vehicle positioning and also for data exchange. Due to the time delay, the vehicle has to choose between the waypoint, the opportunistic goal and the communication point, where the last one has a priority equal to 3.

The higher the priority, the higher the criterion is taken into account. If the vehicle cannot achieve all the three different criteria at a specific time, it has to achieve at least a maximal number of them with a higher priority.

4.4 STATISTICAL RESULTS USING DIFFERENT COOPERATIVE SCENARIOS

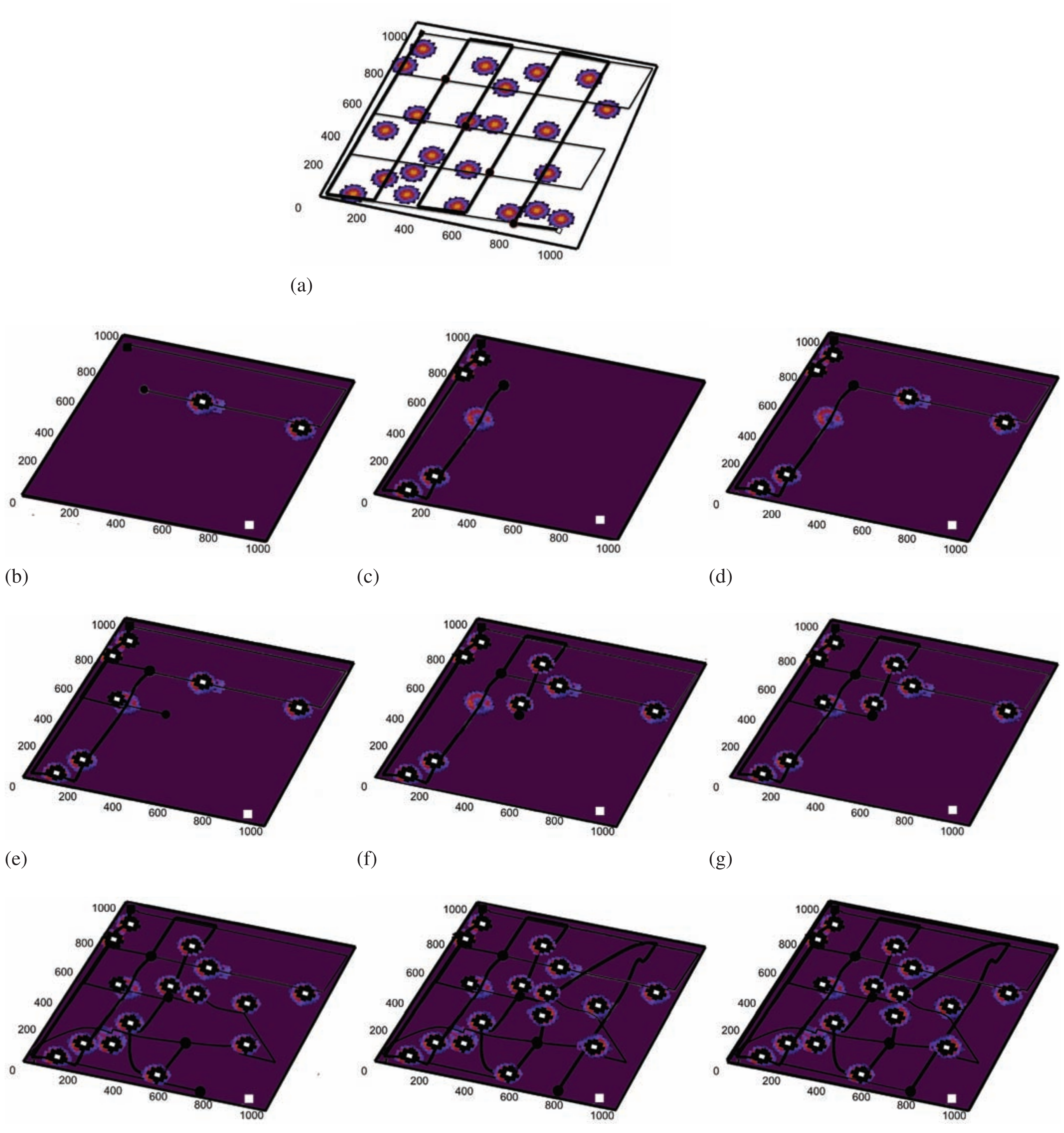
We constructed two groups of cooperative scenarios for statistical assessment. The first group aims at showing that the adaptive exploration strategy is more interesting than non-adaptive strategy. These scenarios consist of using two AUVs and one ASV to localize a maximal number of targets. The second group, on the other hand, aims at showing the importance of the coverage different cooperative scenarios. In this sense, they show the importance of the predefined sensor range and the transect width.

4.4.1 Adaptive vs. Non-adaptive Exploration Strategies

We conducted fifty experiments to show that adaptive exploration strategy locates more targets. We setup two vehicles to explore the same area and sharing their maps to cooperate. Figure 4.5.(a) represents all the targets in the area (23 targets). The black dots on the diagonal represents the communication points where the AUVs share data with the ASVs. These communication points aim to exchange maps and correct the vehicle drift. Due to the restricted energy of each underwater vehicle and the use of energy when communicating, these rendezvous have to be reduced. This is why the vehicle can explore areas without communication until some maximal distance is reached. This maximal distance is defined by the motion model, and is set so that the position uncertainty remain below the cell size. The idea is to find a minimal number of these rendezvous to extend mission exploration. It is a complex problem, and we proposed a simple way to generate communication points for two orthogonal swathing patterns. For that, we define a matrix, that has the same dimensions as the exploration map, for each vehicle. This matrix $A^k(i,j)$ represents the time that the vehicle "*k*" will be at the position (i,j). The communication point between two (or more) vehicles is defined when two (or more) vehicles are at the same place at the same time (*i.e.* if $(A^l(i,j) = A^m(i,j))$ then the vehicle "*l*" will meet the vehicle "*m*" at the place (i,j)). After defining all communication points between vehicles (*i.e.* $Com_{x,y}$), the second objective is to reduce the number of rendezvous. Taking into account that each vehicle cannot exceed a maximal distance of exploration without communicating, some communication points cannot be removed. The main idea of the resolution is that a communication point is removed, when it is not needed by the other vehicle.

Example : Consider the following two matrices of two AUVs.

$$A^1(x,y) = \begin{pmatrix} 0 & 1 & 2 \\ 5 & 4 & 3 \\ 6 & 7 & 8 \end{pmatrix}, A^2(x,y) = \begin{pmatrix} 0 & 5 & 6 \\ 1 & 4 & 7 \\ 2 & 3 & 8 \end{pmatrix}, Com(x,y) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



(a)

(b) (c) (d)

(e) (f) (g)

(h) (i) (j)

Figure 4.5 – *Illustration of two vehicles sharing data for the same area. For the other figures: the left column represents the target detected by the first UAV, the middle column the target detected by the second UAV, and right column the targets detected when merging both maps. Times goes downwards, each line corresponds to a communication point.*

We suppose that the maximal exploration distance of the vehicle “1” is 8 moves and for the vehicle “2” is 5 moves. The intersection matrix $Com_{x,y} = \ 1\ if\ A^1(x,y) = A^2(x,y)$, otherwise $Com_{x,y} = 0$. The proposed algorithm explores the $Com_{x,y}$ using the exploration strategy of the AUV1, and removes the additional communication points, but does not remove the necessary communication points like $Com_{2,2}$ that is necessary for the AUV2. So the matrix of communication points becomes $Com(x,y) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$.

Figure 4.5.(b) represents the adaptive exploration strategy of the first AUV. Using the adaptive exploration strategy, 10 targets are found by the first AUV. Figure 4.5.(c) represents the adaptive exploration strategy for the second AUV. The number of found targets for the second AUV is 9. In this experiment, if we combine found targets by both vehicles, the percentage of found targets raises up to 82% at the end of the mission. The adaptive exploration strategy (G.I.G) works also with cooperative exploration approaches and improve the results of the nonadaptive exploration strategy (in this case, it is 54%).

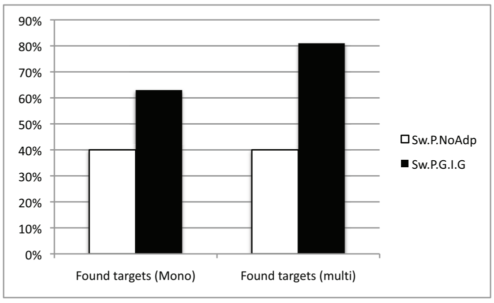


Figure 4.6 – *Average percentage of found targets for the adaptive and non-adaptive strategies, in the single UAV and two-UAVs cases.*

We have performed the experiments with different initial target distributions and computed the average coverage for each exploration strategy. The average results are gathered in figure 4.6: for the number of found targets criterion, the best strategy is the G.I.G, using multiple vehicles (81%). The overall mission execution time for each mono robot and multi robots in our experiments is naturally reduced by a factor of two when using two AUVs.

In general, for mission execution, each criterion (the number of found targets, the precision and the exploration time) is weighted to favor one strategy or another. In the case where all the criteria have the same weight, the best strategy is the Adaptive G.I.G for multiple vehicles.

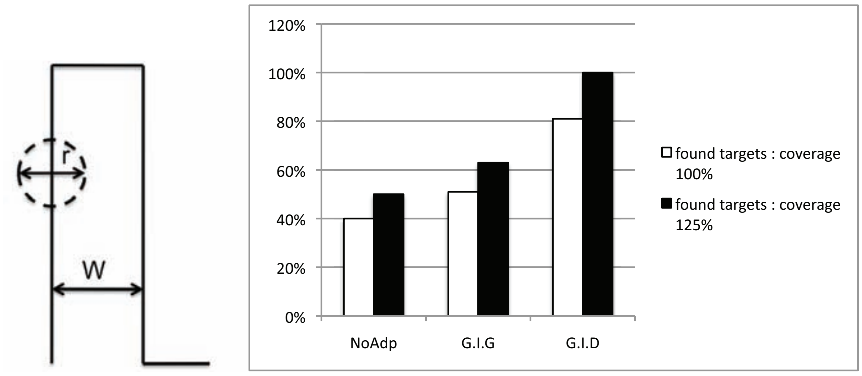
4.4.2 Importance of the Coverage

Each AUV has a sensor range r and an transect width W (see figure 4.7a), which define the coverage r/W. This coverage value naturally impacts the obtained results.

We conducted fifteen experiments to assess the influence of this coverage rate for the non-adaptive and adaptive G.I.G strategies. The strategies are compared for given values of r and W with respect to the execution time, the precision of the target location and the number of confirmed targets.

Figure 4.7b represents two coverage percentages. The white bars represent the found targets using 100% coverage[2], and the black bars represent found targets using 125% coverage.

The figure shows that the more effective strategy is the G.I.D for a coverage of 125%. This percentage means that there is a redundant information at the vehicle sensor. This redundancy is useful to localize targets. For each strategy, we have to find the best percentage to avoid unused redundancy.



(a) (b)

Figure 4.7 – *figure 4.7a : Definition of the vehicle sensor range r and the transect width W. figure 4.7b: Illustration of the various strategies percentage of found targets for two different values of coverage percentage.*

CONCLUSION

In this chapter we presented two types of exploration strategies. The evaluation of the two exploration strategies is based on the number of found targets, where cooperative adaptive exploration strategy naturally imporves the number of found targets. To improve our results we wanted to see what is the effect of the sensor range and the transect width in the percentage of found targets, where these two help the vehicle to improve the number of found targets.

---

[2] which was the case of the results shown figure 4.6.

5

PROPOSED ARCHITECTURE AND DEPLOYED SYSTEM

In this chapter we discuss the requirements of our system and explain the used architecture. It relies on the `T-ReX` architecture, to which we add two components to allow the system to achieve these requirements. All the details about our cooperative system are presented in this chapter.

DAVID HUME, SAYS: «Nothing appears more surprising to those who consider human affairs with a philosophical eye, than the ease with which the many are governed by the few.»

5.1 INTRODUCTION

The SPA (Sense-Plan-Act) paradigm uses an embedded planner and an execution control loop. The planner is the most costly, in terms of computation time, and can limit the reactivity of the system. Nevertheless, the planner reactivity is important when the world is highly dynamic. When the planner deliberates for the whole mission time, it can affect the reactivity of the system. In such cases, distributing the tasks to several subcomponents can be used as a method to reduce the planner workload. This suggests that each subcomponent has its own control loop and its own world perception. Due to that, the need is to synchronize each subcomponent perception with the other.

The `T-ReX` architecture is inspired by the SPA paradigm with the possibility to run at different rates of reactivity. Each subcomponent is associated to a specific objective. We give more details about `T-ReX` architecture in section 5.3, after presenting the requirements of a control architecture and existing work on control architectures in section 5.2. In section 5.4, we present the two components we added in this architecture. One allows to generate opportunistic goals during mission execution, the second component allows the cooperation between vehicles.

5.2 ROBOT CONTROL ARCHITECTURES

A control architecture can be defined as an organization of components for controlled dynamic systems.

5.2.1 Control architecture requirements

Several requirements aim to make autonomous robots accomplish their missions. We detail some requirements for each robot to control the execution of a given mission [Ingrand07]:

Reactivity: The robot has to take into account events with time bounds compatible with efficient achievement of its goals and deal with the dynamics of the environment.

Robustness: The robot system should be able to exploit the redundancy of the processing functions.

Versatility: At any time, the robot system can work with one or more robots, the reactivity of the robot being not dependable on the number of robots.

Autonomy and adaptability: The robot system should be able to modify actions and refine the tasks according to current goal and execution context. New goals can be generated based on on-line measurements, that the robot takes into account to adapt its initial plan.

Goal driven execution: The vehicle tries to accomplish mission objectives given available time, resources and system constraints. All these mission objectives produce a plan that the vehicle will execute.

Flexibility: During the plan execution, some failures can occur and the robot in most of the cases aborts the mission. The objective is to maintain the flexibility of a plan for a given mission.

Passive/Active/Automatic fault detection: When a fault detection and identification does not require perturbation of the system, the fault detection is passive. The active fault detection allows the possibility to issue actions to perturb the system in a manner that will help discrimination between possible fault states. The automatic fault recovery is able to modify the mission plan for the vehicle and safely continue with a modified mission, which is feasible under a new vehicle configuration.

In the next subsection, we discuss different existing control architectures.

5.2.2 Different control architectures

The control architecture for underwater vehicles should be able to perform most of the precedent requirements. Valavanis et al. [Valavanis97] classified different existent control architectures for underwater vehicles. Our classification is inspired by their work.

a. Subsumption architecture

This architecture is a set of behaviors that are working together. The behavior-based control architecture is organized along predefined behaviors [Dudenhoeffer00], that interact with the world. The behavior-based architecture does not create a central world model. This architecture was introduced by Rodney Brooks [Brooks91]: it was one of the first attempts to describe a mechanism to develop behavior-based systems. A subsumption architecture is a decomposition from behaviors into smaller behaviors used to execute one task (e.g. follow a straight line). Each behavior has a hierarchical bottom-up design, where each higher behavior in the hierarchy, takes into account the decision of the lower layer behavior. An example of a behavior-based architecture was deployed in the ADVOCATE II project (ADVanced On-Board Diagnosis and Control of Autonomous Systems II [Sotelo03]).

ADVOCATE II is an on-board diagnosis and control system for an underwater or ground vehicle. The system has several modules where each of them is responsible to execute one task (e.g. diagnosis module, robot piloting module). ADVOCATE II has a system of diagnostic that results from conditional rules.

Similarly, DAMN (a Distributed Architecture for Mobile Navigation [Rosenblatt95]) is a distributed architecture, where several modules concurrently share the control over the robot by sending votes. These votes are weighed and the winning vote is executed by a central *DAMN Arbiter*. This architecture divides the behaviors into five focussed behaviors, represented in

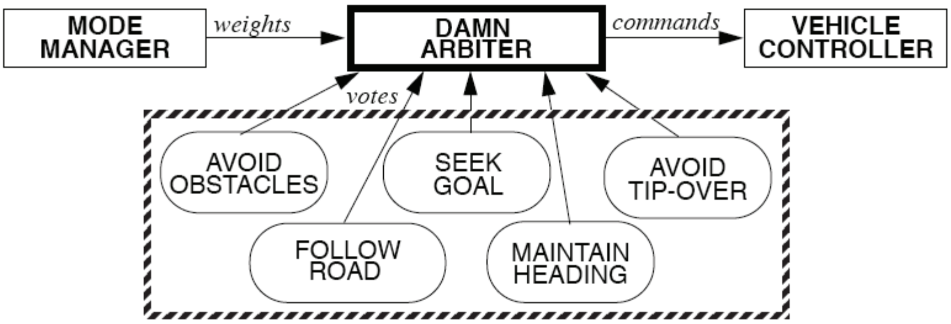

MODE MANAGER
weights
DAMN ARBITER
commands
VEHICLE CONTROLLER
votes
AVOID OBSTACLES
SEEK GOAL
AVOID TIP-OVER
FOLLOW ROAD
MAINTAIN HEADING

Figure 5.1 – *Illustration of the DAMN architecture [Rosenblatt95]*

figure 5.1. Each behavior requires only a part of the world knowledge and receives the sensory data which is directly relevant to its decision needs.

This architecture is interesting for centralized decision making, but at the same time if the *Arbiter* fails, the decision will never occur at all (this is the case for all centralized architectures).

The Orca architecture [Turner95] uses procedural schemas also called p-schemas, to manage different goals. A p-schema is a predefined schema [1] that helps the planner to find the associated schema and then send the associated commands. Figure 5.2 shows different components that manage the vehicle. Given an input from a user or other vehicles, the *Event Handler* handles this input and sends it to the *Agenda Manager* list. The *Schema Applier* takes one event at a time and applies a p-schema using the *Schema Memory*.

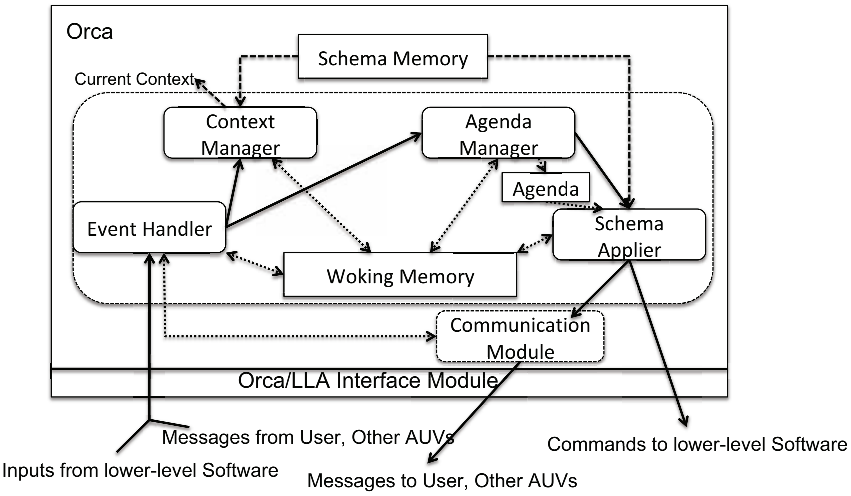

Orca
Schema Memory
Current Context
Context Manager
Agenda Manager
Agenda
Event Handler
Schema Applier
Woking Memory
Communication Module
Orca/LLA Interface Module
Messages from User, Other AUVs
Commands to lower-level Software
Inputs from lower-level Software
Messages to User, Other AUVs

Figure 5.2 – *Illustration of the Orca architecture [Turner95]*

After finding an executable action, the *Schema Applier* sends the action to the *communication module*. This architecture appears to have been used for simulation only. It is not clear how the approach is scalable with respect to the number of schemas. Moreover, it does not reason explicitly about time and resources. An evolution of Orca has been done in 2008 by Albert et al. [Albert08], where a p-schema planner was included. This planner is reactive and it is possible to plan using temporal tasks. IDEA [Muscettola02] is the first architecture to use a time-based representation, with an unified declaration model and an hybrid executive that integrates planning and execution in one framework. This architecture has been used for different robotic missions.

---

[1] It is a set of predefined programs, that plays the role of reactive actions.

For underwater vehicles, we can discuss the architecture used for the Eric AUV [Yuh96] and the Odyssey II AUV [Bellingham94]. Eric AUV is used at the Key Center Robotics Laboratory at the University of Technology at Sydney. It is a three-layered architecture (see figure 5.3): the preservation layer, the exploration layer and the socialization layer. The preservation layer has as objective to avoid obstacles, the exploration layer perform high level navigation and the socialization layer makes object following behavior.

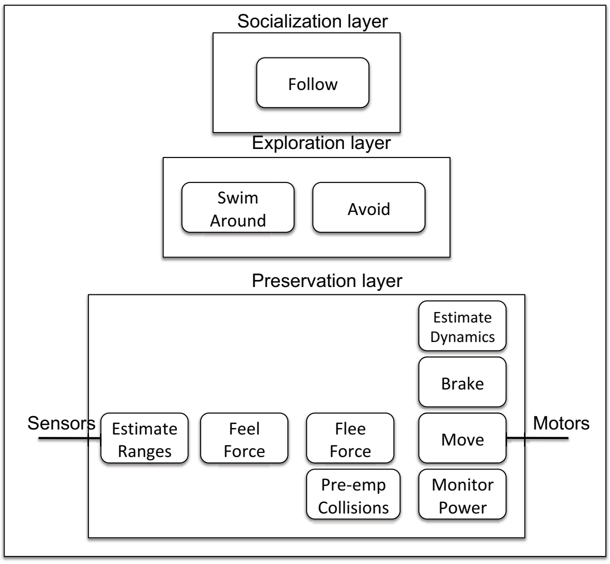


Figure 5.3 – *Illustration of the Eric AUV control architecture*

Behavior-based systems facilitate highly reactive execution but lack a systematic framework for coordination across behaviors. As a consequence, it is difficult to synchronize behaviours and the scalability of the system depends on the number of behaviours. Also, these architectures do not take into account durative actions, which is crucial when one deal with limited mission execution time or cooperative approaches (*e.g.* to communicate, both underwater vehicles need to be in a specific location at a fixed time).

b. Hierarchical architecture

Hierarchical architectures divide the system into several layers, where the highest layer is responsible of the whole mission execution. The lower the layer is, the more specific is the role attributed to the layer. The structure is hierarchical, where the communication between two levels can occur only with adjacent layers. The higher layer sends commands to the lower layer and receives observations from the lower layer.

As example of underwater control architectures, we can refer the ABE architecture [Yoerger94] (autonomous Benthic Explorer). This architecture has two layers, where each layer has different embedded energy and different computational capabilities. The EAVE [Blidberg90], MARIUS [Pascoal95], and OTTER [Rock95] are also hierarchical architectures used for underwater vehicles.

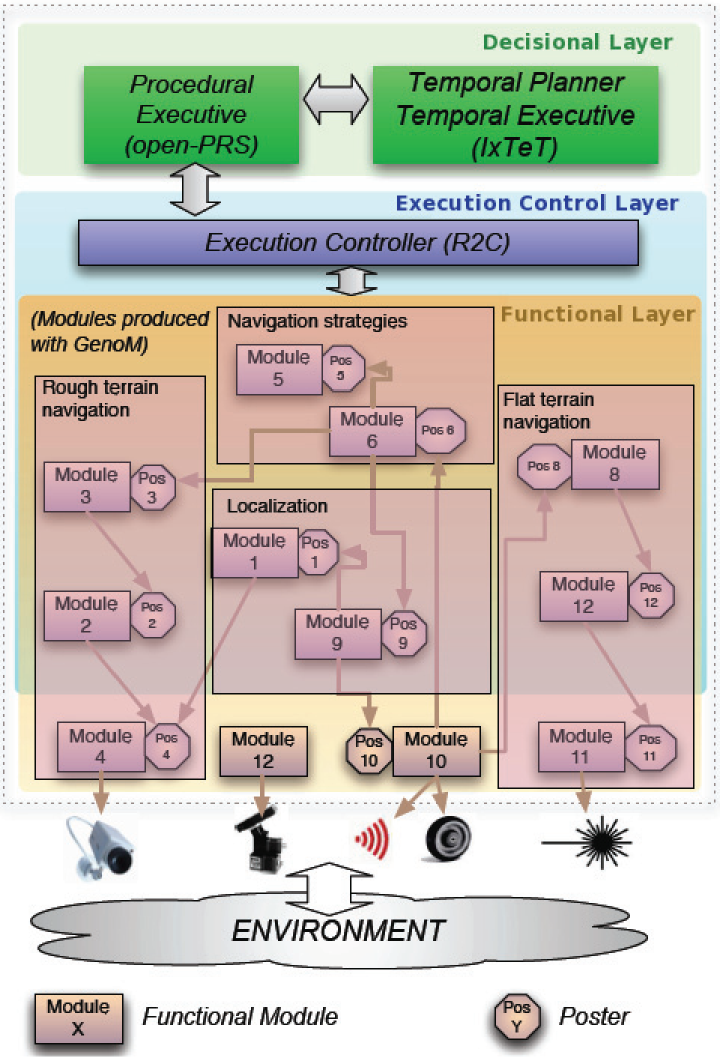

Decisional Layer
Procedural Executive (open-PRS)
Temporal Planner Temporal Executive (IxTeT)
Execution Control Layer
Execution Controller (R2C)
(Modules produced with GenoM)
Navigation strategies
Functional Layer
Module 5
Pos 5
Module 6
Pos 6
Rough terrain navigation
Flat terrain navigation
Pos 8
Module 8
Module 3
Pos 3
Localization
Module 1
Pos 1
Module 12
Pos 12
Module 2
Pos 2
Module 9
Pos 9
Module 4
Pos 4
Module 12
Pos 10
Module 10
Module 11
Pos 11
ENVIRONMENT
Module X
Functional Module
Pos Y
Poster

Figure 5.4 – *Illustration of the LAAS architecture [Ingrand07].*

The LAAS architecture [Alami98] is a three-levels architecture which consists of functional, decisional and executive layers (figure 5.4):

- The Functional layer: This layer contains the perception and action functional components of the robot. Control loops and data interpretation are encapsulated into GenoM [Fleury97] modules. The modules have direct access to the robot hardware. These modules are activated by requests sent by the decisional layer, according to the task to be executed. They send reports upon completion and may export data in posters to be used by other modules.

This layer provides a level of modularity and generality that eases modules integration.

- Decisional layer: This layer provides decision capabilities to the robot. A task planner IxTeT [Gallien06] and a supervisor (OpenPRS [Ingrand96]) are located in this layer. IxTeT is a temporal constraint-based causal link planner. It uses CSP techniques (Constraint Satisfaction Problem) to maintain the consistency of the plan constraints. In particular, the planner uses a Simple Temporal Network [Dechter91] for the temporal constraints, where the plan is a series of state variables over finite symbolic and numerical domains.

- The execution control: It is the interface between the decisional and the functional layer. It controls and coordinates the execution of the functions distributed over the various functional layer modules according to the task requirements. It achieves this by filtering the requests according to the current state of the system and a formal model of allowed and forbidden states.

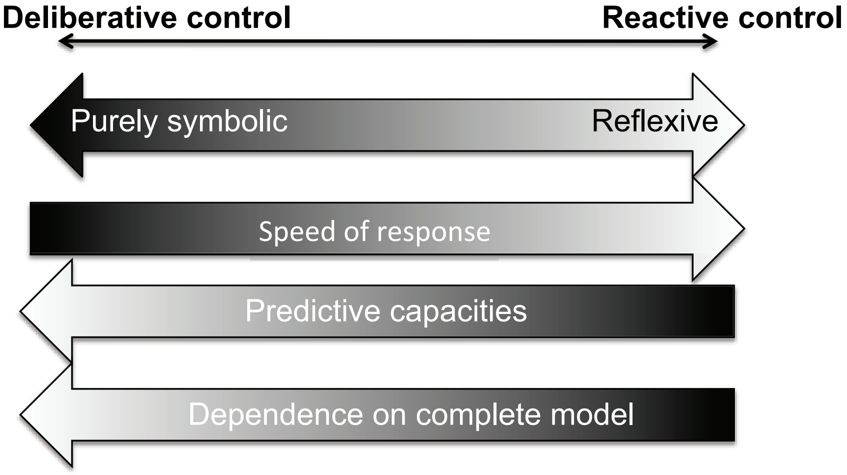


Figure 5.5 – *Illustration of the differences between reactive and delibrative control architectures.*

5.2.3 Summary

In this section we briefly presented different existing control architectures. The reactive or the deliberative architectures are the most used in the literature. Each architecture has different characteristics with respect to the speed of response, model dependence, and predictability (see figure 5.5).

Both reactivity and deliberation are essential to design a control architecture. To deal with these two notions, we have to find a threshold that can make both work together in harmony.

The `T-ReX` architecture (Teleo-Reactive EXecutive) [McGann07] is a goal-oriented architecture, with embedded automated planning and adaptive execution. This architecture provides to each vehicle the capability to generate and execute its plan onboard. Rather than being exclusively deliberative or exclusively reactive, this architecture incorporates planners that can deal with different planning horizons and deliberation times. These different times make `T-ReX` both reactive and deliberative. In the next section, we present the `T-ReX` architecture in more details.

5.3 THE T-REX ARCHITECTURE

To endow each vehicle with some autonomy, we propose to use an architecture for planning and execution control called `T-REX` [McGann07], originally developed at MBARI (Monterey Bay Aquarium Research Institute). This Architecture is inspired by the IDEA [Muscettola02] architecture. The `T-ReX` architecture provides to each vehicle the capability to plan and execute its plan onboard. Before explaining the different components deployed for the `T-ReX` architecture, we explain the used planning system.

5.3.1 Planning system in the T-ReX Architecture

The EUROPA [Frank03] planner is the main component of the `T-ReX` architecture. This planner supposes that the world is fully observable. The world is described by a set of state variables/tokens/activities, that are functions of time. During planning, the evolution of the tokens is specified in a timeline.

a. Tokens and timelines

The basic data structures used in `T-ReX` are timelines that represent the evolution of state variables over time. These timelines have an ordered set of activities called tokens that are mutually exclusive. Each state variable or token can take only one value at a time.

For example :

$$holds(Path, 10, 20, Going(WaterPlace1, WaterPlace2)) \quad (1)$$

Path represents a timeline and *Going(WaterPlace1, WaterPlace2)* from 10 to 20 is a token. It means that the robot will move from the *WaterPlace1* location to the *WaterPlace2* location between 10 *ut* [2] to 20 *ut*. Temporal constraints can be associated to tokens.

b. Configuration rules

These rules define a set of constraints of precedence or exclusion between tokens. These constraints can be *compatibilities* (temporal constraints taken among the thirteen temporal relations of [Allen83]) or *guards* (conditional *compatibilities* similar to the conditional statement in traditional programming language).

---

[2] unit of time

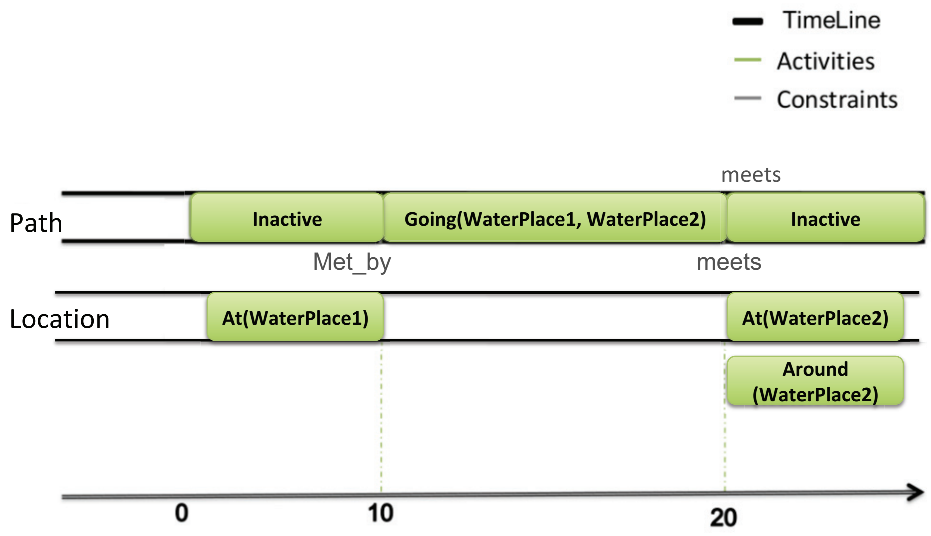


Figure 5.6 – *An example that illustrates the use of timelines, activities and constraints in* `T-ReX`.

A *compatibility* represents all Allen's temporal relationships: *meets, contains, met_by, starts, equals, parallels, start_before_end, starts_during, starts_before, starts_after, ends, contains_start, contains_end.*
A disjunction of compatibilities can be defined, where a variable regulates the choice between disjunctions. This variable is called a *guard*.

Figure 5.6 represents the implemented timeline for the precedent example. To perform a *Going* from the *WaterPlace1* location to the *WaterPlace2* location, the robot has to be *At* the *WaterPlace1* place before doing the *Going*. So, the associated *compatibility* between *At(WaterPlace1)* and *Going(WaterPlace1, WaterPlace2)* is *Met_by*.
The activity *Going(WaterPlace1, WaterPlace2)* starts. At the end of the *Going* token execution, the vehicle should be *At* the *WaterPlace2* or *Around WaterPlace2*. This disjunction of constraints can be written as follow:

$$case\ \gamma = 0 : At(WaterPlace2)\ meets\ Going(WaterPlace1, WaterPlace2)$$

$$case\ \gamma = 1 : Around(WaterPlace2)\ meets\ Going(WaterPlace1, WaterPlace2)$$

γ is called a *guard*. All these components (timelines, activities and constraints) are implemented in NDDL (New Domain Description Language [Bernardini08]) developed at NASA Ames.

We want to add more constraints in the precedent example, where the vehicle can only move in the case where its gulper[3] is off, represented by: $holds(Instrument, t'_b, t'_e, Off())$.
Generally, the rule will state that for any interval of the form

$$holds(Path, t_b, t_e, Going(A, B))$$

---

[3] A gulper is used to take water sample.

there exists another interval $holds(Instrument, t'_b, t'_e, Off())$. The associated constraint is written as $Off()$ *contains* $Going(A,B)$.

This constraint is represented in the temporal constraint by $t'_b \leq t_b$ and $t_e \leq t'_e$ (see figure 5.7).

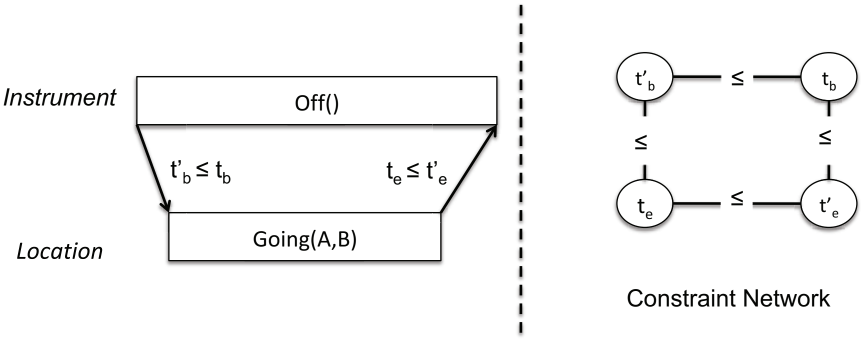


Figure 5.7 – *Illustration of a compatibility using EUROPA and the associated constraint network.*

5.3.2 Planning problem and plan resolution in the `T-ReX` architecture

The planner is given a *planning domain* (*D*) and a *planning problem* (*P*). The *planning domain* consists in the declaration of tokens, timelines and configuration rules. The *planning problem* represents the initial world state tokens to achieve and variables declaration (e.g. in the previous example *WaterPlace1* is a variable).

a. Planning instance

A *planning instance* (*I*) is represented as a pair of *planning domain* (*D*) and *planning problem* (*P*) where $I = <D,P>$.

b. A valid plan

We define a valid plan as a solution if all timelines are filled with tokens without any gap[4] (see figure 5.8) and all the associated configuration rules are satisfied[5].

c. Plan resolution

Given a *planning instance*, EUROPA starts the resolution from the *planning problem* and incrementally refines the plan by adding and ordering tokens on timelines until finding a valid plan. This resolution is a search in the space of partial plans [McAllister91]. Using the *planning instance* EUROPA finds a final plan following these steps (see figure 5.9):

---

[4] In IxTeT a causal link between state variables is imposed

[5] The configuration rule is satisfied if all the constraints are in the resulting plan.

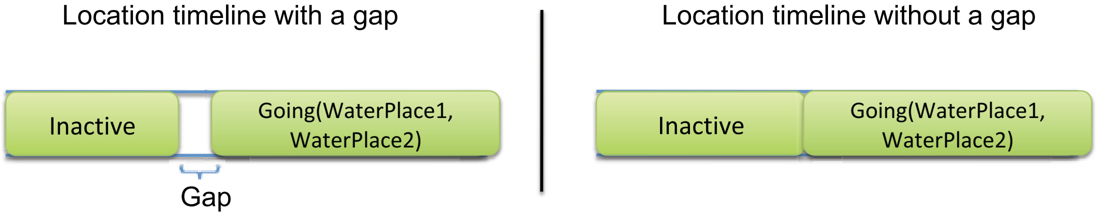

Location timeline with a gap
Inactive
Going(WaterPlace1, WaterPlace2)
Gap
Location timeline without a gap
Inactive
Going(WaterPlace1, WaterPlace2)

Figure 5.8 – *Illustration of location timeline with and without a gap.*

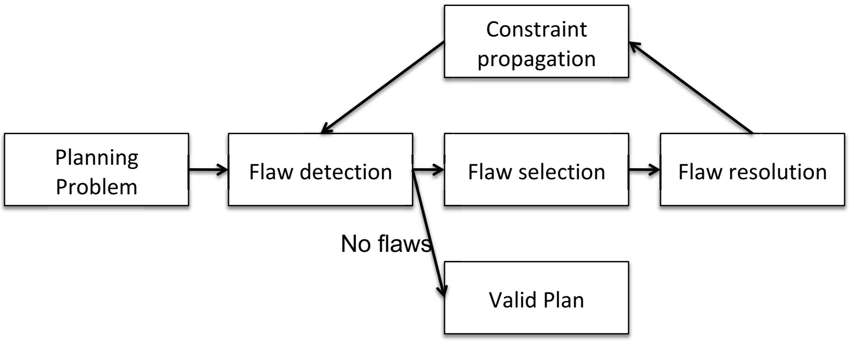

Constraint propagation
Planning Problem
Flaw detection
Flaw selection
Flaw resolution
No flaws
Valid Plan

Figure 5.9 – *Illustration of a plan construction using the EUROPA planner*

1. **Planning problem:** We define a candidate plan as an ordered or unordered tokens intervals. A plan is valid when all the constraints are satisfied. The plan is represented as a constraint network, each token interval is generalized by variables (e.g. $Going(WaterPlace1, WaterPlace2)$ is generalized by $Going(Variable1, Variable2)$), which generate a set of constraints on the temporal variables. So, at each new token in the timeline, several constraints are added to the constrained network (see figure 5.7). To find a valid plan according to a candidate plan, the planner has to: detect the flaws, choose the flaw to resolve and apply the flaw resolver to the plan resolution (see figure 5.9).

2. **Flaw detection:** EUROPA defines three types of flaws:
   - A threat or object flaw: This flaw occurs when an order is required for tokens in an associated timeline. This occurs when two mutually exclusive tokens are in the same plan. For example, if this two tokens *Going* and *InactiveGoing* are in the plan, that means they have to be ordered in an exclusive way (i.e. the execution interval of the *Going* is excluded of the executed before the execution interval of the *InactiveGoing*).

   - An open condition or token flaw: When a token is required to satisfy another token but not ordered yet in the plan, there is an open condition. Token flaws are resolved by inserting the new token in the plan.

   - An unbound variable or variable flaw: This occurs if there are variables of tokens already in the plan with unspecified values. The resolution of variable flaws are addressed by specifying values from variable's domain.

These three flaws that can occur in the partial plan should be selected and then resolved by the planner to complete the plan. When there are no flaws left (the plan database is not inconsistent), the plan is considered valid and the search is complete.

3. **Flaw selection:** It establishes the order in which flaws are treated for resolution. An algorithm is defined to make flaw selection, where this selection will depend on the type of the flaw. This algorithm gives priorities on the timelines, and chooses to resolve the flaw that is on the priority timeline. It can also choose the flaw according to its execution, for example the earliest first. In our case, the priority is given for the *PathController* (discussed in section 5.4.1).

4. **Flaw resolution:** when the flaw is selected, the planner executes appropriate procedures to solve a flaw. These procedures include token activation, token merging, token ordering, variable binding. For the IxTeT planner, the flaw selection strategy proposed in [Ghallab94] extends the principle of least commitment to temporal planning. A least-commitment strategy defines a selection criterion applicable to all types of flaw. The flaw preferably chosen at a search step is the one for which the branching factor in the search tree is minimal, i.e. the flaw with fewest resolvers (more details about the flaw resolution can be found in [Lemai-Chenevier04]).

5.3.3 The deployed `T-ReX` architecture

To ease the work of the planner, the idea is to divide its work into subcomponents called *reactors* ($R = \{r_1 \dots r_n\}$), according to the principles initially introduced in IDEA [Muscettola02]. Each reactor is associated to achieve specific goals and needs to be synchronized with the other reactors. Every reactor has a planner with a different *planning horizon* λ_r, that represents the time to deliberate and a *look-ahead* π that represents the window of planning, where the planner can order tokens. Every reactor can be deliberative or reactive, depending on the horizon length and the deliberation time. Each reactor owns a number of timelines called internal timelines and a number of observable timelines called external timelines (which are internal to another reactor). To avoid incompatibilities between timelines and to guarantee a converging update process, each of them is owned by only one reactor. For internal timelines, the reactor can modify the execution order of the tokens. For external timelines, the reactor cannot modify the execution order and can just observe the values on the timeline.

The reactor: Figure 5.10 shows the architecture of one reactor in the `T-ReX` architecture.

The main elements of the reactor are the following:

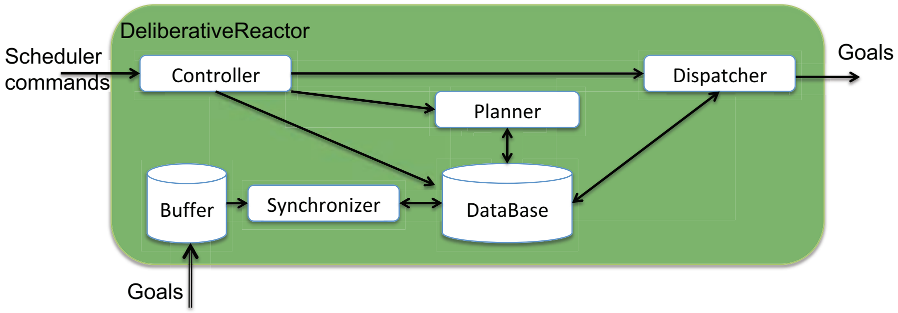


Figure 5.10 – *The structure of a reactor in the* `T-ReX` *architecture*

- Buffer: It records goal requests and observations. This is important to clearly distinguish observations from planner choices.

- Plan database: It stores the partial plan.

- Planner: It is an algorithm to resolve flaws in the plan. The planner looks out in the future by a specified horizon that is defined in the reactor configuration. The `T-ReX` architecture uses the EUROPA planner [Frank03].

- Synchronizer: The synchronizer is a second instance of the EUROPA solver, configured to only solve a restricted set of flaws at the current tick.

- Dispatcher: It is an algorithm to dispatch goals to internal timelines and observations to external timelines. Goals are dispatched according to the timing capabilities of the server. At each clock time the dispatcher send goals from one external timeline to the owner of the timeline.

- Controller. It implements methods invoked by the agent for handling clock start and stop events. It controls the planner, synchronizer and dispatcher.

We now explain in details the main used components in the `T-ReX` architecture.

5.3.4 A database

A database is the *planning instance* which holds the current plan, all the tokens to the planner, the constraints between tokens and the associated timelines.

5.3.5 A deliberative (planner)

A planner is used to populate timelines with tokens according to what is defined in the database. Each reactor has a planner with a different *planning horizon* λ_r and *look-ahead* π.

The EUROPA [Frank03] planner is composed by the *planning domain* (D) and the planning problem P that are defined in the database. Additionally, the planning instance (I) has a *planning horizon* (H). This horizon means that the planner only cares about tokens of the system in the temporal window (or the look-ahead) between $[0, H]$. The initial configuration I is a set of tokens placed on their associated timeline. The figure 5.11 shows different time scopes used in the `T-ReX` architecture. The following example explains the different reactors and the associated time scope of the figure 5.11.

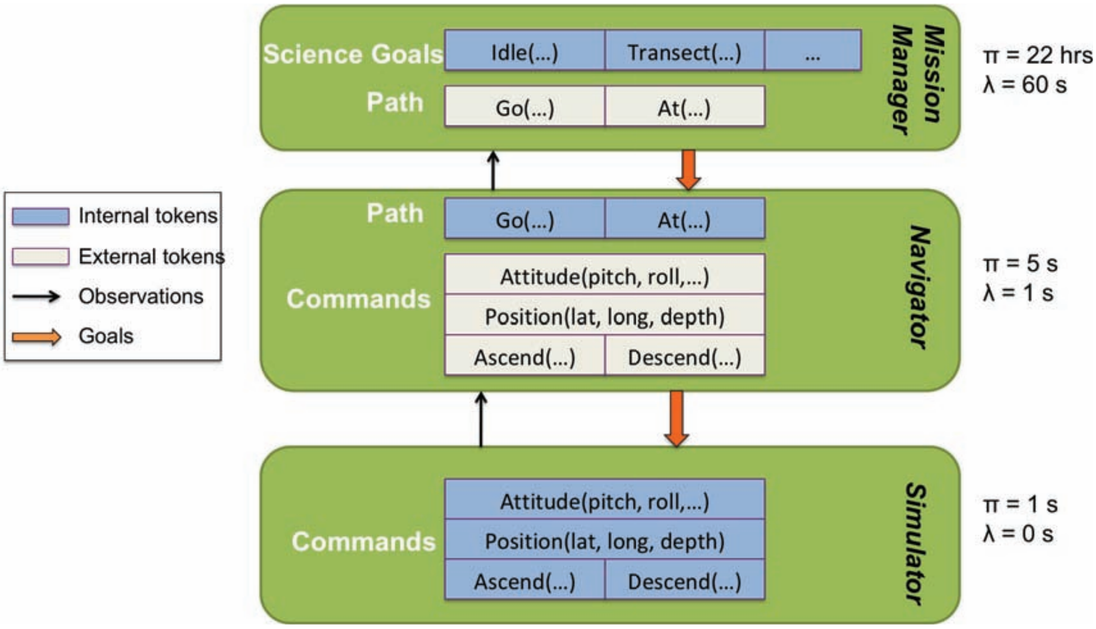

Science Goals
Idle(...)
Transect(...)
...
Path
Go(...)
At(...)
Mission Manager
π = 22 hrs
λ = 60 s
Internal tokens
External tokens
Observations
Goals
Path
Go(...)
At(...)
Commands
Attitude(pitch, roll,...)
Position(lat, long, depth)
Ascend(...)
Descend(...)
Navigator
π = 5 s
λ = 1 s
Commands
Attitude(pitch, roll,...)
Position(lat, long, depth)
Ascend(...)
Descend(...)
Simulator
π = 1 s
λ = 0 s

Figure 5.11 – *Illustration of different temporal scopes and tokens used in the `T-ReX` architecture at MBARI. Three different reactors are shown, with different look-ahead and planning horizon. The dark color represents the internal timeline and the fair color represents the external timeline [McGann08].*

> The *Mission Manager* reactor has to execute a transect[6] with a *look-ahead* of 22 hours (the whole mission) and a *planning horizon* of 60 seconds. The *Mission Manager* reactor can send goals to the *Executive* reactor. This last reactor has less time to deliberate (equal to 1 second) and a lower *planning horizon* (equal to 5 seconds) than the *Mission Manager* (see figure 5.11). In this example, the *Mission Manager* reactor is the deliberative reactor and the *Executive* reactor is the reactive reactor with less deliberation time.

In the figure 5.11 the *Mission Manager* can modify the transect timeline but it can just send goals to the *Go* timeline.

The planner is responsible for producing a consistent plan to accomplish the goals in this horizon, for a given observed system state.

The planner solver contained in the planner (see figure 5.10) is an instance of the EUROPA Solver. In the `T-ReX`, the configuration of a solver is easy to change.

---

[6] It is a straight line, defined by a beginning and end point.

5.3.6 A dispatcher and a synchronizer

The dispatcher allows goals(G_r) to be transmitted to other reactors. This is done at each unit of time called a tick.

The synchronizer coordinates observations (O_r) from other reactors. These observations are given by external timelines (E_r). These external timelines can be observed but cannot be modified by this reactor. This synchronizer is a second instance of the EUROPA solver, configured to resolve a restricted set of flaws at the current tick. If a consistent update for a token is possible, the synchronization will be done very quickly. When there is a failure (inconsistency), it will force the controller to throw away prior planner commitments.

The dispatcher is an algorithm that sends goals and facts to external and internal timelines respectively. Goals are dispatched according to the timing capabilities of the server. Observations (facts) are dispatched to all observers (i.e. any reactors referencing this timeline as an external timeline) if the value changed at that tick.

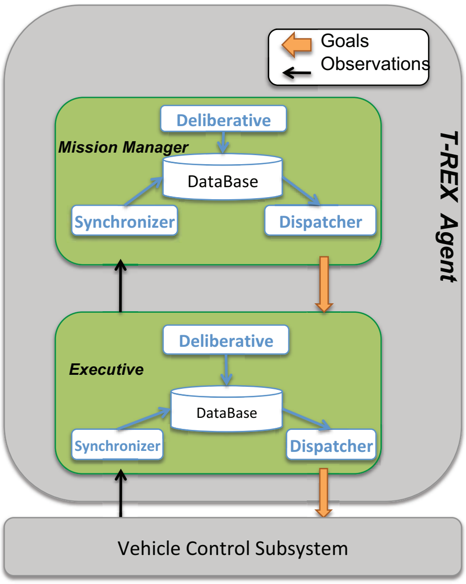

Goals
Observations
T-REX Agent
Mission Manager
Deliberative
DataBase
Synchronizer
Dispatcher
Executive
Deliberative
DataBase
Synchronizer
Dispatcher
Vehicle Control Subsystem

Figure 5.12 – *Illustration of the* `T-ReX` *Architecture with two reactors. The Mission Manager reactor is a deliberative reactor and the Executive reactor is the reactive reactor.*

5.3.7 The developed reactors in the `T-ReX` architecture

The `T-ReX` architecture at MBARI uses two reactors and the VCS (Vehicle Control Subsystem). The different used reactors (see figure 5.12) are:

- The Mission Manager represents the high level reactor that satisfy the scientific objectives. This reactor has a *look-ahead* of the whole mission.

- The Executive reactor is the reactive reactor. It takes into account all generated goals from the Mission Manager, plan and sends behaviors to the vehicle controller. The planning horizon of this reactor is smaller than the Mission Manager reactor; that is why it is considered as a *reactive* reactor.

- The Vehice Control Subsystem (VCS) gives an interface to the existing AUV functional layer and encapsulates access to commands and vehicle state variables. The VCS has a latency equal to zero.

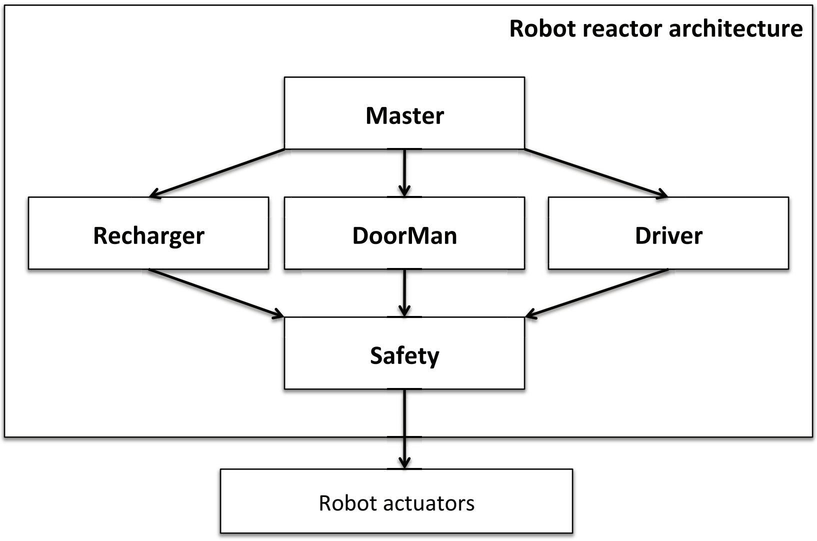


Figure 5.13 – *Illustration of the `T-ReX` Architecture used at Willow Garage.*

5.3.8 The `T-ReX` architecture for PR2

The `T-ReX` architecture has been also used at *Willow Garage* by Connor McGann for the PR2 (Personal Robot 2) robot [Meeussen10]. This robot is able to move all around the office, open doors and when it needs energy, it can plug in a power source in the laboratory. The `T-ReX` architecture is used as an executive controller that executes and integrates these high level tasks. The `T-ReX` architecture of PR2 is composed by (see figure 5.13):

- master: this reactor is the high level planner. It is responsible of breaking the navigation into traverses of open regions and doorways. These regions are helpful when the robot needs to plugin or to open doors.
- driver: this reactor is responsible of driving the vehicle from one place to another, depending on the high level tasks given by the master reactor.
- doorMan: this reactor detect the door and can compute the distance between the robot and the door.
- recharger: this reactor is related to management of the battery power.

- safety: this reactor ensures safety during navigation, for example obstacle avoidance, or checking that the robot arm or power plug are stowed before the robot moves.

5.4 DESCRIPTION OF THE IMPLEMENTED MODEL AND THE ALGORITHMS IN THE COT−REX ARCHITECTURE

To find a cooperative architecture that is reactive and deliberative the robot has to achieve its mission and adapt its plan. In the scenario we consider, the objective is to detect and localize targets. We assume that the vehicle has a predefined ordered set of goals to achieve (way points, communication rendezvous) that defines *a priori* exploration strategy. We added the MapReactor as a new reactor in the `T-ReX` architecture: it is the component that takes into account the perception of the vehicle and generates new goals for the Mission Manager, using the processes presented in chapter 3 and 4.

To make each vehicle cooperative we added a CoopReactor that gets sensory data from other vehicles and passes them to the MapReactor and vice versa. Figures 5.14 and 5.15 show our proposed architecture respectively in simulation and real mode, where each reactor has a specific role. The new architecture is called `CoT-ReX`, where the first two reactors are similar to the one proposed by the MBARI (see figure 5.12). This architecture consists of the following reactors:

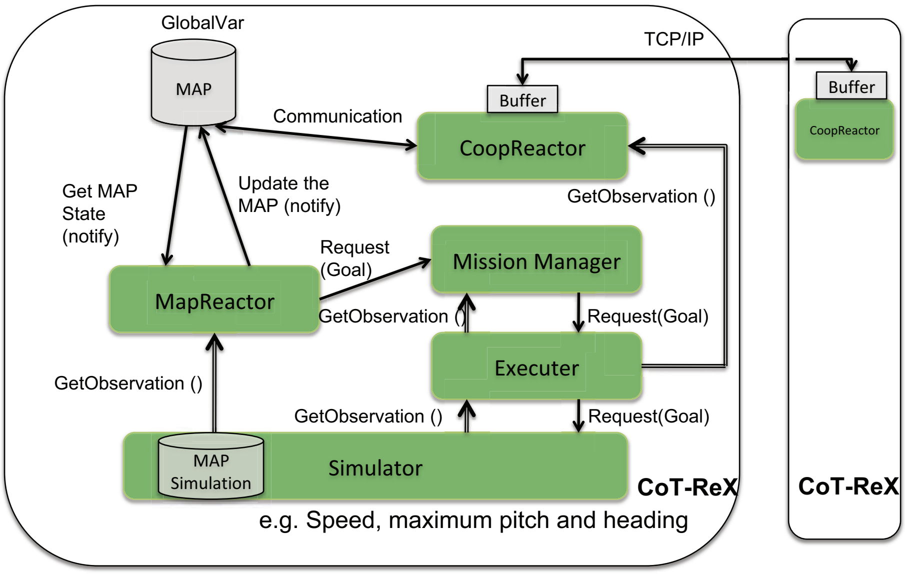

GlobalVar
MAP
TCP/IP
Buffer
Communication
CoopReactor
Buffer
CoopReactor
Get MAP State (notify)
Update the MAP (notify)
GetObservation ()
Request (Goal)
Mission Manager
MapReactor
GetObservation ()
Request(Goal)
Executer
GetObservation ()
GetObservation ()
Request(Goal)
MAP Simulation
Simulator
CoT-ReX
CoT-ReX
e.g. Speed, maximum pitch and heading

Figure 5.14 – *Illustration of the proposed architecture in simulation mode*

- The Mission Manager represents the deliberative reactor. It manages high level goals of all the mission. This reactor has a *look-ahead* of the whole mission.

- The Executer (Functional level) is the reactive reactor. The planning horizon of this reactor

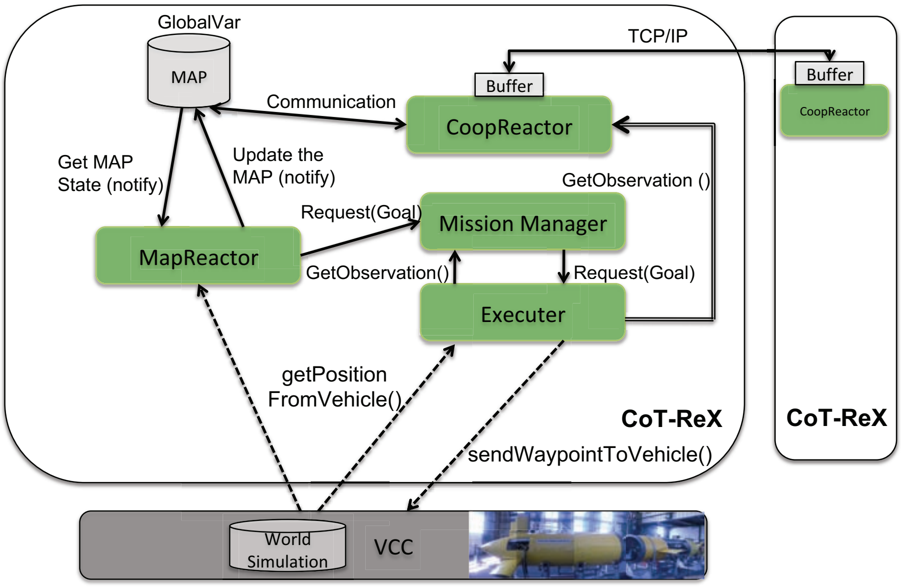

GlobalVar
MAP
TCP/IP
Buffer
CoopReactor
Communication
Buffer
CoopReactor
Get MAP State (notify)
Update the MAP (notify)
GetObservation ()
Request(Goal)
Mission Manager
MapReactor
GetObservation()
Request(Goal)
Executer
getPosition FromVehicle()
CoT-ReX
CoT-ReX
sendWaypointToVehicle()
World Simulation
VCC

Figure 5.15 – *Illustration of the proposed architecture on the real platform*

is smaller than the Mission Manager reactor.

- The MapReactor (Generation of opportunistic goals) has the role to generate opportunistic goals. These goals are taken into account by the Mission Manager.
- The CoopReactor (Collaborative mechanism) is the cooperative reactor which communicates with the other vehicles through the ASV.
- The Simulator: This reactor simulates the vehicle motion by defining maximum and minimum limits of the vehicle speed, pitch, heading, depth, etc.

5.4.1 The timelines in the Mission Manager

The different timelines in the mission manager reactor (see figure 5.14) are explained below:

- MissionGoals: This internal timeline contains three different tokens: *Inactive*, *Pickup* and *PathSurvey*. The *Inactive* token does not do anything. The *Pickup* token lets the vehicle to take sample of a specific area. The *PathSurvey* is a token that defines according to two points the survey that the vehicle can do. For example, in figure 5.16 the use of the *PathSurvey* defines the two points A, B and the transect width for the exploration. The *PathSurvey* will generate the other waypoints according to a specified distance of transect. Each waypoint is represented by a *Going* from the beginning of the transect to the ending point. Between each generated waypoint there are other constraints of precedences that should be also taken into account. For this example the *PathSurvey contains* different

Going(x,y) tokens that achieve the required path (x and y are the computed waypoints, represented in figure 5.16 by fair circles).

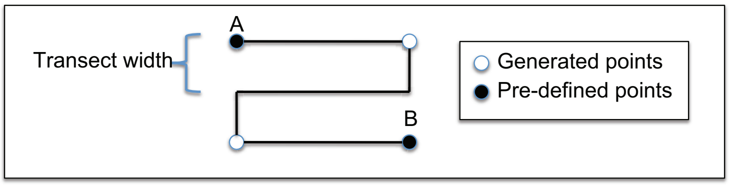


Figure 5.16 – *Illustration of an example using a PathSurvey. The points A and B are the parameter points defined at the PathSurvey and the other points are the generated points using the PathSurvey according to a predefined transect width.*

- PathController: It is an external timeline that implements the constraints of the navigation path. PathController is used for general vehicle positioning and to deal with the path constraints. Note that this is not designed to spatially constrain the vehicle motion in general, but only make the vehicle traverse a path when required to do it. This timeline is active when the token *Going* is active otherwise the associated token is *Inactive*.

5.4.2 The timelines in the Executer

The executer reactor has less look-ahead (5 *ut*) and planning horizon (1 *ut*) than the Mission Manager reactor. The used timelines and the associated tokens are:

- Actions: This timeline is an internal timeline, that is executed on one unit of time. Two tokens are associated to this timeline: *Idle* that does nothing and *DoBehavior* that is used to give properties to other tokens. For example, if a token is preceded by the token *DoBehavior*, at the end of the execution of the *DoBehavior*, the token can be directly executed. In another case, if the token is not preceded by this *DoBehavior*, the token will be executed at the last time of the token interval.

- PathController: This timeline is an internal timeline that allows the vehicle to move and control its path.

- Mutex: It is an internal timeline used to execute one token at time. Two tokens are defined: *InUse* means that there are tokens that are executed, *Free* means that there are no tokens that are executed.

- Communication: This timeline is an external timeline that allows the vehicle to communicate or not. Communication timeline contains a token called *Active* and another *Inactive*. When the token is *Active*, then the vehicle has to communicate, otherwise the communication is not allowed.

- VehicleState: It is an external timeline that gives the state of the vehicle at any time. The state of the vehicle is represented by the depth, northing, easting, etc.

- SetPoint, Ascend, Descend and Waypoint: All these timelines are external and they give the location of the ascend/decscent/setpoint. In the simulation case, the path is computed with predefined functions and values.

5.4.3 The architecture of the MapReactor

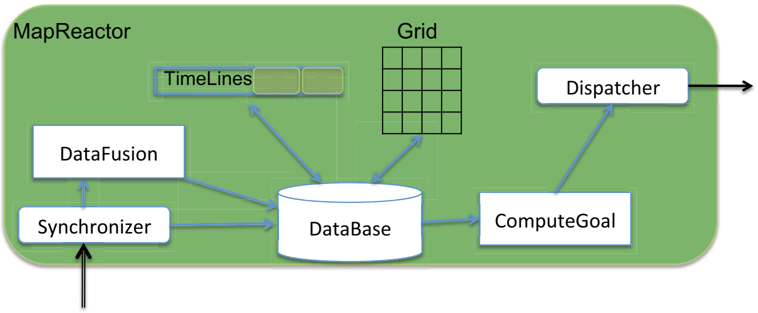


Figure 5.17 – *Illustration of different function in the MapReactor*

The MapReactor (see figure 5.17) component is composed of a synchronizer, a database and a dispatcher. The database contains the map of the whole exploration zone and the different associated timelines. According to the computed map and the existing timelines, the vehicle changes its strategy of exploration by generating goals.

timelines: The implemented timelines are (figure 5.18):

- Communication: This timeline is an external timeline used to know the next nearest communication point. This timeline can be *Active* or *Inactive* depending on the defined execution time. When the MapReactor generates an opportunistic goal, it estimates the needed time to reach both communication point and the opportunistic goal (see chapter 4 for more details). To validate the new goal, the MapReactor needs to know the predefined time of the communication token. This is why the communication token is necessary for the MapReactor.

- VehicleState: This timeline is external to the MapReactor and is used to provide the vehicle state for example *heading*, *nothing*, etc.

- SetPoint, Ascend and Descend are the behaviors of the vehicle, that are external timelines.

Algorithms: To allow the vehicle to change its exploration strategy we have to represent the perception of the vehicle, the used timelines, the fusion of the measured data with preceding measurement, and finally find a function to compute the next goal (see chapter 4). The algorithm for exploring a discrete environment (grid) is outlined below:

1. Update the grid using the measured data.
2. Make decisions: This allows to define adaptive strategies, in which the AUVs selects the next motions in order to confirm the presence of a target.
3. Adaptive Cooperative Exploration Strategy (ACES) (see chapter 4).

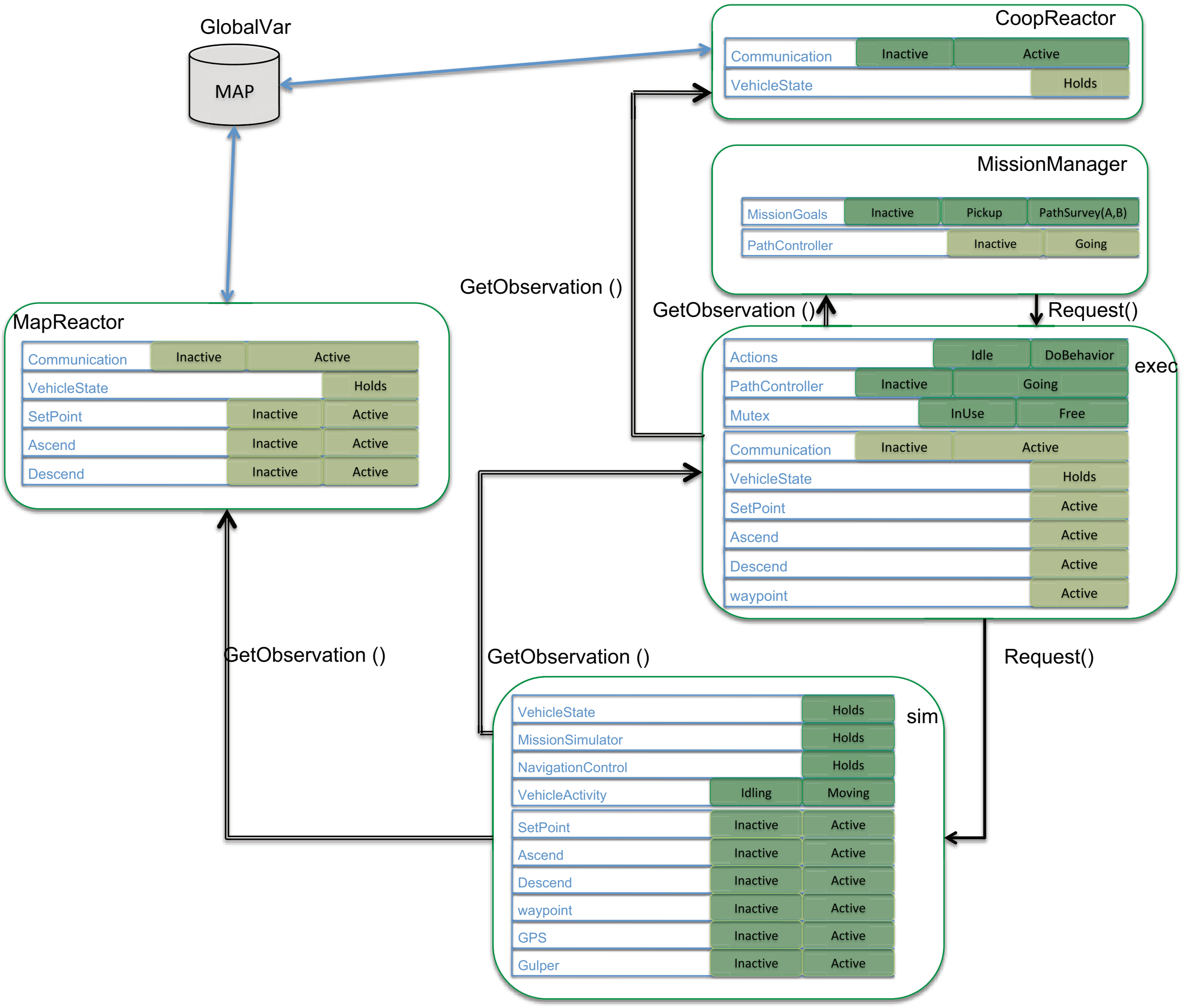

GlobalVar
MAP
CoopReactor
Communication
Inactive
Active
VehicleState
Holds
MissionManager
MissionGoals
Inactive
Pickup
PathSurvey(A,B)
PathController
Inactive
Going
GetObservation ()
GetObservation ()
Request()
MapReactor
Communication
Inactive
Active
VehicleState
Holds
SetPoint
Inactive
Active
Ascend
Inactive
Active
Descend
Inactive
Active
exec
Actions
Idle
DoBehavior
PathController
Inactive
Going
Mutex
InUse
Free
Communication
Inactive
Active
VehicleState
Holds
SetPoint
Active
Ascend
Active
Descend
Active
waypoint
Active
GetObservation ()
GetObservation ()
Request()
sim
VehicleState
Holds
MissionSimulator
Holds
NavigationControl
Holds
VehicleActivity
Idling
Moving
SetPoint
Inactive
Active
Ascend
Inactive
Active
Descend
Inactive
Active
waypoint
Inactive
Active
GPS
Inactive
Active
Gulper
Inactive
Active

Figure 5.18 – *Illustration of different timelines used at each reactor in the* `T-ReX` *architecture*

5.4.4 The architecture of the CoopReactor

To allow the cooperation between vehicles, they have to be at predefined rendezvous points to communicate with the ASV. These rendezvous points allow vehicles to exchange information. To manage these exchanges we added a new reactor called “CoopReactor”. This reactor is composed of a synchronizer, a dispatcher and a database. This database contains predefined rendezvous points for the whole mission. The internal timeline of this reactor is the Communication timeline, when the used token is *Active* than the communication is active otherwise the token *Inactive* is used (see figure 5.18).

In our experiments a client/server protocol is implemented where data can be exchanged between vehicles. The exchanged data can be the explored grid or the failed tasks. All these exchanged information can help the vehicle to localize the target.

CONCLUSION

In this chapter we presented a cooperative architecture for multiple robots with the possibility to communicate. This architecture allows each vehicle to be autonomous and to use the collected information by the fleet of vehicles to localize the target. We used an existing architecture for planning and execution control (`T-ReX`) which allows for deliberation as well as reactive behaviors.

Two new reactors were implemented to improve the predefined strategy for area coverage exploration and to allow for cooperation.

The MapReactor was tested using a simulator platform at IFREMER. This is an “hardware in the loop” simulator which uses the same controller than the one used on `Asterix` (one of IFREMER’s AUVs): these tests are presented in section 6.3 of the next chapter.

In the next chapter, we show the realization of various target detection cases to illustrate the capability of our architecture to embed cooperative algorithms.

6

SIMULATION RESULTS

We illustrate the capacities of our system for a target detection scenario in different configurations. We then show the integration of our architecture with the simulation environment at IFREMER. The results we get are encouraging in terms of portability of our system.

EDWARD DEBONO, SAYS: « Creativity involves breaking out of established patterns in order to look at things a different way.»

6.1 INTRODUCTION

To prove the flexibility, extensibility and efficiency of the Adaptive Cooperative Exploration Strategy (ACES), it is necessary to evaluate the cooperative algorithms by using different case studies. For this purpose, we defined different cases that uses two AUVs and one ASV. All of the cases are for target localization and are evaluated in terms of number of located targets, the errors between the constructed map and the real world, and the type of exploration strategy (adaptive or not).

We begin this chapter by first describing the obtained results in our testbed. The obtained results using the IFREMER simulator are then explained in section 6.3. Finally, we conclude the chapter by discussing the obtained results and the future extensions.

6.2 SIMULATION RESULTS FOR TARGET DETECTION

6.2.1 Simulation environment

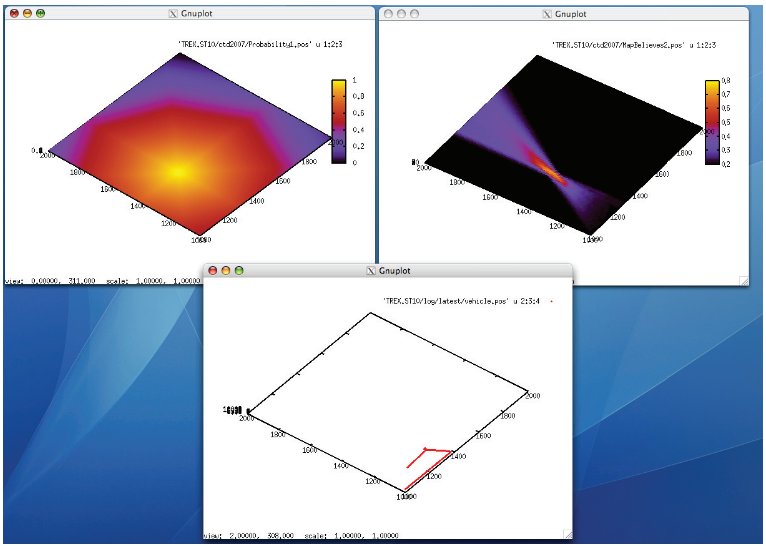

Gnuplot
'TREX.ST10/ctd2007/Probability1.pos' u 1:2:3
Gnuplot
'TREX.ST10/ctd2007/MapBelieves2.pos' u 1:2:3
Gnuplot
'TREX.ST10/log/latest/vehicle.pos' u 2:3:4

Figure 6.1 – *An example of an obtained map and trajectory using one underwater vehicle.*

The architecture we used is a cooperative `T-ReX` architecture that includes the exploration strategy ACES. Each cooperative `T-ReX` architecture represents one underwater vehicle. Every vehicle simulation is run in a MAC OSX, 2GHz Intel Core 2 Duo processor and a memory of 2 Go.

We use `Gnuplot` to show the path evolution of the vehicle and the final map obtained for each vehicle. Figure 6.1 shows an example of an obtained map (represented in the upper right of the figure) and the trajectory of the AUV (represented at the bottom part of the figure). The upper left part of figure 6.1 represents the real world (the darker the color, the lower the target presence is).

Moreover, in our experiments we represent, in figures, waypoints as cross points and communication points as circles. In our studies the marine target is a hotspot, and we only considered its temperature properties.

6.2.2 Mission definition and role of each underwater vehicle

We simulate how underwater vehicles collaboratively locate the targets. In the first case study, our aim is to find the best predefined exploration strategy for underwater vehicles and trying different strategies that can be used for underwater vehicles. The ASV moves at the surface and the AUVs explore the deep sea to locate targets. The ASV is both used to help the communication between AUVs at predefined communication points and to re-localize AUVs. The AUVs, on the other hand, are used to execute pre-defined missions with a possibility of modifying the trajectory.

In the second case study, our aim is to evaluate the best exploration strategy during mission execution. In this sense, two AUVs are required to explore the same area to localize and sample a maximum number of targets. AUV2 explores the area at a deeper depth than AUV1. AUV1 explores and localizes targets, and also plays the role of the ASV to re-localize AUV2.

In the last case study, three underwater vehicles are deployed to localize one single target. Two AUVs are at different levels and the ASV is at the surface but can also collect data at the surface to localize the target. In the first and the second case of study, the ASV is replaced by the AUV1. This AUV is at 10 meters deep from the surface, that makes it easy to re-localize itself and the other vehicle.

6.2.3 The phases of case studies

All case studies are composed of two main phases:

1. Mission preparation: The operator specifies the exploration strategy of each vehicle (e.g. waypoints). In this manner, the operator can use different kinds of predefined strategies as shown in figure 6.2. We will evaluate these types of exploration strategies in the first case study.
2. Mission execution: The AUVs explore the area using a predefined exploration strategy (defined at the mission preparation phase). This strategy can then be modified by using the data collected from the deep sea to have a precise location of the targets. The second and third case studies show the obtained target detection using two and three underwater vehicles respectively.

6.2.4 Case 1: Evaluation of cooperative predefined exploration strategies

In the first experiment, we use the predefined exploration strategy shown in figure 6.2.(a). Figure 6.3 represents the mission preparation, where predefined waypoints are shown as crosses and RDV points as black circles.

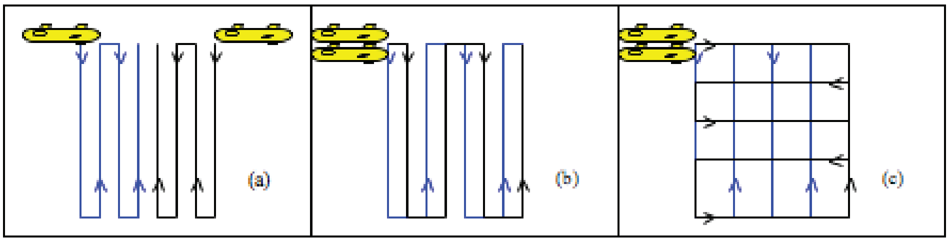


Figure 6.2 – *Different predefined exploration strategies that can be used for underwater vehicles.*

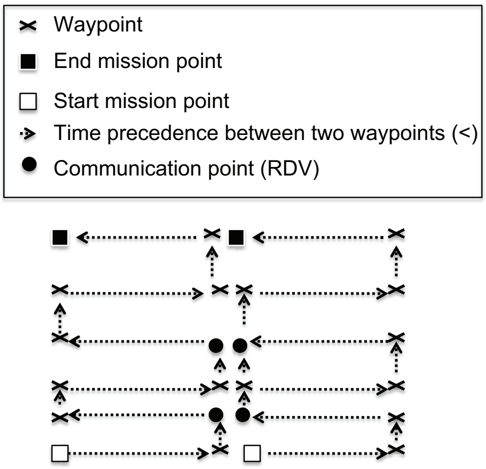
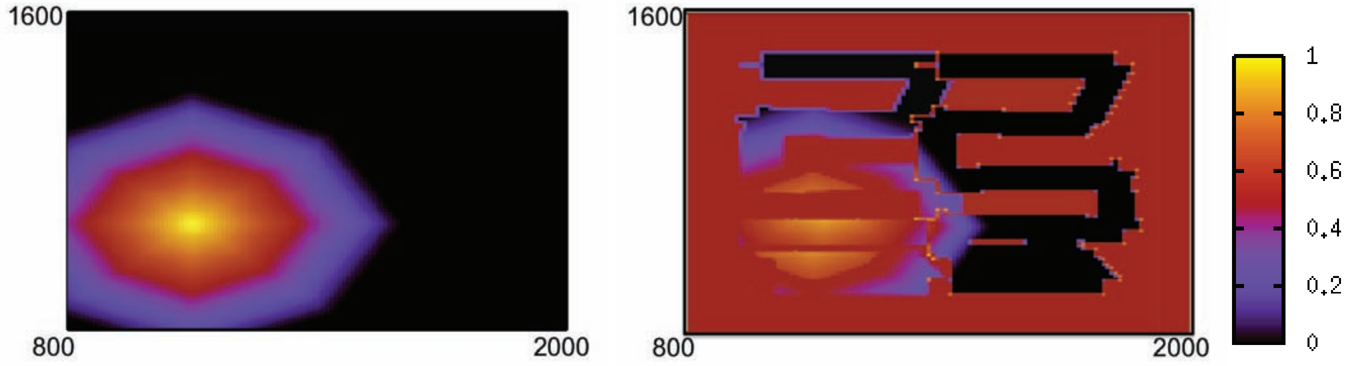

Waypoint
End mission point
Start mission point
Time precedence between two waypoints (<)
Communication point (RDV)

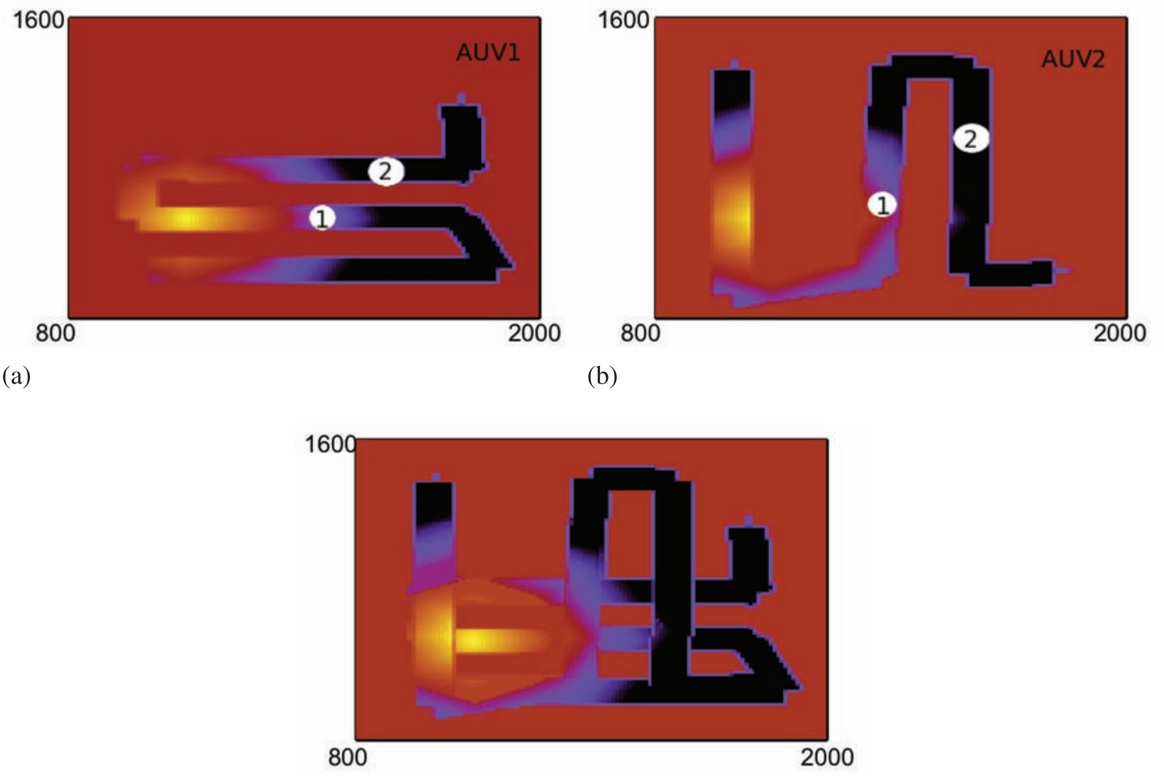
Figure 6.3 – *All predefined waypoints and communication points for each underwater vehicle. Arrows represent the temporal execution precedence between two waypoints. This precedence determines the transects of each underwater vehicle.*



(a) (b)

Figure 6.4 – *Illustration of a map that results from an exploration strategy of two underwater vehicles using the predefined mission described in figure 6.3. (a) shows the real world. (b) shows the resulting trajectory and the obtained map at the end of the mission execution of both vehicles.*

Figure 6.4b shows the obtained exploration strategy and the map at the end of the mission execution using a non-adaptive exploration strategy. The map resulting from an adaptive cooperative exploration strategy is not different than an adaptive exploration strategy without coopera-



(a)

(b)

(c)

Figure 6.5 – *Illustration of the obtained results using two AUVs. The AUVs share the exploration area and communication points. (a) and (b) are the predefined path for AUV1 and AUV2 respectively. (c) represents the obtained path and map after execution*

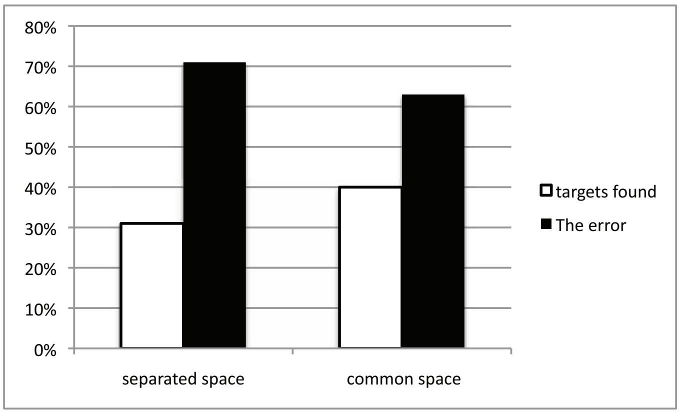


Figure 6.6 – *Statistical results comparing two types of predefined exploration strategies with separated and a common space. The two criterions of comparison are: (a) the percentage of found targets (b) and the error of the obtained map at the end of the mission exploration.*

tion. This result is a consequence of the separated exploration space of each AUV: an AUV does not add any information to the other vehicle measures.

That is why, we want to show that other predefined types of exploration strategies can help each vehicle for a better map construction and in general it is the same with cooperative approaches.

In the second experiment, we use the predefined exploration strategy shown in figure 6.2.(c). Figure 6.5a and figure 6.5b, respectively, shows a predefined exploration strategy for both AUV1 and AUV2. Two communication points are represented with white circles on each path of AUV1 and AUV2. At the first communication point AUV2 has more information about the left area than AUV1. After communication, the AUV1 modifies its exploration strategy to verify and find more precision on what the AUV2 has found. Figure 6.5c represents the obtained map at the end of the mission execution. Comparing this map with the precedent obtained map (see figure 6.4b), we can see that the actual map is more precise about the target location than the other map. This result justifies the importance of sharing collected information, when each vehicle share a common explored area. To prove this result, we have performed experiments with different initial exploration areas. The constructed map, when using the exploration strategy with separate exploration area is less precise than using a common exploration strategy.

Figure 6.6 shows statistical results comparing the map error between the exploration strategy shown in figure 6.2.(a) and the one shown in figure 6.2.(c). The map error reprensent the differences between the world map and the obtained probability map at the end of the mission. It is computed as follows:

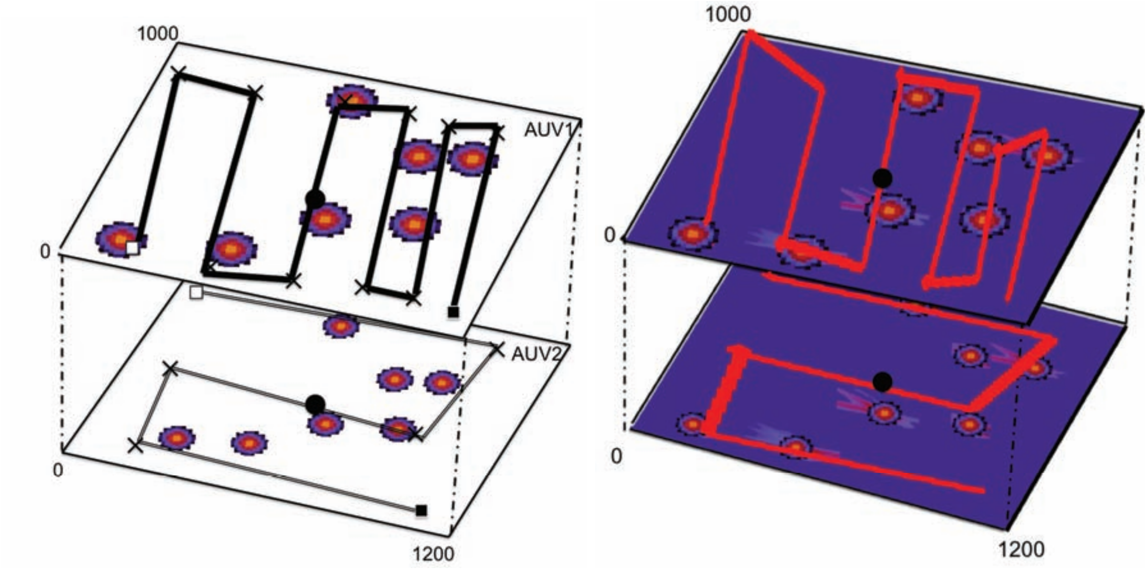
$$\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}|P(Xreal_{i,j}^{k}) - P(x_{i,j}^{k})|}{N+M}$$

$P(Xreal_{i,j}^{k})$ is the real value of the target propagation at (i, j) place. The k value represents the end time of the mission. $P(x_{i,j}^{k})$ is the measured or the predicted value of the target propagation at (i, j) place at the end of the mission (k). N and M are the dimension of the grid.

In figure 6.6 a comparison between the separated exploration space strategy and the common space exploration strategy. For the separated space exploration strategy, the number of found targets is 31% where the error is 71%. For the common space exploration, the number of found targets is 40% and the error is 63%.

These results clearly show that the best exploration strategy is the one with common exploration space. These results confirm that for exploration strategies the cooperation is useful when the exploration space is in common or, in other words, when there is redundant exploration coverage.

For the other scenarios, we use this common exploration space as a predefined exploration strategy.



(a) (b)

Figure 6.7 – *Illustration of the obtained results using two AUVs. Each AUV has a common exploration area and communication points. Figure 6.7a is the predefined path for AUV1 and AUV2. Figure 6.7b represents the obtained path and the constructed map, after execution, for both AUVs.*

6.2.5 Case 2: Evaluation of cooperative exploration strategies at different depths

In this simulation results, we compare three kinds of exploration strategies as follows:

1. Non-adaptive exploration strategy: This strategy has predefined waypoints and does not adapt its strategy to build a source map. There is a time window to reach each waypoint. The order of these waypoints implicitly defines exploration transects.
2. Adaptive G.I.D exploration strategy: The vehicle heads toward the closest source hypoth-

esis until its probability exceeds P_{loc}. A high value denoted P_{loc} leads to a confirmed well localized target.

3. Adaptive G.I.G exploration strategy: The vehicle heads toward the closest source hypothesis until its probability exceeds P_{conf}. This probability is a lower value that leads to the confirmation of a source presence, but not precisely localized. This strategy favors the reduction in the number of vehicle actions, at the cost of less precisely localized sources.

Each predefined strategy has a common exploration space with another vehicle. Figure 6.7 illustrates the predefined mission and the obtained map for each AUV1 and AUV2. Figure 6.7a is the predefined exploration strategy for AUV1 and AUV2 at different depth with different target model. The target model is more precise for AUVs that are deeper and in our case, the AUV2 has a more precise location than the AUV1.Figure 6.7b shows the obtained map at mission execution end of AUV1 and AUV2. In this example, seven targets are defined and all of them are localized.

To generalize precedent experiments, different target locations are generated, where the average error and the number of targets found for each exploration strategy are shown in figure 6.8.

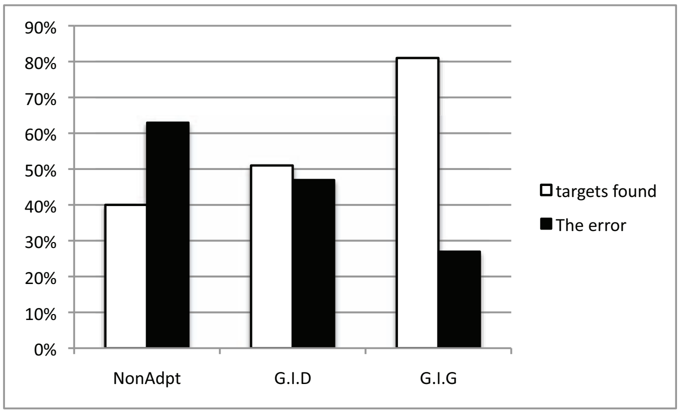


Figure 6.8 – *Evaluation of three different exploration strategies according to the percentage of found targets and the error between the obtained map and the real world map.*

The non-adaptive exploration strategy found 40% of the existent targets. This result is less than the G.I.D and G.I.G exploration strategy. We can conclude from this statistical results that the best exploration strategy is the G.I.G exploration strategy with 80% of found targets with 27%of error.

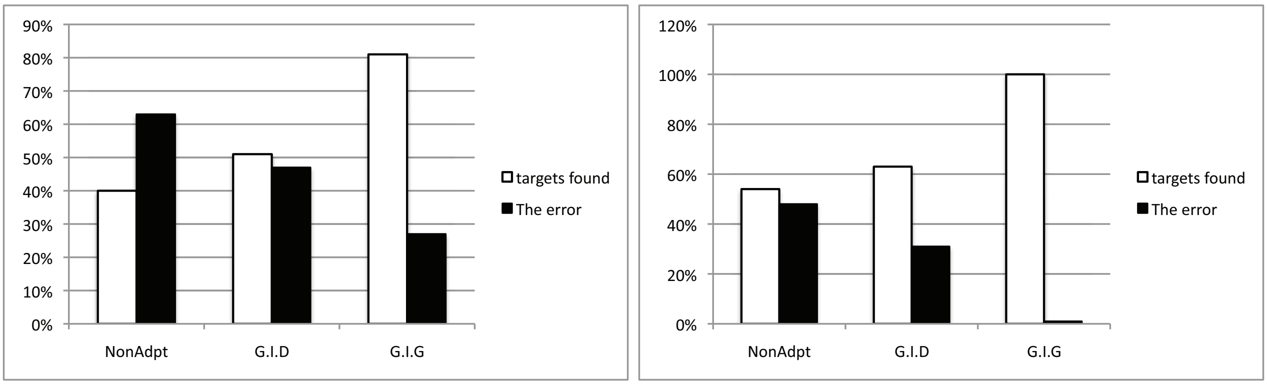
From these results a question rises that is how can we get 100% of found targets?

To improve the result to a 100% of target found, we have given other experiments, where we changed two criterions as follow:

- The sensor range (r): It is the predefined range of a given sensor.
- The transect width (W): This value is related to the predefined transect in the mission.

To evaluate these two criterion relatively with each other, we generate a new criterion called coverage rate (CR). This coverage represents the percentage of redundant exploration area and is computed as r/W. Figure 6.9a and figure 6.9b represent the error and the percentage of found targets for a coverage rate of 100% and 125% respectively. We can conclude from these two figures that the number of found targets for a percentage of coverage rate (CR) equal to 125% is

100%, with a negligent error of 0.1%. This redundancy is useful, to localize targets, where for each strategy we should find the best percentage to avoid unused redundancy.

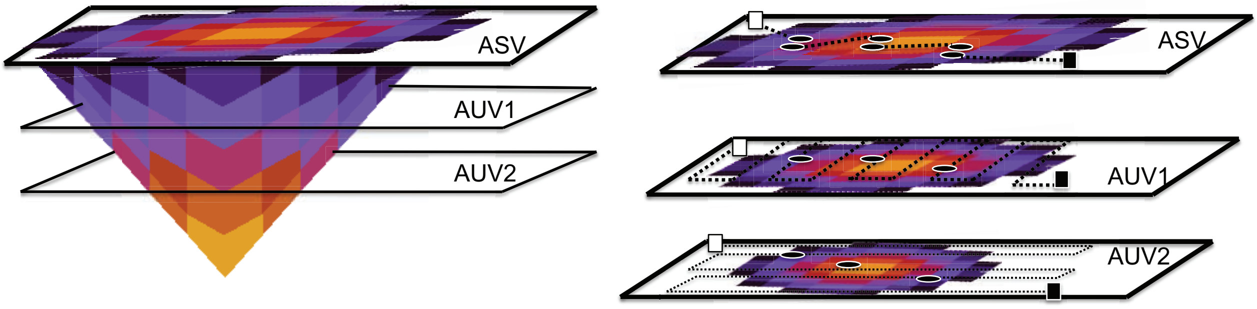


(a) (b)

Figure 6.9 – *Illustration of different obtained results varying the coverage rate (CR) from 100% to 125%.*

6.2.6 Case 3: Exploration strategies at three different depths

Figure 6.10a shows the evaluated case to find one target. The target shape in this figure is without any marine current, but at different depths the target model changes, according to the model presented in chapter 3, figure 3.3. Three layers are represented in this figure. The upper layer is the real world for the ASV, the second layer is the real world for the AUV1 and the last layer is the real world for the AUV2. For each underwater vehicle, a predefined exploration strategy and RDVs points are shown in figure 6.10b, where black circles are the RDVs with the ASV. White and black rectangles represent the beginning and the ending mission of underwater vehicles respectively. The dashed lines represent the predefined path for each underwater vehicle.



r

(a) (b)

Figure 6.10 – *Illustration of the predefined exploration strategy for each underwater vehicle to shape one target. Communication points are represented with black circles and the beginning and end point with white and black rectangle in oder.*

The figure 6.11 shows the obtained maps at the end of the mission of each underwater vehicle. On the basis of these maps, one could reconstruct a fairly faithful model the temperature plume.

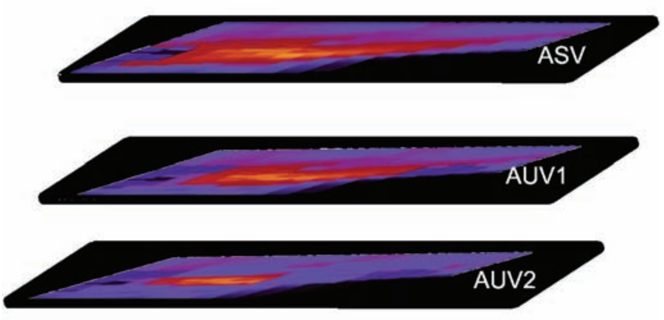


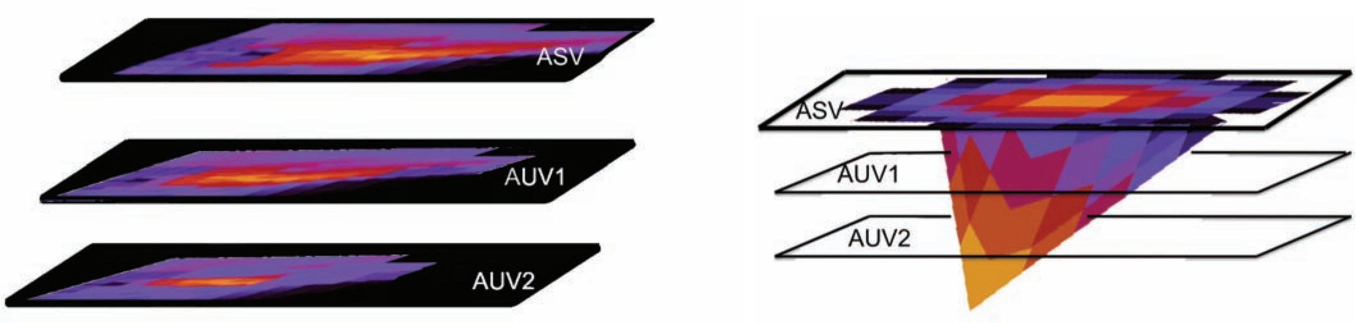
Figure 6.11 – *Illustration of the obtained map using three underwater vehicles.*

Figure 6.12a shows an example of a result of the target model with an heterogeneous marine current. In this example, the vehicles do not know that there is a marine current. Nevertheless, the integration of these maps can yield the estimation of the plume model, as illustrated figure 6.12b.

6.3 SIMULATION RESULTS USING IFREMER SIMULATOR

6.3.1 Simulation environment

We integrated our architecture with the AUV simulator developped at Ifremer. The considered AUV is `Asterix`, for which the following characteristics are set:
- Maximum depth = 3000 meters
- Drift = 2m/h
- Min/Max speed = [0.8, 1.5] meters per seconds.
- Min/Max pitch = [-10.0, 10.0] degrees.



(a) (b)

Figure 6.12 – *Illustration of the target shape (with marine current) and the obtained map using three underwater vehicles.*

- Min/Max roll = [-10.0, 10.0] degrees.
- Maximum exploration distance for one mission = 100 km.

6.4 IMPLEMENTATION AT IFREMER

6.4.1 Mission definition and simulation

Three steps for the mission definition are used at IFREMER [Manual]:

1. Offline mission programming: this mission preparation is done at the surface. The mission is programmed for one AUV using the MIMOSA tool (detailed in the next subsection). At the end the mission is uploaded in the vehicle. In the simulation case, the mission is uploaded to the VCC (Vehicle Control Computer).
2. Mission execution: During mission execution the mission is executed and all the collected data are saved for the next step. The VCC (Vehicle Control Computer) behaves exactly like the real vehicle, this is why, the simulation is hardware in the loop.
3. Recovery and processing the collected data: This allows to collect the gathered data from the underwater vehicle from the mission preparation until the mission execution.

MIMOSA (Mission Management fOr Subsea Autonomous vehicles)

MIMOSA is a tool that manage the definition of missions for an AUV. This tool allows the operator to define the mission using the information contained by a GIS [1]. A navigation is composed by a set of missions, operational constraints and constraints related to the environment (e.g. forbidden regions). Each mission contains a set of used underwater vehicles characterized by (figure 6.13):

- Mission plan: the plan is represented by dives and trajectories.
- Payloads: this is allows the definition of different types of payloads that can the vehicle carry for example a camera.
- Parameters: these parameters are the available energy for the vehicle, the maximal distance between waypoints, maximal and minimal vehicle speed, etc.

6.4.2 Mission experimentation

In our case, we defined several waypoints that the vehicle has to achieve during intervals. The used architecture is represented in figure 6.14. The goals and the position of the vehicle are given by the commands $sendWaypointToVehicle()$ and $getPositionFromVehicle()$ respectively. The used protocol to communicate between `T-ReX` and the VCC is TCP/IP.

The predefined and the executed mission execution of the Asterix is represented in figure 6.15. The red line represents the predefined mission execution of the underwater vehicle.

---

[1] Geographic Information System.

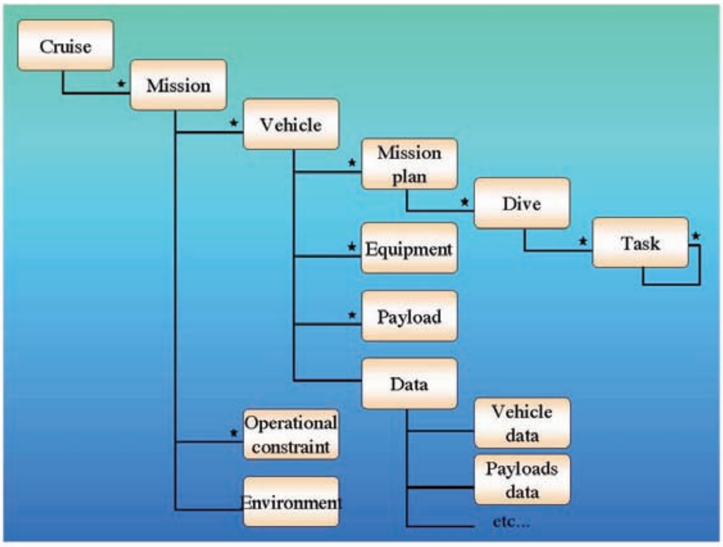

Cruise
Mission
Vehicle
Mission plan
Dive
Task
Equipment
Payload
Data
Vehicle data
Payloads data
etc...
Operational constraint
Environment
*

Figure 6.13 – *Illustration of the link between different mission tasks.*

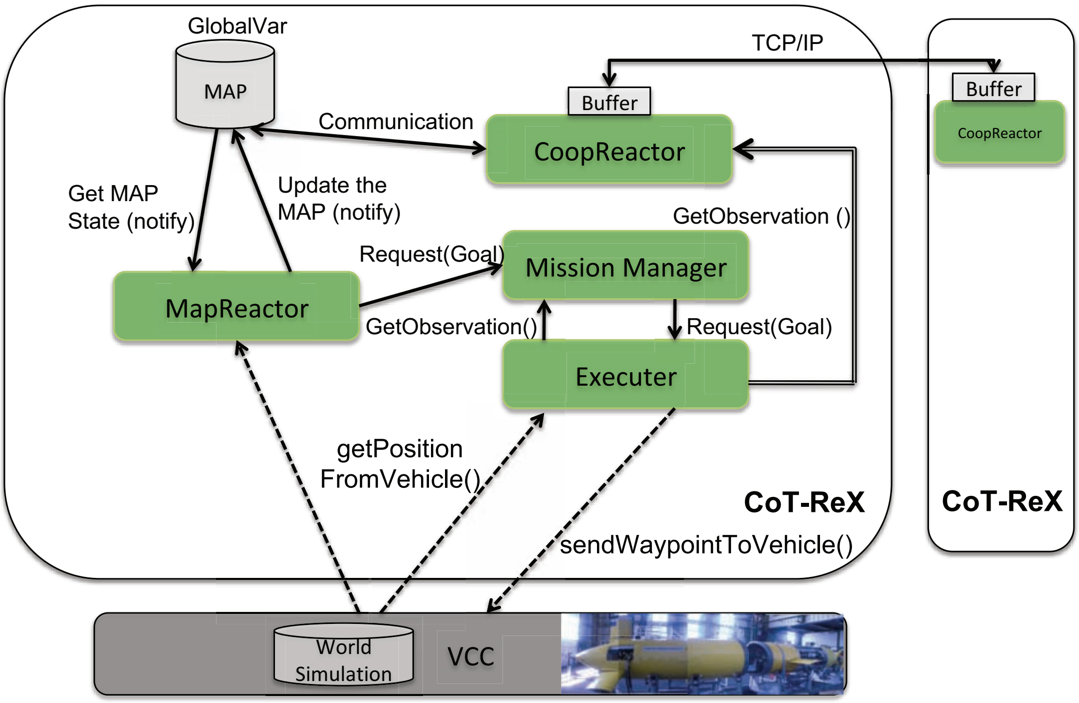

GlobalVar
MAP
TCP/IP
Buffer
Communication
CoopReactor
Buffer
CoopReactor
Get MAP State (notify)
Update the MAP (notify)
GetObservation ()
Request(Goal)
Mission Manager
MapReactor
GetObservation()
Request(Goal)
Executer
getPosition FromVehicle()
CoT-ReX
CoT-ReX
sendWaypointToVehicle()
World Simulation
VCC

Figure 6.14 – *Illustration of the used architecture to link between our implemented* `T-ReX` *architecture and the used architecture at IFREMER.*

The orange line represents the executed mission of the underwater vehicle. The execution path takes into account the real vehicle ability presented in the subsection 6.3.1.

In this section, we show that our system can be connected into the VCC used at IFREMER to send commands and control the execution of the vehicle. However, a real experimentation is needed to validate our approach.

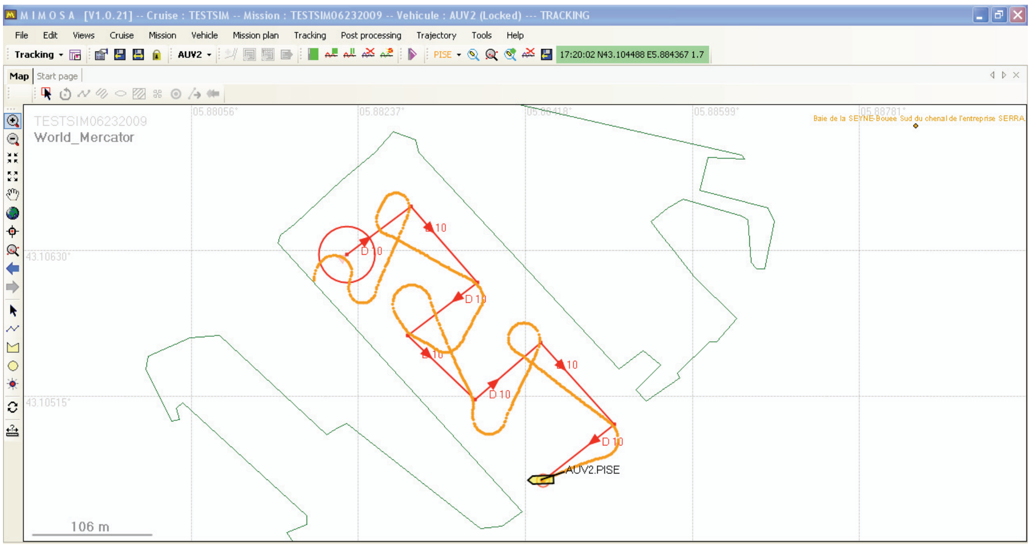

M I M O S A [V1.0.21] -- Cruise : TESTSIM -- Mission : TESTSIM06232009 -- Vehicule : AUV2 (Locked) --- TRACKING
File Edit Views Cruise Mission Vehicle Mission plan Tracking Post processing Trajectory Tools Help
Tracking
AUV2
PISE
17:20:02 N43.104488 E5.884367 1.7
Map
Start page
TESTSIM06232009
World_Mercator
05.88056°
05.88237°
05.88418°
05.88599°
05.88781°
43.10630°
43.10515°
Baie de la SEYNE Bouee Sud du chenal de l'entreprise SERRA
10
D 10
AUV2.PISE
106 m

Figure 6.15 – *The path of the vehicle display on IFREMER simulator.*

DISCUSSION

In this section, we have presented different simulation results. Three case studies were evaluated in our simulator and another case was validated at IFREMER's simulator.

In the first case study the importance of predefined exploration strategy for a cooperative exploration is shown. In the second case study three exploration strategies are evaluated to localize a maximal number of targets, where some criterion has to be taken into account to improve the exploration strategy. These criterion are the sensor range and the transect width. At the end of this case, the G.I.G exploration strategy gives improvement in term of the number of found targets than the other strategies. The third case study uses three underwater vehicles to locate in 3D one target with or without marine current.

As a result, we have shown that our architecture has evaluated different types of exploration strategies and can be used to evaluate other different scenarios involving more than one ASV and two AUVs. Moreover, the use of an already deployed architecture is very encouraging to consider deploying the proposed approach and the selected strategies on real systems.

CONCLUSION AND PERSPECTIVES

DALAI LAMA, SAYS: « Old friends pass away, new friends appear. It is just like the days. An old day passes, a new day arrives. The important thing is to make it meaningful: a meaningful friend - or a meaningful day.»

Our contribution aims at defining an architecture and a framework to evaluate various types of scenarios for multiple autonomous underwater vehicles.

In our experiments, an autonomous surface vehicle is used to collect and redistribute sensor maps, to help the AUVs to relocalize without surfacing, and to act as a communications relay with an operator. Each AUV produces a probabilistic sensor map of possible interesting sources using different types of sensors (range-only or bearing-only), and shares them with the other vehicles to improve their perception of the environment, and find the sources of interest. We propose a mechanism to allow the vehicles to share these maps. We used for our experiments an extension to the `T-ReX` architecture, which has already been used at MBARI to control one vehicle. The architecture `T-ReX` (Teleo-Reactive EXecutive) [McGann07], is a goal-oriented system, with embedded automated planning and adaptive execution. This architecture provides to each vehicle the capability to generate and execute a plan onboard. Rather than being exclusively deliberative or exclusively reactive, this architecture incorporates planners that can deal with different planning horizons and deliberation times. These different times make `T-ReX` both reactive and deliberative.

To extend the `T-ReX` architecture we added two new components (reactors), one for communication with the other vehicles and one to establish new opportunistic goals with respect to the chosen strategy. An exploration strategy is presented where each vehicle has to weight the advantage of accurately localizing sources while respecting time constrained navigation waypoints and communication rendez-vous. Our main contributions are the following:

1. **Models:** We give in this manuscript a model of the target or the hotspot and the different types of sensors used by vehicles (range-only sensors and bearing-only sensors). We also consider simplified models of AUVs motions and AUV/ASV communications.

2. **Adaptive exploration strategy:** We implemented different types of adaptive exploration strategies that use the collected data of the vehicle to direct its path. We found that G.I.G (Global Information Gain driven) has improved non adaptive strategies and other adaptive

approaches in terms of target error location and the number of targets found.

3. **Cooperative architecture:** We deployed a system that can make two underwater vehicles cooperate and exchange collected data at predefined communication points with a reduced amount of data.

4. **Deliberative and reactive system:** We wanted to give a compromise between the data driven approaches and the task driven approaches. In this architecture priorities are given between tasks and generated goals. For example, rendezvous points (considered here as a task driven) is favored over opportunistic goals (these goals have as objective to get more information about the target), but at the same time the approach favors opportunistic goals over waypoints (considered as task driven).

At this point we have shown that the framework is ready and can now be effectively used to evaluate more complex strategies involving the ASV. We have implemented our system to evaluate three case studies. All these case studies prove that our framework is versatile. Moreover, the use of an already deployed architecture is very encouraging to consider deploying the proposed approach and the selected strategies on real systems.

PERSPECTIVES

Buddha says, « I never see what has been done; I only see what remains to be done. »

In the context of our approach that mixes adaptive motions with predefined constraints for autonomous underwater vehicles, there are several directions for future work.

First, since the position information is essential in sensor networks, the use of AUVs to locate sensors once they are deployed according to target seeking strategies makes sense: our work could readily be applied to this problem.

Second, our target localisation use case can also be extended in other contexts. Other sensor types and use cases could also benefit from the proposed approach, such as mapping scenario (e.g.bathymetry, localization of algal blooms), or fish localization (we need here to predict their future location according to the actual fish location). For these purposes, one must adapt the target and sensor models to fit the tackled problem. For dynamic targets the ability of computing a predictive model of the environment online, for vehicles, can yield the definition of adaptive navigation strategies.

Finally, to improve the realism of the approach, one must more explicitly consider the position error of the AUVs, and exploit the rendezvous with the ASV to refine the position estimate. An important point to tackle is then the automatic generation of rendezvous points to plan the mission, and also their dynamic generation during mission execution. This can be interesting when rendezvous points are restricting the collection of the data from the target. The dynamic rendezvous generation has nevertheless to be restricted, to satisfy pre-defined constraints.

A

RÉSUMÉ DES TRAVAUX

A.1 INTRODUCTION

Nous nous intéressons à l'architecture de robots marins et sous-marins autonomes dans le cadre de missions nécessitant leur coopération. Cette coopération s'avère difficile du fait que la communication (acoustique) est de faible qualité et de faible portée. Afin d'illustrer notre travail, nous nous intéressons à un scénario de localisation d'une source d'eau chaude sous-marine. Pour cela, le véhicule sous marin parcourt des segments de droite et rejoint des points de rendez-vous (points de communication). Ces derniers sont importants car ils permettent la mise en œuvre d'une coopération entre les véhicules sous-marins. Au fur et à mesure du déplacement d'un véhicule, celui ci détecte (grâce à ses capteurs) sa distance à une zone pouvant contenir une source d'eau chaude. Afin de localiser une source, on doit permettre au véhicule de modifier sa trajectoire initiale, tout en s'assurant d'atteindre le point de rendez-vous. D'autre part, les rendez-vous permettent à chaque véhicule d'échanger ses données pour une coopération. Vu que le débit de communication accoustique est réduit, chaque véhicle doit extraire les informations utiles pour les communiquer. Nous présentons nos travaux effectués dans ce contexte, et une proposition d'architecture qui permet de trouver un compromis entre la modification de la trajectoire et l'atteinte de points de rendez-vous.

A.2 CONTRIBUTIONS

Les missions d'exploration sous-marines poussent les scientifiques à concevoir des véhicules sous marins de plus en plus performants, en termes de capacités technologique et aussi en termes d'autonomie. Pour des missions d'exploration, l'utilisation de plusieurs robots présente naturellement des avantages sur l'utilisation d'un seul [Burgard05], les robots coopératifs ayant le potentiel d'accomplir des tâches plus rapidement qu'un seul robot, notamment en partageant les informations qu'ils acquièrent.

Dans le contexte de notre travail, les robots utilisés sont des véhicules sous-marins, où la portée et les débit des communications accoustiques sont particulièrement contraints [Meyer06]. Ainsi un véhicule de surface ne peut communiquer efficacement avec un robot sous-marin que s'il est situé proche de la verticale de celui-ci. De telles contraintes imposent la définition de "rendez-vous" entre les engins, définis par des contraintes géométriques entre leurs positions et naturellement des contraintes temporelles (Voir figure A.1).

Nous nous intéressons à explorer une zone inconnue pour la localisation d'une cible le plus rapidement possible. Pour cela, on considère une flottille de véhicules hétérogènes, composée d'au moins un véhicule de surface (ASV pour Autonomous Surface Vehicles) et deux robots sous-marins (AUV pour Autonomous Underwater Vehicles). Un AUV parcourt une zone de recherche, atteint les points de rendez-vous pour la transmission des données collectées et localiser la cible en utilisant les données des autres AUVs. L'ASV joue le rôle d'un "hub" (nœud de communication) entre les véhicules sous-marins.

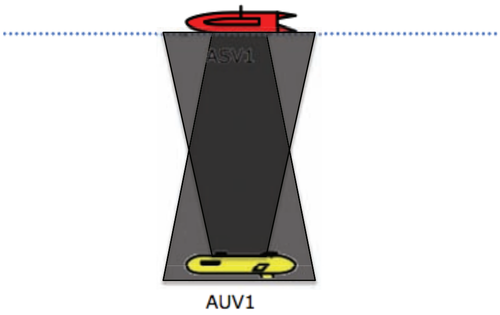


Figure A.1 – *Representation de la zone de communication entre AUV1 et ASV1. L'intersection des zones de communication entre les deux véhicules (de couleur noir) représente l'emplacement que l'un des véhicule peut avoir afin de communiquer, à ce moment, avec l'autre véhicule.*

Pour résoudre ce problème, nous proposons une approche pour la coopération entre les véhicules centrée sur la réalisation de rendez-vous, pendant lesquels ils échangent des informations dans le but de localiser la plus rapidement la cible. L'approche se base sur un algorithme qui permet un compromis entre l'exploration et l'atteinte de points de rendez-vous, en utilisant plusieurs fonctions qui mesurent l'importance des informations collectées, dans le but de répondre à ces deux questions: quand communiquer ces informations ? Et comment les exploiter ?

Le chapitre est composé de quatre sections: une discussion sur les travaux existants constitue la section 3. Dans la section 4, nous présentons le scénario utilisé ainsi que nos objectifs. Dans la section 5 nous détaillons l'architecture utilisée. Dans le section 6 nous présentons l'approche proposée. Nous détaillons les résultats obtenus dans diffÈrents cas dans le chapitre 7. Le chapitre se conclut par une discussion sur les travaux effecutés et perspectives.

A.3 ETAT DE L'ART

Les recherches en intelligence artificielle distribuée ont suscité de nombreuses contributions : plus de 900 publications selon Lynne E.Parker [Parker08], qui propose une classification des différents axes de recherches sur l'intelligence distribuée, basée sur le type d'interactions entre les différentes entités.

Dans le projet Martha [Alami98], les robots utilisent un paradigme de fusion de plan de déplacement entre les robots appelé “plan merging”. D'autres auteurs utilisent des approches “market-based” pour allouer les tâches aux différents robots [Stentz04]. Van Dyke Parunak et al. [Parunak01] utilisent les algorithmes basés sur les colonies de fourmis pour guider les véhicules vers une cible. La plupart de ces approches se focalisent sur des robots terrestres, où il y a peu de contraintes de communication entre les robots.

Plusieurs méthodes de parcours ont été élaborées pour des véhicules afin de localiser et d'atteindre une cible dans des terrains inconnus. Certains auteurs proposent l'utilisation des méthodes basées sur la perception du véhicule où ils utilisent le gain sur les données collectées pour diriger le véhicule. D'autres utilisent la notion d'ordonnancement des tâches, où pour accomplir une tâche on doit satisfaire toutes ses contraintes. Dans [Thompson08], les auteurs proposent un algorithme qui permet au robot d'explorer une zone selon les données acquises. Dudek et al. [Dudek07], utilisent un graphe pour l'exploration d'une zone, où le choix du prochain nœud à explorer se fait en fonction du nombre existants d'arêtes. Des chercheurs se sont intéressés à localiser une cible en utilisant une technique basée sur la quantité de phéromones [Parunak01]. Chaque véhicule dépose de la phéromone sur une cellule, la quantité de phéromone détectée dirigeant les véhicules vers la prochaine cellule à explorer. D'autres auteurs proposent un algorithme d'exploration où chaque véhicule calcule une utilité et un coût de déplacement vers une cible. La distance du véhicule au point cible constitue son coût. L'approche proposée dans [Burgard05] permet de répartir les points cibles à explorer aux différents véhicules.

Toutes ces approches sont considérées de type “data driven”, ce qui signifie que c'est la perception du véhicule qui guide l'exécution de la mission. Elles sont intéressantes si la mission est constituée d'une seule tâche. D'autres approches de type “Task driven” permettent d'ordonnancer des buts donnés et de générer d'autres tâches pour atteindre les buts. Plusieurs types de planificateurs permettent cet ordonnancement comme IxTeT [Laborie95, Lemai04], EUROPA [Frank03], etc. Des techniques d'allocation de tâches aux différents robots par des algorithmes de Market-based ont été mis en œuvre dans [Stentz04] et [Goldberg02]. L'approche “Task driven” prend en considération deux aspects :

1. La communication entre les véhicules est continue, ce qui n'est pas le cas dans notre contexte de travail.
2. Les tâches à accomplir sont pré-définies, ce qui n'est pas le cas de la détection d'une cible qui n'a pas d'emplacement connu.

Dans le cas où la mission contient plusieurs tâches à effectuer et que la stratégie de recherche est inconnue, alors les deux types d'approches “data driven” et le “task driven” doivent être exploitées en même temps. La première approche permet la gestion de plusieurs tâches et la seconde permet d'exploiter les données collectées pour diriger la recherche du véhicule.

A.4 CONTEXTE DE TRAVAIL

A.4.1 Scénario considéré

Dans notre scénario nous définissons trois véhicules : un ASV et deux AUVs. Chaque AUV a comme rôle de localiser la source d'eau chaude et d'atteindre les points de communication avec l'ASV. Les points de communication sont prédéfinis à l'avance et sont fixes. Ils permettent à l'AUV de réduire sa consommation d'énergie (éviter de remonter à la surface) mais aussi de transmettre les informations collectées à l'ASV. L'ASV est considéré comme un nœud de communication entre les différents véhicules. C'est lui qui détient toutes les informations collectées par les deux véhicules et les transmet d'un véhicule à un autre lors rendez-vous (RDVs).

A.4.2 Proposition

Nous nous intéressons à accomplir la mission d'une manière coopérative, en considérant l'incertitude sur l'environnement sous-marin, l'ignorance de la localisation de la cible et les restrictions des communications entre les véhicules sous-marins. Pour cela, nous proposons :

- Une autonomie décisionnelle de chaque véhicule : Chaque véhicule est capable d'accomplir sa tâche malgré l'incertitude du terrain, mais aussi de modifier sa trajectoire en fonction des informations collectées, pour une localisation rapide de la cible. Pour cela, nous utilisons l'architecture `T-ReX` [McGann07], détaillée dans la section suivante.

- Une coopération entre véhicules : Chaque véhicule doit être capable de fusionner ses données collectées avec les autres pour une rapide localisation de la cible. Pour que la communication ne dure pas longtemps, chaque véhicule doit réduire son débit d'envoi de données en n'envoyant que les données utiles.

A.5 L'ARCHITECTURE MONO-ROBOT

`T-ReX` est une architecture logicielle utilisée au sein du MBARI. Celle-ci permet d'affiner, de décomposer et d'ordonnancer des tâches (planifier), tout en prenant en considération l'exécution de celles-ci. `T-ReX` est composé de plusieurs réacteurs. Chaque réacteur contient une base de données (Database) où le modèle des tâches est défini, un planificateur (Planner) qui utilise un horizon de planification, un synchroniseur (Synchronizer) et un répartiteur (Dispatcher).

1. *Structures de données* : Les données que `T-ReX` manipule sont de type "timelines". Ces dernières représentent un ordonnancement de tâches instanciées. Chaque tâche instanciée est appelée "token". Pour ordonnancer des tokens un ensemble de contraintes peut être mis en place en utilisant NDDL (New Domain Definition Language). La représentation des tokens et de ses contraintes est appelée "modèle de la tâche".

2. *La base de données (Database)* : C'est une structure de données où les buts sont des tokens. Elle fournit au planificateur tous les buts à atteindre et le modèle des tâches à exécuter.

3. *Le planificateur (Planner)* : Celui-ci permet d'ordonnancer les tâches en utilisant sa base de données. Chaque réacteur possède un planificateur mais avec différents horizons de planification. Plus on descend dans la hiérarchie, plus l'horizon de planification se réduit. Par conséquent, chaque réacteur peut être délibératif (horizon de planification lointain) ou réactif (horizon de planification proche). Le planificateur utilisé est EUROPA [Frank03]. Celui-ci permet d'ordonnancer des tokens sur des timelines. Le système est composé de :

- Attributs (timelines) : A chaque attribut est associé un ensemble de valeurs qu'il peut prendre et qui sont décrites à travers des intervalles. Chaque attribut peut prendre une seule valeur à un instant donné. Ce qui correspond à des exclusions mutuelles entre intervalles de mêmes attributs.

- Un intervalle décrit un état ou une activité avec une extension temporelle. Cet intervalle est décrit par:

  $$holds(Att, td, tf, P)$$

  Att est un attribut, td est le temps début, tf est le temps fin et P est un prédicat de l'attribut.

  > **Exemple** : si on possède un attribut Location qui permet de situer un robot, et que cet attribut possède plusieurs prédicats appelé aussi état ou activité. Le prédicat $Going(x,y)$ signfie que le robot peut se déplacer d'un emplacement x vers un autre y. On peut définir un intervalle:
  >
  > $$holds(Location, 10, 20, Going(Hill, Lander))$$
  >
  > Cela signifie que le robot va se déplacer de la colline vers le sol et cela de la dixième unité de temps à la vingtième.

- Et un ensemble de règles de configurations décrivant comment les sous-systèmes agissent et interagissent entre eux, et qui sont des contraintes sur le domaine de planification (planning domain contraints).

  Chaque règle de configuration concerne un intervalle. Elle associe à un intervalle une disjonction de configurations Oi. Chaque Oi définit une conjonction d'autres intervalles Jik qui doivent exister dans un plan valide contenant l'intervalle I.

4. *Le répartiteur (Dispatcher)*. Il permet de gérer l'envoi des buts vers un autre réacteur.

5. *Le synchroniseur (Synchronizer)*. Il permet la coordination des observations effectuées par d'autres réacteurs. Ces observations permettent au réacteur de suivre l'exécution d'une tâche. Cette coordination est faite à chaque tick (unité temporelle utilisé, de l'ordre d'une seconde).

D'une manière plus formelle, `T-ReX` définit le monde (W) par un ensemble d'agents :

$$W = \{A_1 \ldots A_m\}$$

Les agents possèdent plusieurs *variables d'état* $S_w = \{s_1 \ldots s_n\}$ qui représentent les états que peut prendre le système (véhicule), des *réacteurs* : $R = \{r_1 \ldots r_n\}$, un *temps de vie* : $\mathcal{H} = [0, \Pi]$

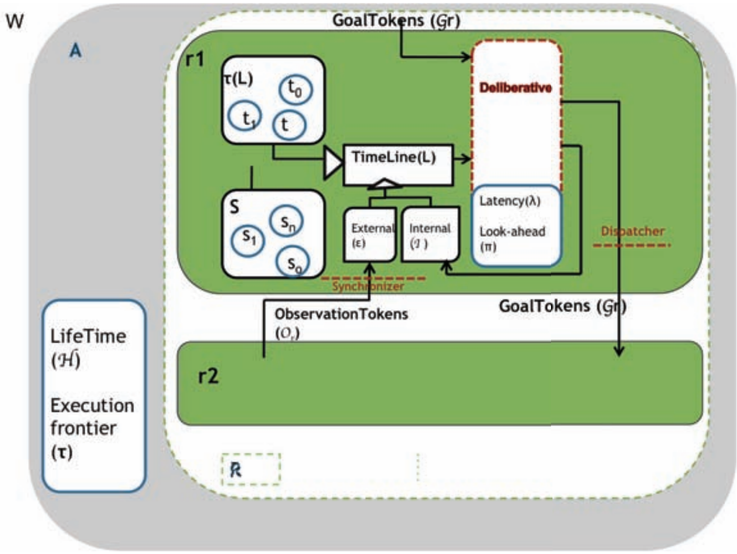

W
A
GoalTokens ($\mathcal{G}r$)
r1
τ(L)
t_0
t_1
t
TimeLine(L)
Deliberative
S
s_n
s_1
s_0
External
(ε)
Internal
($\mathcal{I}$)
Latency(λ)
Look-ahead
(π)
Dispatcher
Synchronizer
ObservationTokens
($\mathcal{O}_r$)
GoalTokens ($\mathcal{G}r$)
LifeTime
($\mathcal{H}$)
Execution
frontier
(τ)
r2
R

Figure A.2 – *Architecture de* `T-ReX`

où Π représente le temps total de la mission et une représentation du *temps actuel d'exécution* : τ.

Les réacteurs possèdent un temps de délibération λ_r, un “look-ahead” (jusqu'où le planificateur peut ordonner ses tâches) : $\pi < \Pi$, des timelines internes que le planificateur peut modifier : $I_r = \{I_1 \dots I_k\}$ ou externes que le planificateur ne peut qu'observer : $E_r = \{E_1 \dots E_l\}$, des buts : G_r et des observations : O_r.

En résumé, un réacteur est décrit de la manière suivante : $r_i = \{\lambda_r, \pi, I_r, E_r, G_r, O_r\}$. La Figure A.2 représente schématiquement les variables utilisées dans `T-ReX`.

Chaque réacteur peut poster des buts ou des observations à un autre réacteur d'une manière synchrone. C'est le “dispatcher” qui coordonne l'envoi des buts et le “synchronizer” qui coordonne la réception des observations. Cette architecture possède une forte notion de synchronisation entre les réacteurs, ce qui leur permet de percevoir, au fur et à mesure de l'exécution, toutes les modifications faites sur les timelines. Elle propose par ailleurs une notion de hiérarchie de délibération entre les réacteurs, ce qui permet à certain réacteurs d'être délibératif ou réactif.

A.6 PROPOSITION D'ARCHITECTURE

On veut permettre à chaque véhicule d'avoir une autonomie décisionnelle d'action, pouvant modifier sa trajectoire mais aussi coopérer avec d'autres véhicules. On utilise l'architecture `T-ReX` au sein de laquelle nous avons implémenté de nouveaux réacteurs illustrés dans la Figure A.3.

- Le MapReactor génère des buts selon la perception du véhicule (voir Figure A.4).

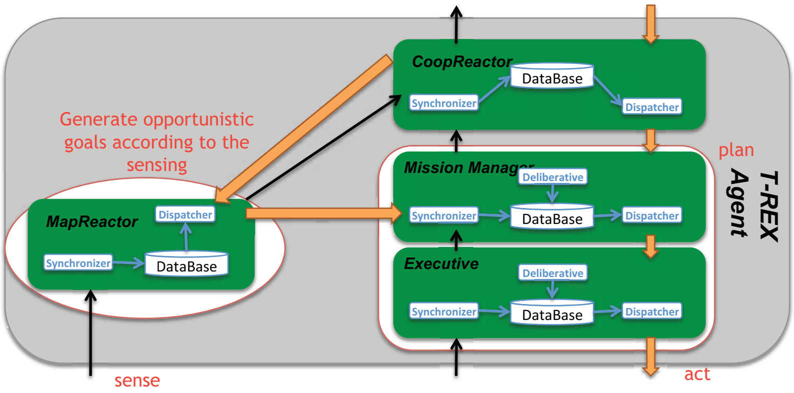

CoopReactor
DataBase
Synchronizer
Dispatcher
Generate opportunistic goals according to the sensing
plan
T-REX Agent
Mission Manager
Deliberative
Synchronizer
DataBase
Dispatcher
MapReactor
Dispatcher
Synchronizer
DataBase
Executive
Deliberative
Synchronizer
DataBase
Dispatcher
sense
act

Figure A.3 – *L'architecture de* `T-ReX` *modifiée*

1. La database contient une carte de toute la zone de mission, grille cartésienne appelée *Grid* de dimension $N \times M$. L'idée est de générer des buts pour changer la stratégie initiale de parcours quand l'information est pertinente. La grille est de deux dimensions :

   - Une valeur Booléenne : Quand le véhicule explore une cellule de la grille, la valeur Booléenne est mise à 1, dans le cas contraire cette valeur est de 0.

   - La deuxième valeur $P^k(x_{i,j})$ représente la probabilité perçue du véhicule d'être à côté de la cible à l'emplacement i, j où $i \in [0, N[$, $j \in [0, M[$. On suppose que cette probabilité est calculée selon un modèle. Ce dernier permet d'associer pour chaque perception une probabilité d'être proche de la cible. Par exemple dans [McGann08] les auteurs utilisent des HMM pour générer cette probabilité.

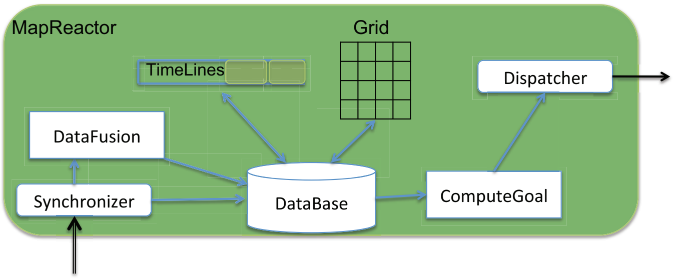

MapReactor
Grid
TimeLines
Dispatcher
DataFusion
Synchronizer
DataBase
ComputeGoal

Figure A.4 – *Description du MapReactor*

La database contient aussi les timelines et les contraintes associées aux timelines.

2. Fusion de données (DataFusion) : Cette fusion permet de mettre à jour les mesures de la grille. L'équation de fusion de données est calculée pour chaque cellule i, j de la grille comme suit :

$$P^k(x_{i,j}) = \frac{P(T^k|x_{i,j} = \text{vent})P^{k-1}(x_{i,j})}{P(T^k)} \tag{A.1}$$

$P^{k-1}(x_{i,j})$ est l'ancienne valeur de probabilité de la grille à l'emplacement i, j. La probabilité $P^k(x_{i,j})$ intègre les différentes mesures faite durant l'exploration ($P^k(x_{i,j}) = P(x_{i,j} = \text{vent}|T^0, \dots, T^k)$) qui est calculé d'une manière incrémentale en utilisant un paradigme bayésien avec une assomption markovienne. $T^0, \dots, T^k$ étant les différentes mesures de températures aux différents instants $0, \dots, k$.
Selon le type de senseur, la probabilité $P^k(x_{i,j} = vent)$ sera mise à jour :

(1) Si le senseur est un range-only la formule sera analogue à celle utilisée par Elfes et al. [Elfes89] pour la cartographie d'obstacles à l'aide de télémètres.

> La figure A.5 représente la mise à jour de la grille dans un environnement contenant une seule cible marine en utilisant un capteur de type "range-only". Trois mesures sont illustrées : La figure A.5.(a), décrit la grille obtenue pour une mesure, la figure A.5.(b) et la figure A.5.(c) montrent le résultat de deux et trois mesures du véhicule.

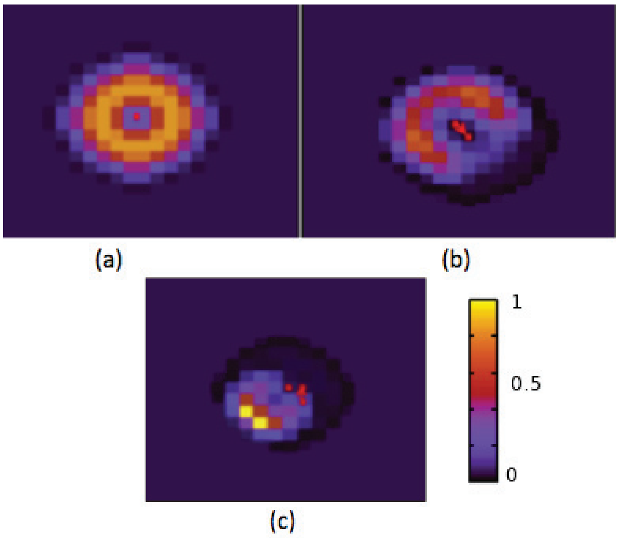
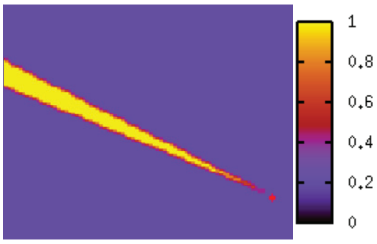


Figure A.5 – *Représente l'évolution de la probabilité de contenir la cible pour chaque cellule de la grille, en utilisant trois mesures sur une même profondeur. Les trois emplacements du véhicule sont représentés par des croix.*

(2) Dans un autre cas, si le véhicule utilise un senseur bearing-only, la probabilité $P^k(x_{i,j} = vent)$ est calculée par rapport à la distance de la direction. Où plus une cellule $x_{i,j}$ est loin de la direction calculé de la cible, plus sa probabilité est moindre (voir figure A.6a).



(a) (b)

Figure A.6 – *Illustration de la grille obtenue en utilisant un senseur bearing-only, pour un emplacement*

Quand la grille est mise à jour, le véhicule doit choisir la prochaine cellule à explorer. Ce choix de cellule est la stratégie d'exploration du véhicule.

3. Calcul de la prochaine cellule à explorer (ComputeGoal) : Plusieurs types de stratégie d'exploration peuvent être définis. Nous avons choisi de définir deux types de stratégies d'exploration : stratégie d'exploration non adaptative et une stratégie d'exploration adaptative.

   (a) Statégie d'exploration non adaptative : Cette stratégie d'exploration possède des buts prédéfinis, dans notre cas appelé waypoints, où le véhicule ne rallie que ses buts prédéfinis. Chaque waypoint est un emplacement dans la grille, où il lui est associé un intervalle temporel. Cet ordre de waypoints est implicitement donné par ces intervalles temporels.

   (b) Stratégie d'exploration adaptative : Cette stratégie a des buts prédéfinis mais avec la possibilité de rediriger le véhicule selon ses propres mesures. Deux seuils sont définis pour considérer la cible comme "détectée". Une valeur élevée dénotée par P_{loc} mène à bien confirmer la cible et l'autre valeur P_{conf} a une valeur inférieure à la précédente et qui permet de confirmer sa présence sans toute fois la localiser. Sur la base de ces deux seuils, nous avons définit deux types de stratégies d'exploration adaptatives comme suit :

      - G.I.D (Greedy Information Driven) : Le véhicule suit le plus proche hypothèse d'une source, jusqu'à ce que la probabilité exede celle définit sur P_{loc}. Cette stratégie se focalise sur la cible la plus proche jusqu'à ce qu'elle soit considéré comme localiser.

      - G.I.G (Global Information Gain driven) : Le véhicule se dirige sur la source la plus proche jusqu'à ce qu'il atteigne la probabilité P_{conf}. Cette stratégie d'exploration favorise la réduction d'actions au coût d'une moins bonne précision que la stratégie G.I.D.

- “Mission Manager” et l’“Executive” : Le Mission Manager prend en considération tous les buts de la mission de haut niveau et les buts envoyés par le MapReactor pour ensuite envoyer des sous buts à l’exécutif. Le Mission Manager ayant un horizon de planification et une latence plus grande que celle de l’exécutif, il est considéré comme le niveau délibératif. Cependant, l’exécutif est un composant plus réactif où son horizon et la latence du planificateur sont moins importantes. Ces deux composants permettent à l’architecture d’être réactive et en même temps délibérative.

- “CooperativeReactor” : Ce réacteur doit permettre de gérer l’envoi et la réception de messages d’autres véhicules. Celui-ci peut recevoir deux types de messages : des données qui seront fusionnées par le MapReactor et d’autres de type tâches qui seront utilisées par le Mission Manager. Ce qui est important au niveau de ce réacteur est de réduire le débit d’envoi de messages entre véhicules. De ce fait, chaque véhicule n’envoie que ses mesures perçues au fur et à mesure de l’exploration au lieu d’envoyer toute la grille. L’efficacité de la communication dépend des données transmises. Dans notre cas, nous prenons une bande passante de 100 bps [1]. Nous détaillons en premier lieu, la dimension des données échangées entre chaque véhicule, en second lieu nous allons calculer la complexité de la mise à jour de la grille par chaque véhicule après avoir collecté les données des autres véhicules.

  La compléxité des mises à jour de la grille après communication : A chaque point de communication, le véhicule envoi ses probabilités et les données reçues des autres véhicules. Pour éviter la redondance d’envoie de données, le véhicule envoie toutes ses observations entre l’instant de son dernier point de communication et cet actuel point de communication. L’ASV met à jour sa grille dédiée au véhicule. Le nombre d’opérations dépend du nombre de cellules explorées n_d, où le nombre d’opérations est calculé par $n_d * N * M$. Cela implique que la complexité de la mise à jour de la grille est de $O(N * M)$.

A.7 IMPLEMENTATIONS ET RÉSULTATS

Nous avons utilisé l’architecture inspirée de l’architecture `T-ReX`. Notre `CoT-ReX` architecture a été évaluée en utilisant plusieurs types de scenarios de détection de cibles.

A.7.1 Scenario I

Ce scénario utilise deux AUVs et un ASV en surface. L’objectif de ce scénario est d’évaluer les stratégies prédéfinies. L’ASV est utilisé comme un hub de communication entre les deux AUVs, et chaque AUV a une mission prédéfinie avec la possibilité de modifier sa trajectoire en fonction de ses mesures.

---

[1] bits per seconds

Comme première expérimentation, on définit pour chaque véhicule une stratégie d'exploration avec un espace d'exploration séparée, comme représenté dans la figure A.7b. La figure A.7a représente la diffusion de température de la cible, qui est une source d'eau chaude: plus la couleur du terrain est foncée plus la tempÈrature est basse.

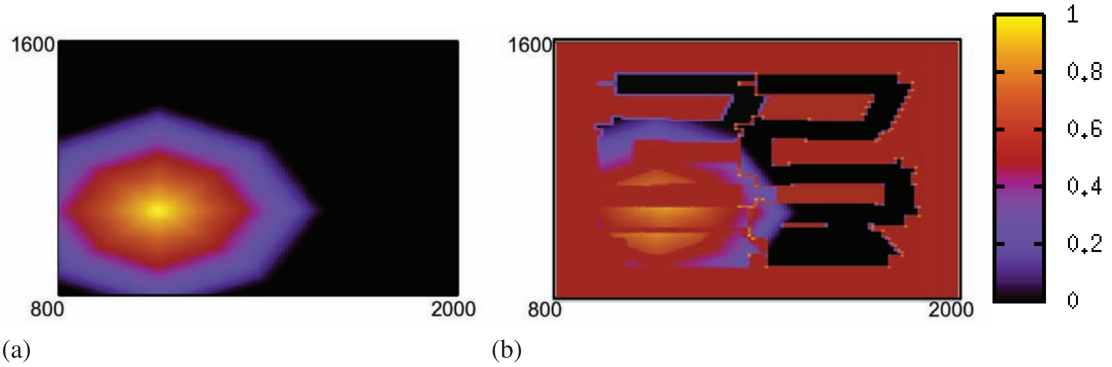


Figure A.7 – *Illustration du résultat de la carte construite par deux AUVs en utilisant une stratégie d'exploration à terrain distinct (a) représente le monde réel. (b) représente le résultat des stratégies d'exploration des deux véhicules.*

La séparation de terrain d'exploration fait que l'échange des cartes entre les deux véhicules n'a aucun impact sur l'exécution de la mission des AUVs. Afin de voir l'intérêt de la coopération, nous avons évalué une autre stratégie prédéfinie d'exploration, représentée figures A.8a et A.8b. Deux points de communications sont définis pour pouvoir faire communiquer les deux UAVs entre eux. Nous pouvons remarqué que la précision de construction du terrain est plus grande qu'à l'issue de la première stratégie: le fait d'utiliser des zones d'exploration communes permet d'augmenter la connaissance des véhicules, et donc d'améliorer le résultat de l'exploration.

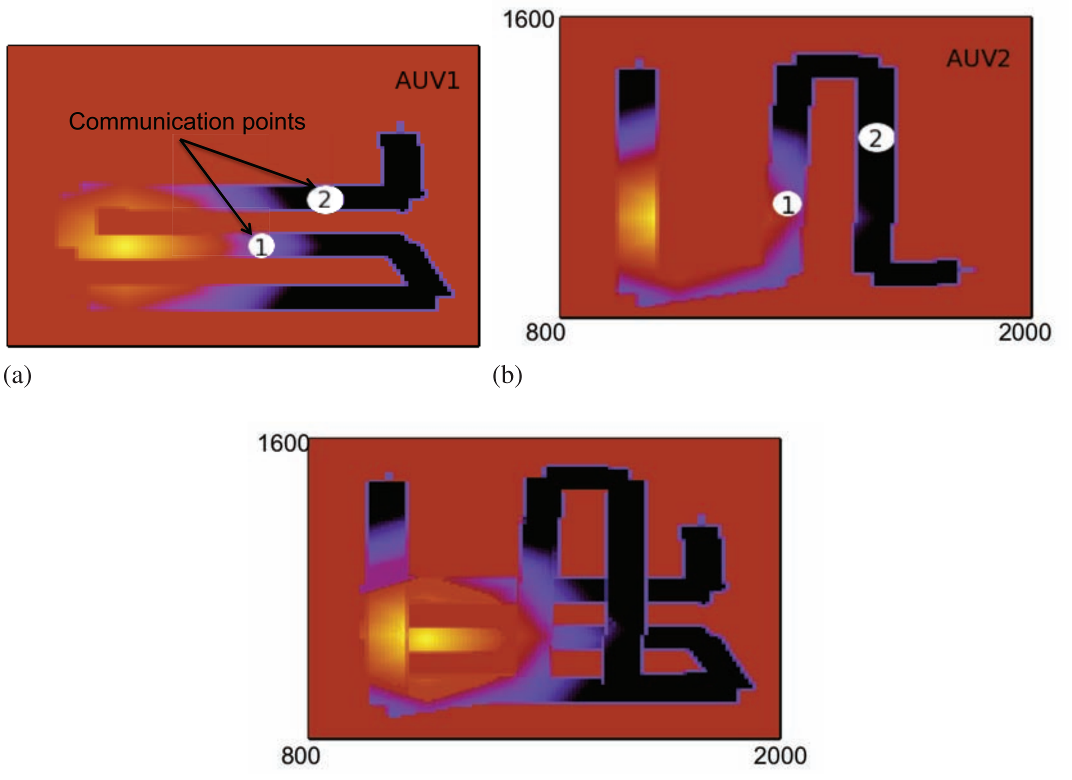
A.7.2 Scenario II

L'objectif de ce scénario est de localiser un nombre maximal de cibles, avec deux AUVs et un ASV. Trois types de stratégies d'explorations sont évaluées :

1. Stratégie d'exploration non adaptative : Cette stratégie d'exploration ne prend pas en considération les mesures des véhicules, qui exécutent simplement les t‚ches prédéfinies.
2. Stratégie d'exploration adaptative : Cette stratégie d'exploration prend en considération les mesures et génère des buts pour obtenir plus d'information sur les cibles. Deux types de stratégies d'exploration adaptatives sont évaluées : G.I.D et G.I.G expliqués dans la section A.6.

Chaque stratégie d'exploration à un espace d'exploration en commun. La figure A.9 représente une étude statistique sur le nombre de cibles trouvées et l'erreur entre la carte et le monde réel des trois types de stratégies d'exploration.

Nous remarquons que d'après les deux critères de selection (pourcentage de cibles trouvées et l'erreur de la map), la meilleure stratégie d'exploration est la G.I.G.



(a) (b)

(c)

Figure A.8 – *Illustration de la carte obtenue par les deux AUVs. Chaque AUV a une stratégie d'exploration commune avec l'autre AUV. (a) et (b) sont les stratégies prédéfinies d'exploration des deux véhicules AUV1 et AUV2. (c) montre la trajectoire rÈalisée par les deux véhicules ainsi que la carte finale obtenue.*

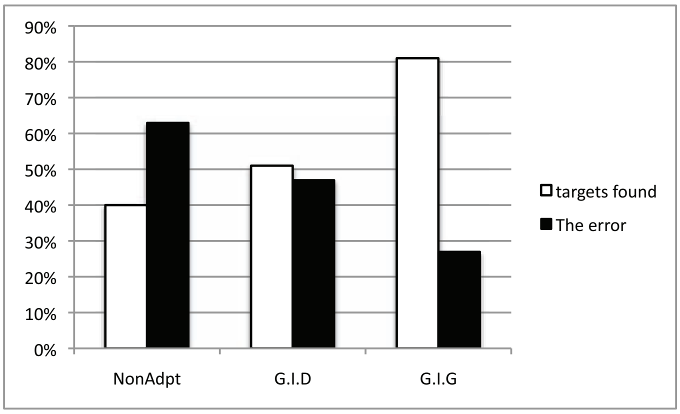


Figure A.9 – *Evaluation de trois différentes stratégies d'exploration en fonction du pourcentage de cibles trouvées et de l'erreur entre le monde réel et la carte construite.*

CONLUSION ET PERSPECTIVES

Dans ce document, nous avons mis en place une architecture pour la coopération multi robots dans un contexte marin et sous marin. Cette architecture est basée sur `T-ReX` où nous avons rajouté des composants pour l'étendre en une architecture coopérative. Cette architecture multi véhicules donne une autonomie d'action pour le véhicule tout en lui permettant d'accomplir sa mission. L'architecture proposée permet de prendre en considération la perception du véhicule (au fur et à mesure de son parcours) et de modifier sa stratégie initiale sans affecter l'atteinte de ses points de rendezvous. Nous avons validé notre approche dans différents scénarios, qui sont : (1) l'évaluation de stratégies prédéfinis (2) la localisation d'un nombre maximal de cibles marine avec une étude statistique. Ces scénarios nous ont permis de valider notre approche et de valider sa diversité d'action. Pour les travaux avenir, l'implémentation de cette architecture dans des réels véhicules sous marins validera nos simulations.

BIBLIOGRAPHY

[Akyildiz04] Ian F. Akyildiz, Dario Pompili, and Tommaso Melodia. Challenges for efficient communication in underwater acoustic sensor networks. *SIGBED Rev.*, 1(2):3–8, 2004. (Pages cited 13 et 38.)

[Alami98] R. Alami, R. Chatila, S. Fleury, M. Ghallab, and F. Ingrand. An architecture for autonomy. *International Journal of Robotics Research*, 17:315 – 337, 1998. (Pages cited 61 et 98.)

[Albert08] Erik Albert, Elise H Turner, and Roy M Turner. Appropriate commitment reactive planning. In *Workshop on A Reality Check for Planning and Scheduling Under Uncertainty*, 2008. (Page cited 60.)

[Alexander2009] Alexander Bahr. *Cooperative Localization for Autonomous Underwater Vehicles*. PhD thesis, Massachusetts Institute of Technology and the Woods Oceanographic Institution, 2009. (Page cited 13.)

[Allen83] J.F Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM 26*, 1983. (Page cited 63.)

[Baker95] E. T. Baker, C. R. German, and H. Elderfield. Hydrothermal plumes over spreading-center axes: Global distributions and geological inferences. In *In Humphris, S.*, 1995. (Page cited 29.)

[Bellingham94] J. G. Bellingham and J. J. Leonard. Task configuration with layered control. In *MIT Sea Grant College Program, Autonomous Underwater Vehicles Laboratory Report*, 1994. (Page cited 60.)

[Bernardini08] S. Bernardini and D. Smith. Translating pddl2.2. into a constraint-based variable/value language. In *ICAPS-08 Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*, 2008. (Page cited 64.)

[Bhatta05] P. Bhatta, E. Fiorelli, F. Lekien, N. E. Leonard, D. A. Paley, F. Zhang, R. Bachmayer, and R. Sepulchre. Coordination of an underwater glider fleet for adaptive sampling. In *International Workshop on Underwater Robotics*, 2005. (Page cited 21.)

[Biao08] Biao Wang, Yu Li, Haining Huang, and Chunhua Zhang. Target localization in underwater acoustic sensor networks. In *CISP '08: Proceedings of the 2008 Congress on Image and Signal Processing, Vol. 4*, pages 68–72, Washington, DC, USA, 2008. IEEE Computer Society. (Page cited 21.)

[Blidberg90] D.R. Blidberg, S. Chappel, J. Jalbert, R. Turner, G. Sedor, P. Eaton, A.M. Bradley, and B. Waldem. The eave auv program at the marine systems engeneering laboratory. In *In proceedings of the IARP 1nd Workshop on: Mobile Robots for Subsea Environments*, pages 23–26, 1990. (Page cited 61.)

[Botelho00] Silvia Botelho and Rachid Alami. Robots that cooperatively enhance their plans. In *In Proceedings of the 5th International Symposium on Distributed Autonomous Robotic Systems (DARS*. Springer Verlag, 2000. (Page cited 18.)

[Brignone09] L. Brignone, J. Alves, and J. Opderbecke. Grex sea trials: first experiences in multiple underwater vehicle coordination based on acoustic communication. In *Oceans*. IEEE, Bremen/Germany June 2009. (Page cited 21.)

[Brooks91] R. A. Brooks. Intelligence without representation. *Artificial Intelligence Journal*, 47:139,Äì–159, 1991. (Page cited 58.)

[Burgard02] W. Burgard, M. Moors, and F.Schneider. Collaborative exploration of unknown environments with teams of mobile robots. In M. Beetz, J. Hertzberg, M. Ghallab, and M.E. Pollack, editors, *Plan-Based Control of Robotic Agents*, volume 2466 of *Lecture Notes in Computer Science*. Springer Verlag, 2002. (Page cited 18.)

[Burgard05] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21:376–386, 2005. (Pages cited 9, 17, 18, 97 et 99.)

[BurgardJ05] W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–378, 2005. (Page cited 18.)

[Christopoulos05] V. Christopoulos and S.I. Roumeliotis. Adaptive sensing for instantaneous gas release parameter estimation. *In Proc. 2005 IEEE International Conference on Robotics and Automation*, pages 4461–4467, 2005. (Page cited 16.)

[Dearden07] R. W. Dearden, Z. A. Saigol, J. L. Wyatt, and B. J. Murton. Planning for auvs: Dealing with a continuous partially-observable environment. *ICAPS*, 2007. (Page cited 16.)

[Dechter91] R. Dechter, I. Meiri, and J Pearl. Temporal constraint network. *Artificial Intelligence Journal*, 49:61–95, 1991. (Page cited 62.)

[Dudek07] G. Dudek and D. Marinakis. Topological mapping with weak sensory data. In *AAAI National Conference on Artificial Intelligence*, 2007. (Page cited 99.)

[Dudenhoeffer00] D. D Dudenhoeffer and M.P. Jones. A formation behavior for large-scale micro-robot force deployment. In *Proceedings of the 2000 Winter Simulation Conference 2000, (WSC '00)*, 2000. (Page cited 58.)

[Edwards04] D. B. Edwards, T. A. Bean, D. L. Odell, and M. J. Anderson. A leader-follower algorithm for multiple auv formations. *IEEE/OES on Autonomous Underwater Vehicles*, pages 40–46, 2004. (Page cited 21.)

[Elfes89] Alberto Elfes. *Occupancy grids : a probabilistic framework for robot perception and navigation*. PhD thesis, Carnegie Mellon University, 1989. (Pages cited 34, 35 et 104.)

[Farrell03] J. A. Farrell, S. Pang W. Li, and R. Arrieta. Chemical plume tracing experimental results with a remus auv. *MTS/IEEE Oceans*, pages 962–968, 2003. (Page cited 16.)

[Ferri08] Gabriele Ferri, Michael V. Jakuba, and Dana R. Yoerger. A novel method for hydrothermal vents propspecting using an autonomous underwater robot. In *ICRA*. IEEE, 2008. (Pages cited 15 et 27.)

[Fiorelli06] E. Fiorelli, N.E. Leonard, D. Paley P. Bhatta, R. Bachmayer, and D.M. Fratantoni. Multi-auv control and adaptive sampling in monterey bay. *IEEE Journal of Oceanic Engineering*, pages 935–948, 2006. (Page cited 21.)

[Fleury97] Sara Fleury, Matthieu Herrb, and Raja Chatila. Genom: A tool for the specification and the implementation of operating modules in a distributed robot architecture. In *In International Conference on Intelligent Robots and Systems*, pages 842 – 848, 1997. (Page cited 61.)

[Frank03] Jeremy Frank and Ari Jónsson. Constraint-based attribute and interval planning. *Journal of Constraints, Special Issue on Constraints and Planning*, 8:339–364, 2003. (Pages cited 63, 68, 69, 99 et 100.)

[Gallien06] Matthieu Gallien and Félix Ingrand. Controlability and makespan issues with robot action planning and execution. In *IWPSS 2006 The 5th International Workshop on Planning and Scheduling For Space*, 2006. (Page cited 62.)

[German08] Christopher R. German, Dana R. Yoerger, Michael Jakuba, Timothy M. Shank, Charles H. Langmuir, and Ko-Ichi Nakamura. Hydrothermal exploration with the autonomous benthic explorer. *Deep Sea Research Part I: Oceanographic Research Papers*, 55(2), 2008. (Page cited 12.)

[Ghabchello09] R. Ghabchello, A. Aguiar, A. Pascoal, C. Silvestre, I. Kaminer, and Hespanha J. Coordinated path following in the presence of communication losses and time delays. *SIAM - Journal on Control and Optimization*, 48(1):234–265, 2009. (Page cited 21.)

[Ghallab94] M. Ghallab and H. Laruelle. Representation and control in ixtet, a temporal planner. In *In Proceedings of the International Conference on AI Planning Systems (AIPS)*, 1994. (Page cited 67.)

[Goldberg02] Dani Goldberg, Vincent Cicirello, M. Bernardine Dias, Reid Simmons, Stephen Smith, Trey Smith, and Anthony Stentz. A distributed layered architecture for mobile robot coordination: Application to space exploration. *In Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*, 2002. (Pages cited 18 et 99.)

[Haraksim09] Rudolf Haraksim, Lorenzo Brignone, and Jan Opderbecke. Multiple auv control in an operational context: a leader follower approach. In *IEEE Oceans, Bremen (Germany)*, June 2009. (Page cited 21.)

[Herman88] M. Herman and J. Albus. Overview of the multiple autonomous underwater vehicles (mauv) project. In *In IEEE International Conference on Robotics and Automation, Philadelphia, PA*, 1988. (Page cited 20.)

[Howard06] A Howard, L. E. Parker, and G. Sukhatme. Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment, and detection. *International Journal of Robotics Research*, 25(5–6):431–447, 2006. (Page cited 17.)

[Ingrand07] Félix Ingrand, Simon Lacroix, Solange Lemai-Chenevier, and Fredéric Py. Decisional autonomy of planetary rovers. *Journal of Field Robotics*, 2007. (Pages cited 57 et 61.)

[Ingrand96] R. Alami F. F. Ingrand, R. Chatila and F. Robert. Prs: A high level supervision and control language for autonomous mobile robots. In *IEEE ICRA*, 1996. (Page cited 62.)

[Ishida01] Hiroshi Ishida, Takamichi Nakamoto, Toyosaka Moriizumi, Timo Kikas, and Jiri Janata. Plume-tracking robots: A new application of chemical sensors. *The Biological Bulletin*, pages 222–226, 2001. (Page cited 16.)

[Jakuba08] M. Jakuba and D. Yoerger. Autonomous search for hydrothermal vent fields with occupancy grid maps. In *In Proc. of ACRA'08*, 2008. (Page cited 16.)

[Kim08] Sang-Chul Kim, Kee-Hyun Shin, Chong-Woo Woo, Yun-Shick Eom, and Jae-Min Lee. Performance analysis of entropy-based multi-robot cooperative systems in a manet. *International Journal of Control, Automation, and Systems*, 6(5):722–730, 2008. (Page cited 18.)

[Kitano99] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjoh, and S. Shimada. Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. *IEEE SMC*, pages 739–743, 1999. (Page cited 18.)

[Konolige03] K. Konolige, D. Fox, B. Limketkai, J. Ko, and B. Stewart. Map merging for distributed robot navigation. In *International Conference on Intelligent Robots and Systems, Las Vegas, USA*, 2003. (Page cited 17.)

[Konolige06] Kurt Konolige, Fox Dieter, Ortiz Charlie, Agno Andrew, Eriksen Michael, Limketkai Benson, Ko Jonathan, Morisset Benoit, Schulz Dirk, Stewart Benjamin, and Vincent Regis. Centibots : Very large scale distributed robotic teams. *Journal Experimental Robotics IX*, pages 131–140, 2006. (Page cited 18.)

[Kube93] Ronald Kube and Hong Zhang. Collective robotics: From social insects to robots. *Adaptive Behavior*, 2:189–218, 1993. (Page cited 17.)

[Laborie95] P. Laborie and M. Ghallab. Ixtet: an integrated approach for plan generation and scheduling inria/ieee symposium on. *Emerging Technologies and Factory Automation*, 1:485–495, 1995. (Page cited 99.)

[Lemai-Chenevier04] Solange Lemai-Chenevier. *IxTeT-eXEC: Planning, plan repair and execution control with time and resource management.* PhD thesis, l'Institut National Polytechnique de Toulouse, 2004. (Page cited 67.)

[Lemai04] Solange Lemai and Félix Ingrand. Interleaving temporal planning and execution : Ixtet-exec. *14ème Congrès Francophone AFRIF-AFIA de Reconnaissance des Formes et Intelligence Artificielle*, 2004. (Page cited 99.)

[Liu02] Juan Liu, Reich, and Feng Zhao. Collaborative in-network processing for target tracking. *J. on Applied Signal Processing*, 4:378–391, 2003. (Page cited 21.)

[Lonsdale77] P. Lonsdale. Clustering of suspension-feeding macrobenthos near abyssal hydrother-mal vents at oceanic spreading centers. *Deep-Sea Res.*, 9(24):857–863, 1977. (Page cited 27.)

[Low08] Kian Hsiang Low, John M. Dolan, and Pradeep Khosla. Adaptive multi-robot wide-area exploration and mapping. *AAMAS '08: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, pages 23–30, 2008. (Page cited 22.)

[Low09] Kian Hsiang Low, John Dolan, and Pradeep Khosla. Information-theoretic approach to efficient adaptive path planning for mobile robotic environmental sensing. In *Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS-09)*, September 2009. (Pages cited 22 et 34.)

[Manual] IFREMER. *User's Manual, MIMOSA (MIssion Managment fOr Subsea Autonomous vehicles).* (Page cited 90.)

[Masao94] Kubo Masao and Kakazu Yukinori. Learning coordinated motions in a competition for food between ant colonies. In *SAB94: Proceedings of the third international conference on Simulation of adaptive behavior : from animals to animats 3*, pages 487–492, Cambridge, MA, USA, 1994. MIT Press. (Page cited 17.)

[Maurizio07] Maurizio Porfiri, D. Gray Roberson, and Daniel J. Stilwell. Tracking and formation control of multiple autonomous agents: A two-level consensus approach. *Automatica*, 43(8):1318–1328, 2007. (Page cited 21.)

[McAllister91] David McAllister and David Rosenblitt. Systematic nonlinear planning. In *In Proceedings of AAAI*, pages 634,Äì–639, 1991. (Page cited 65.)

[McGann07] C. McGann, F. Py, K. Rajan, H. Thomas, R et Henthorn, and R et McEwen. T-rex: A deliberative system for auv control. *ICAPS*, 2007. (Pages cited 62, 63, 93 et 100.)

[McGann08] C. McGann, F. Py, K. Rajan, H. Thomas, R. Henthorn, and R. McEwen. Preliminary results for model-based adaptive control of an autonomous underwater vehicle. In *11th Int'l Symp. on Experimental and Robotics (ISER)*, July 2008. (Pages cited 69 et 103.)

[Meeussen10] Wim Meeussen, Melonee Wise, Stuart Glaser, Sachin Chitta, Conor McGann, Patrick Mihelich, Eitan Marder-Eppstein, Marius Muja, Victor Eruhimov, Tully Foote, John Hsu, Radu Bogdan Rusu, Bhaskara Marthi, Gary Bradski, Kurt Konolige, Brian P. Gerkey, and Eric Berger. Autonomous door opening and plugging in with a personal robot. In *ICRA*, 2010. (Page cited 71.)

[Meier06] D. Meier, C. Stachniss, and W. Burgard. Cooperative exploration with multiple robots using low bandwidth communication. *In J. Beyerer , F. Puente Le'on, K.-D. Sommer (Eds.), Informationsfusion in der Mess- und Sensortechnik*, page pp. 145157, 2006. (Page cited 18.)

[Meliou07] Meliou Alexandra, Krause Andreas, Guestrin Carlos, and Hellerstein Joseph M. Nonmyopic informative path planning in spatio-temporal models. In *AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence*, pages 602–607. AAAI Press, 2007. (Page cited 21.)

[Meyer06] Matthias Meyer and Jean-Pierre Hermand. Backpropagation techniques in ocean acoustic inversion: time reversal, retrogation and adjoint model – A review. *Acoustic Sensing Techniques for the Shallow Water Environment*, 2006. (Pages cited 3, 38 et 97.)

[Michel03] J. L. Michel, M. Klages, F. J. A. S. Barriga, Y. Fouquet, M. Sibuet, P. M. Sarradin, P. Siméoni, and J. F. Drogou. Victor 6000 : design, utilisation and first improvements. *ISOPE*, 2003. (Page cited 9.)

[Moorehead01] S. Moorehead, R. Simmons, and W. Whittaker. A multiple information source planner for autonomous planetary exploration. In *ISAIRAS*, 2001. (Page cited 18.)

[Muscettola02] Nicola Muscettola, Gregory A. Dorais, Chuck Fry, Richard Levinson, and Christian Plaunt. Idea: Planning at the core of autonomous reactive agents. *3rd International NASA Workshop on Planning and Scheduling for Space*, 2002. (Pages cited 60, 63 et 67.)

[Ogren04] P. Ogren, E. Fiorelli, and N.E. Leonard. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *IEEE Transactions on Automatic Control*, 49(8):1292–1302, Aug. 2004. (Page cited 18.)

[Pang06] S. Pang and J. A. Farrell. Chemical plume source localization. *IEEE Systems, Man, and Cybernetics - Part B*, pages 1068–1080, 2006. (Page cited 16.)

[Parker08] L. E. Parker. Distributed intelligence: Overview of the field and its application in multi-robot systems. *Journal of Physical Agents, special issue on multi-robot systems*, 2:5–14, 2008. (Pages cited 17 et 98.)

[Parker98] Lynne E. Parker. Alliance: An architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, 14:220–240, 1998. (Page cited 19.)

[Parunak01] Van Dyke Parunak and Sven Brueckner. Entropy and self-organization in multi-agent systems. *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, 2001. (Pages cited 17, 98 et 99.)

[Pascoal95] A. Pascoal, C. Silvester, P. Oliviera, D. Fryxell, G. Sedor, and P.Eaton. Undersea robotics research at ist: The auv marius programme. In *In proceedings of the International Program Development in Undersea Robotics and Intelligent Control (URIC): A joint U.S./Portugal Workshop*, pages 111–118, 1995. (Page cited 61.)

[Patron08] Pedro Patron, Emilio Miguelanez, Yvan R. Petillot, and David M. Lane. Fault tolerant adaptive mission planning with semantic knowledge representation for autonomous underwater vehicles. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems Acropolis Convention Center*, 2008. (Page cited 16.)

[Philibert06] M.T. Stacey, E.A. Cowen, T.M. Powell, S.G. Monismith, J.R. Koseff, and E. Dobbins. Adolf fick and diffusion equations. *Defect and Diffusion Forum*, 249:1–6, 2006. (Page cited 28.)

[Popa04] Dan.O Popa, Sanderson Arthur C., Komerska Rick J., Mupparapu Sai S., Blidberg D. Richard, and Chappel Steven G. Adaptive sampling algorithms for multiple autonomous underwater vehicles. *Autonomous Underwater Vehicles IEEE/OES*, pages 108–118, 2004. (Page cited 21.)

[Rahimi04] M. Rahimi, R. Pon, W. J. Kaiser, G. S. Sukhatme, D. Estrin, and M. Srivastava. Adaptive sampling for environmental robotics. In *IEEE International Conference on Robotics and Automation*, pages 3536–3544, 2004. (Pages cited 3 et 20.)

[Rigaud94] Vincent Rigaud, E.Ke Rest, L.Marce, A.Peuch, and M.Perrier. Vortex, versatile and open subsea robot for technical experiment : Prototyping software architecture for the next auv and rov generation. *International Ofshore and Polar Engineering Conference.Osaka, Japan, April 10-15*, 1994. (Page cited 10.)

[Rock95] S.M. Rock, H.H. Wang, and M.J. Lee. Task-directed precision control of the mbari/stanford otter auv. In *In proceedings of the International Program Development in Undersea Robotics and Intelligent Control (URIC): A joint U.S./Portugal Workshop*, pages 131–138, 1995. (Page cited 61.)

[Rosenblatt95] Julio Rosenblatt. Damn: A distributed architecture for mobile navigation - thesis summary. In *Journal of Experimental and Theoretical Artificial Intelligence*, pages 339–360. AAAI Press, 1995. (Pages cited 58 et 59.)

[S.Moorehead01] Stewart J. Moorehead, Reid G. Simmons, and William Whittaker. Autonomous exploration using multiple sources of information. In *ICRA*, pages 3098–3103, 2001. (Page cited 34.)

[Sinha09] A. Sinha, A.Tsourdos, and B.A. White. Monitoring the dispersion of a contaminant cloud in an urban region by a swarm of uav sensors. *Proceedings of IFAC Workshop on Networked Robotics*, 2009. (Page cited 16.)

[Sotelo03] Miguel Ángel Sotelo, Luis Miguel Bergasa, Ramón Flores, Manuel Ocaña, Marie-Hélène Doussin, Luis Magdalena, Joerg Kalwa, Anders L. Madsen, Michel Perrier, and Damien Roland Pietro Corigliano. Advocate *II*: Advanced on-board diagnosis and control of autonomous systems *II*. In *EUROCAST*, pages 302–313, 2003. (Page cited 58.)

[Sotzing08] Chris C. Sotzing and David M. Lane. Improving the coordination efficiency of multiple auv operations using prediction of intent. In *Proceedings of the 3rd SEAS DTC Technical Conference*, 2008. (Page cited 21.)

[Stentz04] Anthony Stentz, M. Bernardine Dias, Robert Zlot, and Nidhi Kalra. Market-based approaches for coordination of multi-robot teams at different granularities of interaction. *Proceedings of the ANS 10th International Conference on Robotics and Remote Systems for Hazardous Environments*, pages 727–753, 2004. (Pages cited 18, 98 et 99.)

[Stenz04] Anthony Stentz, Bernardine Dias, Robert Michael Zlot, , and Nidhi Kalra. Market-based approaches for coordination of multi-robot teams at different granularities of interaction. In *Proceedings of the ANS 10th International Conference on Robotics and Remote Systems for Hazardous Environments*, 2004. (Page cited 3.)

[Stone99] Peter Stone and Manuela Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. In *Artificial Intelligence*, volume 110, pages 241–273, 1999. (Page cited 19.)

[Stracey00] M.T. Stacey, E.A. Cowen, T.M. Powell, S.G. Monismith, J.R. Koseff, and E. Dobbins. Plume dispersion in a stratified, near-coastal flow: measurements and modeling. *Continental Shelf Research*, 20:637–663, 2000. (Page cited 29.)

[Stroupe05] Ashley W. Stroupe and Tucker R. Balch. Value-based action selection for observation with robot teams using probabilistic techniques. *Robotics and Autonomous Systems*, 50(2-3):85–97, 2005. (Page cited 19.)

[Tambe02] Milind Tambe and Stacy Marsella. Task allocation in the robocup rescue simulation domain: A short note. *Book RoboCup 2001: Robot Soccer World Cup V*, 2377:1–22, 2002. (Page cited 18.)

[Thompson08] David R. Thompson, Trey Smith, and David Wettergreen. Information-optimal selective data return for autonomous rover traverse science and survey. *IEEE International Conference on Robotics and Automation*, pages 19–23, 2008. (Page cited 99.)

[Turner95] Roy M. Turner. Intelligent control of autonomous underwater vehicles: The orca project. In *In Proceedings of the 1995 IEEE International Conference on Systems, Man, and Cybernetics*, 1995. (Page cited 59.)

[TurnerJ01] Roy M. Turner. A two-level, protocol-based approach to controlling autonomous oceanographic sampling networks. *IEEE JOURNAL OF OCEANIC ENGINEERING*, 26(4), 2001. (Page cited 20.)

[Valavanis97] Valavanis K. P., D. Gracanin, M. Matijasevic, R. Kolluru, and G. A. Demetriou. Control architectures for autonomous underwater vehicles. In *IEEE Control Systems Magazine*, volume 17, page 48,Äì64, 1997. (Page cited 58.)

[Vangriesheim92] A. Vangriesheim, J.P. Gouillou, and L. Prieur. A deep-ocean nephelometer to detect bottom and intermediate nepheloid layers. *Deep Sea Research Part A. Oceanographic Research Papers*, 39:1403–1416, 1992. (Page cited 29.)

[Veirs99] S. Veirs, R.E. McDuff, M.D. Lilley, and J.R. Delaney. Locating hydrothermal vents by detecting and modeling buoyant. *advected plumes. J. Geophys. Res.*, 104:29239 – 29247, 1999. (Page cited 16.)

[Veloso98] Manuela Veloso, Peter Stone, and Kwun Han. The cmunited-97 robotic soccer team: Perception and multiagent control. In *In Proceedings of the Second International Conference on Autonomous Agents*, pages 78–85. ACM Press, 1998. (Page cited 18.)

[Woolridge01] Michael Woolridge and Michael J. Wooldridge. *Introduction to Multiagent Systems*. John Wiley & Sons, Inc., New York, NY, USA, 2001. (Page cited 17.)

[Yoerger94] D.R. Yoerger, A.M. Bradley, and B. Waldem. System testing of autonomous benthic explorer. In *In proceedings of the IARP 2nd Workshop on: Mobile Robots for Subsea Environments*, pages 159–170, 1994. (Page cited 60.)

[You05] Guang Xin You, Yong jie Pang, and Da peng Jiang. Market based framework for multiple auvs cooperation. *Journal of Marine Science and Application*, pages 7–12, 2005. (Page cited 20.)

[Yuh96] J. Yuh. Autonomous underwater robot design. In *1996 World Automation Congress, Tutorial2: Underwater Robotic Systems ASL96-01*, 1996. (Page cited 60.)

[Zhang08] B. Zhang and G. S. Sukhatme. Adaptive sampling with multiple mobile robots. In *IEEE International Conference on Robotics and Automation*, 2008. (Page cited 21.)