

# **Intelligent Adaptive Underwater Sensor Networks**

*Veronika Yordanova*

A dissertation submitted in partial fulfillment

of the requirements for the degree of

**Doctor of Philosophy**

of

**University College London.**

UCL DEPARTMENT OF ELECTRONIC AND ELECTRICAL ENGINEERING

UCL DEPARTMENT OF SECURITY AND CRIME SCIENCE

University College London

February 2, 2018

## **Declaration**

I, Veronika Yordanova, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

## Abstract

Autonomous Underwater Vehicle (AUV) technology has reached a sufficient maturity level to be considered a suitable alternative to conventional Mine Countermeasures (MCM). Advantages of using a network of AUVs include time and cost efficiency, no personnel in the minefield, and better data collection.

A major limitation for underwater robotic networks is the poor communication channel. Currently, acoustics provides the only means to send messages beyond a few metres in shallow water, however the bandwidth and data rate are low, and there are disturbances, such as multipath and variable channel delays, making the communication non-reliable.

The solution this thesis proposes using a network of AUVs for MCM is the Synchronous Rendezvous (SR) method — dynamically scheduling meeting points during the mission so the vehicles can share data and adapt their future actions according to the newly acquired information. Bringing the vehicles together provides a robust way of exchanging messages, as well as means for regular system monitoring by an operator.

The gains and losses of the SR approach are evaluated against a benchmark scenario of vehicles having their tasks fixed. The numerical simulation results show the advantage of the SR method in handling emerging workload by adaptively retasking vehicles.

The SR method is then further extended into a non-myopic setting, where the vehicles can make a decision taking into account how the future goals will change, given the available resource and estimation of expected workload. Simulation results show that the SR setting provides a way to tackle the high computational complexity load, common for non-myopic solutions.

Validation of the SR method is based on trial data and experiments performed using a robotics framework, MOOS-IvP.

This thesis develops and evaluates the Synchronous Rendezvous (SR) method, a mission planning approach for underwater robotic cooperation in communication and resource constraint environment.

## Acknowledgements

First of all, I would like to thank everyone that has read or is reading this thesis. And everyone that asked a question, or gave a suggestion, or provided advice, or destroyed me with a critique, or pretended to be interested.

Most of all, I would like to thank my supervisors, Hugh Griffiths and Steve Hailes. You did all of the above things, and more. I really appreciate it you always found the time for me, even on very short notice. Your guidance and encouragement shaped the work in this thesis and prevented a lot of dead ends.

This work would not be possible without the funding from Atlas Elektronik UK, EPSRC, Robocademy and SeCRET. This PhD project required a lot of management effort. Thank you Kostas Siantidis and Richard Brind, John Wickenden and James Kelly.

I was in many good discussions and spirits with my friends in the Luncheon group, the Radar group, UCL and the Robocademy. Thanks everyone, and especially Alex G., Alistair L., Amin A., Erin D., George S., Gwyn E., Horhe S., Matias V., Michaela R., Nilufer T., Tom R., Samy N.

I started as a student in UCL, but was very lucky to also be part of the Robocademy, Atlas Elektronik GmbH and the National Oceanography Centre during my studies. Thanks for the advice and help Aneta N., Andrea M., Catherine H., Hans G., Joerg K.

Lastly, thanks to all of the people from outside academia that distracted me from the constant and nagging PhD thoughts. I have met so many great people for the past years, thanks London people, thanks Bremen people, thanks old Bulgarian friends, thanks new Southampton friends. Thanks to all the rugby people in Kilburn, Bulgarian women national team, Union60, Saints, and everyone I've tackled.

Thanks to my family. And thanks Hughie M.

# Contents

<b>Abbreviations</b>	<b>14</b>
<b>1 Introduction</b>	<b>20</b>
1.1 Motivation . . . . .	21
1.2 Physics of the environment - acoustic propagation . . . . .	24
1.2.1 Navigation . . . . .	27
1.2.2 Imaging sensors . . . . .	28
1.3 Naval mines and countermeasures . . . . .	29
1.3.1 Mine Classification . . . . .	29
1.3.2 Countermeasures . . . . .	33
1.3.3 Mine hunting phases . . . . .	35
1.4 Problem statement and key contributions . . . . .	37
1.5 List of publications . . . . .	38
<b>2 Research Context</b>	<b>39</b>
2.1 Autonomy and Artificial Intelligence . . . . .	39
2.1.1 Definitions . . . . .	40
2.1.2 Classical Planning . . . . .	41
2.1.3 Non-Classical Planning and Probability . . . . .	42
2.2 Classification of Methods . . . . .	43
2.2.1 Reactive vs Deliberative Control Architectures . . . . .	44
2.2.2 Model-driven vs Data-driven Mission Planning . . . . .	45
2.2.3 Planning and Learning . . . . .	47
2.3 Marine Autonomy Applications and Development . . . . .	49
2.3.1 Terrestrial, Space and Marine Environment . . . . .	49
2.3.2 Applications and Advances for Multi-Vehicle Systems . . . . .	50

2.3.3	AUV Development in Defence and Surveillance . . . . .	57
2.3.4	Summary of Common Assumptions . . . . .	59
2.4	Synchronous Rendezvous . . . . .	60
<b>3</b>	<b>Synchronous Rendezvous Method</b>	<b>63</b>
3.1	Synchronous Rendezvous Method Scenario . . . . .	64
3.1.1	Synchronous Rendezvous Loss in Mine Countermeasure Scenario . . . . .	66
3.1.2	The Baseline Mission . . . . .	69
3.2	Synchronous Rendezvous Scheduling Algorithm . . . . .	70
3.2.1	Synchronous Rendezvous Time Interval . . . . .	70
3.2.2	Synchronous Rendezvous Location . . . . .	71
3.2.3	Synchronous Rendezvous Algorithm . . . . .	71
3.2.4	Target Number Distribution Model . . . . .	74
3.3	Results and Analysis . . . . .	75
3.3.1	Simulation Results . . . . .	75
3.3.2	Vehicle Task Allocation . . . . .	77
3.3.3	Search Resource . . . . .	78
3.3.4	Reacquire-Identify (RI) Resource . . . . .	79
3.3.5	Time and Search Area Loss as a Function of Number of Targets . . . . .	81
3.3.6	Evaluation — Synchronous Rendezvous (SR) or Benchmark . . . . .	83
3.4	Summary . . . . .	89
<b>4</b>	<b>Synchronous Rendezvous Planning with Markov Decision Process (SR-MDP)</b>	<b>91</b>
4.1	Markov Decision Process Definition . . . . .	91
4.1.1	Markov Decision Process Elements . . . . .	92
4.2	Markov Decision Process Formulation for MCM with SR . . . . .	93
4.2.1	SR-MDP Goal . . . . .	94
4.3	Modelling Multiple Vehicle MCM Planning with MDP . . . . .	95
4.3.1	State Space Model and Discretisation . . . . .	95
4.3.2	Action Space Model and Discretisation . . . . .	98
4.3.3	World Model Approximation . . . . .	100
4.3.4	Reward Function Model . . . . .	104
4.3.5	Horizon Space . . . . .	106
4.4	MDP Solution Method . . . . .	106
4.4.1	Value Iteration . . . . .	107

4.4.2	Results . . . . .	108
4.4.3	Discussion . . . . .	108
4.4.4	Input Variance Analysis . . . . .	113
4.4.5	Complexity and Convergence . . . . .	116
4.4.6	Contributions . . . . .	117
4.5	Summary . . . . .	118
<b>5</b>	<b>Simulation and Experimental Validation</b>	<b>120</b>
5.1	Simulation Environment . . . . .	120
5.1.1	MOOS Middleware . . . . .	121
5.1.2	IvP Helm Application . . . . .	122
5.2	Synchronous Rendezvous (SR) implementation in MOOS-IvP . . . . .	123
5.2.1	Simulation results . . . . .	127
5.3	Experimental data . . . . .	134
5.3.1	Kapitas Trials . . . . .	135
5.3.2	Lawnmower Path Experiment . . . . .	136
5.3.3	Target Identification Manoeuvre Experiment . . . . .	138
5.3.4	SeaCat Trials . . . . .	141
5.4	Summary . . . . .	143
<b>6</b>	<b>Conclusion</b>	<b>147</b>
6.1	Summary . . . . .	147
6.2	Key Contributions . . . . .	149
6.3	Future Work . . . . .	150
	References . . . . .	152

# List of Figures

1.1	AUV autonomy capabilities . . . . .	21
1.2	AUV designs: top left: SeaCat, bottom left: Remus, top right: ecoSub, middle right: Girona 500, bottom right: ALR . . . . .	22
1.3	Recent damage caused by mines to United States warships . . . . .	23
1.4	Comparison between acoustic and RF attenuation in seawater . . . . .	25
1.5	Acoustic frequency limitation based on 1/Attenuation-Noise (AN)for various distances .	26
1.6	Propagation of noise sources in water . . . . .	26
1.7	Mine classification for different warfare regions . . . . .	29
1.8	Bottom mine type Manta, partially buried due to ocean activity over time . . . . .	30
1.9	Moored mine being neutralised by divers . . . . .	31
1.10	Floating mine . . . . .	31
1.11	Mine layer of the Iraqi Navy . . . . .	33
1.12	An MH-53 helicopter, during a mine sweeping exercises in the Persian Gulf in 2014 - equipment that cuts the chains off sea mines is towed and the mines are revealed by floating to the surface . . . . .	34
1.13	A Seafox mine neutralisation ROV launched from the German minehunter FGS Homburg	35
1.14	A sea lion from the Navy Marine Mammal Program salutes its handler before diving as part of an International Mine Countermeasures Exercise in the Middle East in 2014 . .	36
2.1	Markov Decision Process (MDP) framework . . . . .	43
2.2	Autonomous robots designs: top left: Sonobot surface vehicle, top right: Delivery car, bottom left: Mars Curiosity rover, bottom right: Raven drone . . . . .	49

3.1	Synchronous Rendezvous (SR) approach: Left - search phase, Right - Reacquire-Identify (RI) phase. The area that requires clearing is a rectangle with dimensions $x$ and $y$ . The ‘start’ point defines the launching site of the AUVs. On the left graph, three vehicles are moving in a lawnmower pattern. Each node has a separate search area. The vehicles meet at the SR to share status and data gathered during the search phase. On the right, is an example of decision making and retasking of the vehicles. At the SR, they share locations of potential targets that require further inspection. . . . .	65
3.2	Loss calculation for two vehicles (lines) and three vehicles (dots): different speed values and width of search area ( $x$ ); fixed RP time (1 hour), and fixed length of the y dimension of searched space (5000 m). . . . .	67
3.3	Loss of time resource (y axis) vs total time between SR ( <i>x axis</i> ) – calculation based on Equation 3.1 (parameters used: $n = 3$ ; $x = 2500$ m; $v = 2$ m/s). . . . .	68
3.4	Comparing the search area gain (y axis) for the Synchronous Rendezvous (SR) method and the benchmark scenario given variable mission time (x axis) and different number of targets (increasing the number of targets with each model from the top plot towards the bottom one.) The higher on the y axis the result, the better the approach performance. On the top plot, on average, the SR method is outperforming the benchmark. Increasing the number of targets allows for the benchmark to gain more area and to outperform the SR method (bottom plot). . . . .	76
3.5	Vehicle task allocation - average number of search vehicles (y axis) per mission (x axis) normalised over the number of SRs in the particular mission. The benchmark is set to a constant of two vehicles for all missions as the vehicles’ tasks are fixed. The SR simulations corresponding to the three target models show the dependency of task allocation in response to workload. . . . .	77
3.6	Search area comparison . . . . .	78
3.7	Reacquire-Identify (RI) resource allocation comparison between the Synchronous Rendezvous (SR) and benchmark methods. The RI analysis shows the ability of each method to satisfy the condition that all targets need to be revisited. The green line shows the battery life limit of a vehicle. If the simulation result is above the time limit, this would be a mission failure. Since the SR method can retask the vehicles during the mission, it can swap search resource for RI in order to achieve a successful mission at a cost of reduced area size. The benchmark method has set vehicle tasks and cannot adapt to the workload, resulting in multiple failed missions. . . . .	80

3.8	Percentage of resource loss (y axis) for a number of repeated simulations (x axis). Blue dashed line shows the SR loss. Red continuous line shows the base case loss. First three plots show the loss for different number of simulated targets - 5, 10, 15 on average. The last plot gives the SR loss from the previous graphs together so the loss can be compared as a function of changing the number of targets. . . . .	82
3.9	Histogram showing number of Synchronous Rendezvous (SRs) scheduled per mission ( <i>x</i> axis) for all the simulations ( <i>y</i> axis). The three bars correspond to simulations with 5,10 and 15 simulated targets on average per SR respectively. . . . .	84
3.10	Resource loss due to time spent on Synchronous Rendezvous (SR) in relation to the number of targets. Results are sorted based on increasing loss and are colour coded based on the used target model. . . . .	84
3.11	Average Synchronous Rendezvous (SR) time interval as a function of number of targets simulated per SR . . . . .	85
3.12	Comparison for the time resource loss between the Synchronous Rendezvous (SR) method and the benchmark. No adjustment to address the benchmark loss source. . . . .	86
3.13	Comparison for the time resource loss between the Synchronous Rendezvous (SR) method and the benchmark. The mission failure for the benchmark case is accounted for by setting the resulting simulations to 100 % loss. . . . .	87
3.14	Evaluation of Synchronous Rendezvous (SR) and base case performance. The SR method and the successful benchmark results are fitted using linear regression. The mixed results from the base case are evaluated using logistic regression. The operational regions of both approaches can be evaluated following the approximated curves. .	88
4.1	Reinforcement learning: agent-environment interaction, [1] . . . . .	92
4.2	Discretised state space . . . . .	96
4.3	AUV path examples illustrating Actions 1,2,3 and 4 from Table 1. Actions 5,6 and 7 are similar to Actions 3,2 and 1, respectively, but the SR duration is based on the predefined $t_{SR}$ time, not the adaptive RI time interval. . . . .	99
4.4	Target models based on sampled Gaussian kernel per unit area . . . . .	101
4.5	Expected average number of targets as a function of time spent on search task. . . . .	102
4.6	Average expected shortest path in $km$ ( <i>z</i> axis) as a function of number of targets ( <i>x</i> axis), uniformly distributed within a search area in $km^2$ ( <i>y</i> axis). Calculations follow Equation 4.11. . . . .	103
4.7	Reward as a function of SR duration — area gain time and penalty for exceeding the expected SR. . . . .	105

4.8	Reward as a function of RI state and action — penalty for carrying over workload between SRs ( $x > 0$ ) or being idle ( $x < 0$ ). . . . .	106
4.9	Value function solution. . . . .	109
4.10	Optimal policy solution of the multi-AUV MCM problem. . . . .	109
4.11	2D slices for each action from the 3D policy solution after convergence. . . . .	110
4.12	Action selection after 1 iteration of the Value Iteration algorithm. . . . .	111
4.13	Rendezvous Point interval duration analysis. . . . .	114
4.14	Average number of targets per unit area analysis. . . . .	114
4.15	Discount factor parameter analysis (see Equation 4.3). . . . .	115
4.16	Vehicle speed and sensor swath analysis. . . . .	115
4.17	Value Iteration termination - convergence condition based on small change in the value function. Axis $x$ is the value function maximum change between iterations and axis $y$ is the percentage of action change between iterations . . . . .	117
4.18	Convergence variation . . . . .	118
5.1	A MOOS community representing the publish-subscribe architecture . . . . .	122
5.2	Starting path — predefined lawnmower search for all vehicles. On the left is the behaviour plot showing the heading and speed state space of the vehicles during search mode (waypoint behaviour). On the right is a still from the simulation (post-processing). . . . .	128
5.3	First synchronous rendezvous point - predefined location and time. On the left is the behaviour plot showing the heading and speed state space of the vehicles during SR mode (loiter behaviour). On the right is a still from the simulation (post-processing) . . . . .	130
5.4	After the first SR, the vehicles adapt their paths based on the targets that were found during the search phase. One vehicle is doing RI, following all detections, and 2 vehicles are following a lawnmower path. All vehicles are following the waypoint behaviour, which was configured dynamically based on parameters calculated in uFldMergeHazard and pSynchronousRP classes. (post-processing) . . . . .	131
5.5	After the first SR, the vehicles adapt their paths based on the targets that were found during the search phase. One vehicle is doing RI, following all detections, and 2 vehicles are following a lawnmower path. All vehicles are following the waypoint behaviour, which was configured dynamically based on parameters calculated in uFldMergeHazard and pSynchronousRP classes. (runtime) . . . . .	132

5.6 Mismatch in the synchronous rendezvous time and vehicle paths. The problem is due to the merged reports which did not get synchronised. Some vehicles calculated their path and SR based on a partial report. This happens when a message does not get through or when a vehicle is late to arrive at the SR. . . . .	133
5.7 Vehicle does not get an updated track, follows its previous track instead. The problem is caused by the same issue as in Figure 5.7 - the merged target report was not updated. In this case it was caused due to salutation of targets. The merge list did not get through at all. . . . .	134
5.8 Kapitas (Seahorse) vehicle . . . . .	135
5.9 Experiment 1, Mission 1: Position of vehicle ( $x$ and $y$ coordinates) . . . . .	136
5.10 Experiment 1, Mission 2: Position of vehicle ( $x$ and $y$ coordinates) . . . . .	137
5.11 Target . . . . .	137
5.12 Side scan sonar imagery of the target . . . . .	138
5.13 Experiment 2, Mission 1: Position of vehicle ( $x$ and $y$ coordinates) . . . . .	139
5.14 Experiment 2, Mission 2: Position of vehicle ( $x$ and $y$ coordinates) . . . . .	139
5.15 Depth of vehicle ( $z$ coordinate) . . . . .	140
5.16 Ground speed . . . . .	141
5.17 Conductivity, Temperature, Depth . . . . .	142
5.18 Turbidity, pH, Redox . . . . .	142
5.19 Sound speed . . . . .	143
5.20 Longitude and Latitude data points from SeaCat trials . . . . .	144
5.21 Real (blue) vs command speed (red) from SeaCat trials . . . . .	145

# List of Tables

1.1	Comparison between underwater and ground wireless communications . . . . .	24
2.1	Comparison between classical and non-classical planning . . . . .	44
2.2	Comparison between reactive and deliberative control architecture . . . . .	45
2.3	Comparison between model-driven and data-driven mission planning . . . . .	47
2.4	Comparison between learning and planning steps in Reinforcement learning (RL) . . . . .	48
4.1	Action space . . . . .	99
4.2	Reward function . . . . .	104

# Abbreviations

**AI** Artificial Intelligence.

**AN** Attenuation-Noise.

**ASW** Anti-Submarine Warfare.

**ATR** Automatic Target Recognition.

**AUV** Autonomous Underwater Vehicle.

**CADDY** Cognitive Autonomous Diving Budd.

**COLA2** Component Oriented Layer-based Architecture for Autonomy.

**CTD** Conductivity Temperature Depth.

**CTS** Clear To Send.

**DP** Dynamic Programming.

**DTE** Detect-to-Engage.

**DVL** Doppler Velocity Log.

**EM** Electromagnetic.

**FAMA** Floor Acquisition Multiple Access.

**GNSS** Global Navigation Satellite System.

**INS** Inertial Navigation System.

**IvP** Interval Programming.

**LBL** Long Baseline.

**MAC** Medium Access Control.

**MC** Monte Carlo.

**MCM** Mine Countermeasures.

**MDP** Markov Decision Process.

**MLO** Mine-Like Object.

**MOOS** Mission Oriented Operating Suite.

**MRTA** Multi-Robot Task Allocation.

**PDDL** Planning Domain Definition Language.

**POMDP** Partially Observable Markov Decision Process.

**RF** Radio Frequency.

**RI** Reacquire-Identify.

**RL** Reinforcement learning.

**ROS** Robot Operating System.

**ROV** Remotely Operated Vehicle.

**RTS** Request To Send.

**SAS** Synthetic Aperture Sonar.

**SCM** Search-Classify-Map.

**SLAM** Simultaneous Localisation And Mapping.

**SNR** Signal-to-Noise Ratio.

**SR** Synchronous Rendezvous.

**SSS** Side-Scan Sonar.

**TD** Temporal Difference.

**TDMA** Time Division Multiple Access.

**TSP** Travelling Salesmen Problem.

**USBL** Ultra-Short BaseLine.

**UWSN** Underwater Sensor Network.

## List of Symbols for Chapter 3

$A_S$  Search area for a Synchronous Rendezvous (SR) time interval

$B$  Battery threshold

$L$  Loss due to Synchronous Rendezvous (SR)

$n_{RI}$  Number of Reacquire-Identify (RI) vehicles

$n_S$  Number of search vehicles

$N_{MLO}$  Number of Mine-Like Object (MLO) (targets)

$t_G$  Greedy shortest path time to visit all defined targets

$t_H$  Time to acquire a target and collect high-resolution information

$t_{RI}$  Reacquire-Identify (RI) time interval

$t_{SR}$  Synchronous Rendezvous (SR) time interval

$t_{SR(min)}$  Minimum Synchronous Rendezvous (SR) time interval

$t_{SR(max)}$  Maximum Synchronous Rendezvous (SR) time interval

$U\{a,b\}$  Uniform distribution between bounds a and b

$v$  Speed of Autonomous Underwater Vehicle (AUV)

$w$  Sensor swath width

$x$  Width of the mission search area

$y$  Length of the mission search area

## List of Symbols for Chapter 4

$a$  Agent's action (arbitrary)

$\mathcal{A}$  A set of all possible actions

$A_t$  Action selection at time step  $t$

$A_U$  Unit area used to define the number of targets

$d$  Average distance travelled between the targets

$E_{RI}$  Negative reward for not managing the workload inefficiently (idle vehicle or RI credit)

$G(q, u)$  Sampled Gaussian kernel with  $q$  being the truncated signal  $z$  and  $u$  the scale parameter  $u = \sigma^2$

$G_t$  Expected discounted return for a given time horizon

$G_1, G_2, G_3$  Gaussian distribution model for number of targets

$i, j$  Elements of the state space 2D matrix

$k$  Time steps counter (corresponding to SR number)

$K$  Empirical coefficient to calculate  $d$

$l, m$  Length of state space dimensions

$L(z, u)$  Gaussian kernel with signal dimension  $z$  and scale parameter  $u = \sigma^2$

$n$  Number of vehicles

$N_{MLO}$  Number of mine-like objects

$r(s', a, s)$  Reward for transitioning into new state  $s'$  from state  $s$  and action choice  $a$

$D_{SR}$  Negative reward for deviation from  $t_{SR(min)}$

$p(s'|s, a)$  Probability of transitioning into state  $s'$  given current state  $s$  and action choice  $a$

$q$  Number of samples for the Gaussian kernel

$\mathcal{R}$  A set of all possible rewards

$R_t$  Reward at time step  $t$

$R_a$  Positive reward for gaining search area

$s$  Environment's state (arbitrary)

$\mathcal{S}$  A set of all possible states ( $\mathbf{S}_1$  and  $\mathbf{S}_2$  being the continues sets for battery and workload dimensions and  $\bar{\mathcal{S}}_1$  and  $\bar{\mathcal{S}}_2$  the discrete sets)

$s'$  Environment's future state (arbitrary)

$S_t$  Environment's state at time step  $t$

$s_1$  Battery state space

$s_2$  Workload or Reacquire-Identify (RI) state space

$t$  Markov Decision Process (MDP) discrete time step

$T$  Final time step from the Markov Decision Process (MDP) episode

$V$  Value function

$V^*$  Optimal value function

$\gamma$  Discount factor

$\mu$  Gaussian distribution mean

$\sigma$  Gaussian distribution standard deviation

$\pi$  Policy, a sequence of actions

$\pi^*$  Optimal policy

# Chapter 1

## Introduction

The adoption of Autonomous Underwater Vehicles (AUVs) has been gradual and steady over the past few decades [2]. In contrast to the recent rapid advancement of airborne drones, the progress of ocean platforms is slower due to environmental and operational constraints, cost and commoditisation, as well as the notion of threats faced. However, the potential benefits of using autonomous systems in areas such as search and rescue operations, environment and equipment monitoring, and security has prompted continuous progress in the field.

Currently, AUVs cope better with planned missions in constrained environments than with unexpected circumstances and are, therefore, closely monitored by operators. In deploying AUVs, there are presently two broad approaches. The first is to design missions for single vehicle operation with expensive sensors and mechanisms to minimise the chance of failure or loss. The other is to use multiple vehicles with less sophisticated sensors but with the network of vehicles being able to reconfigure and retask the nodes. Such a system has the potential of reducing overall cost and increasing robustness, but introduces the non-trivial problem of managing the available assets [3].

Since the early prototypes, autonomous underwater vehicles were designed with diverse sensors and aimed at general purpose missions. However, there are performance limitations that leave many important applications outside the capability of a single vehicle. This issue is particularly relevant when there are vast areas that need to be monitored, explored or mapped. Having multiple vehicles could potentially result in increased performance or even permit tasks to be performed that are currently impractical. However, there are still very few examples demonstrating cooperative AUV capabilities [4]. This work aims to explain some of the barriers to such an approach, but, more importantly, it

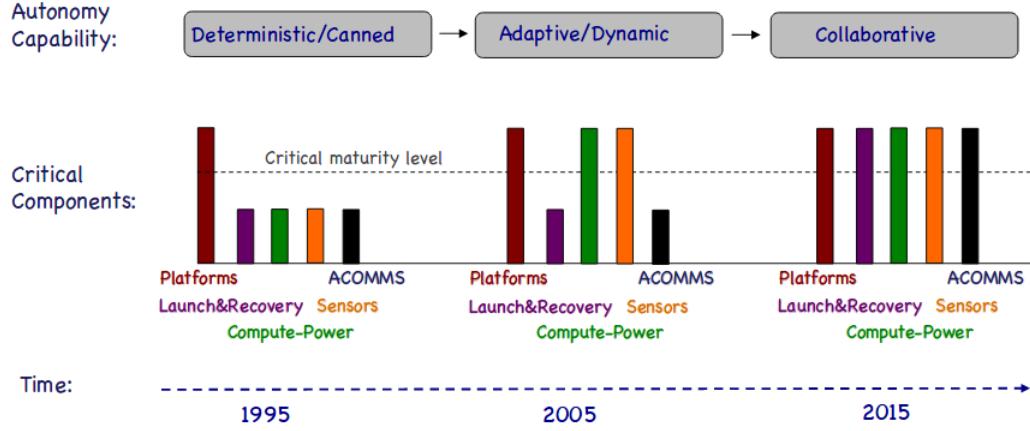


Figure 1.1: AUV autonomy capabilities [5]

presents possible means to tackling them. The targeted goal of the project is designing a system that can adapt to an unknown and dynamic environment, and, at the same time, can reason about the best way to allocate its resources.

## 1.1 Motivation

Due to the adverse conditions of the underwater environment, the early development and experimentation of AUVs involved strict predefined mission planning, with sensors only collecting data. In Figure 1.1, showing the progress of marine autonomy, the first stage from the mid 1990s is defined as deterministic. In order to reach adaptive capabilities, the sensors start to give feedback to the system and the vehicle can adapt its path or plan online. Nowadays, AUVs are able to work in a network by enabling cooperative behaviour — using both their own sensory data, as well as incoming information from other vehicles.

This evolution in autonomy stems from advancement and maturity of multiple critical components within the field of AUV research and development. The platforms required enhanced performance over the early AUV designs. This was a result of autonomous operation in the water requiring robustness and durability, since no monitoring was possible. Current platform design tends to reduce size and weight with the idea of making the AUV portable by people, rather than relying on cranes and additional equipment [6]. Launch and recovery efforts are still ongoing as the task is not trivial in bad weather conditions. Common designs include a cage which is equipped with a homing acoustic beacon. Once the vehicle is close enough, optical guidance [7] can be used to adjust the position of the vehicle with higher precision, or attach a line to recover the vehicle to the cage [8]. The limitations



Figure 1.2: AUV designs: top left: SeaCat, bottom left: Remus , top right: ecoSub , middle right: Girona 500 , bottom right: ALR<sup>1</sup>

of on-board processing and vehicle endurance can be seen in a recent example of power system design and computer architecture developed for the Girona 500 AUV [9]. The variability of vehicle design and capabilities is broad and some AUV examples can be seen in Figure 1.2. Finally, work on real-time sonar imagery processing is also ongoing [10]. Advances in the sensor suit and communications are given in separate subsections, as they are directly relevant to the methods developed in the current work.

AUV technology has reached the state where their capabilities and cost allow organisations to use multiple machines in cooperative missions. There is still very little intelligence left to the vehicles themselves as most decisions are preprogrammed and contingencies dictate mission termination at any discrepancy. Such an approach has little to contribute to the development of a cognitive underwater network. To obtain a truly cooperative system, the need for constant communication between the

<sup>1</sup>Photo credit: SeaCat - Atlas Elektronik GmbH; Remus - Woods Hole Oceanographic Institution; Girona 500 - University of Girona; ecoSub, ALR - National Oceanography Centre

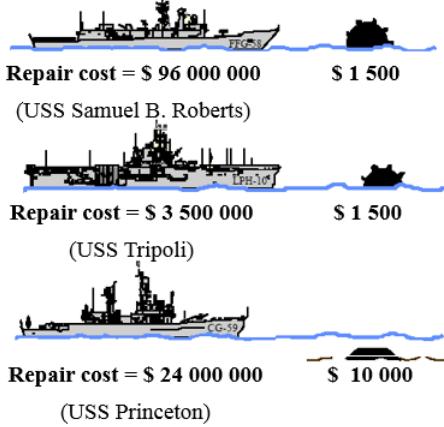


Figure 1.3: Recent damage caused by mines to United States warships [11]

nodes is crucial and this cannot be guaranteed in an underwater environment. Some of the physics behind this is explained in the next section for further understanding.

Although such cooperation is not a trivial problem, the benefit of multiple vehicles working together comes from the fact that their mutual contribution exceeds the sum of the work that each of them could do alone. Such a capability is useful when there are not enough resources in the system to accomplish a task. This problem of resource management often arises when dealing with the threat of naval mines — an asymmetric weapon that is relatively low-cost and easy to deploy but involves significant effort to neutralise. Should the mine damage a ship, the resulting costs can be substantial, as shown in Figure 1.3, [11].

Another issue is the number of mines that are still lying on the ocean bed, having been deployed during times of tension or war. Some mines contain chemicals that leak into the water with time, others are still not disarmed and present a hazard for vessels. The threat, however, is not only from the past wars, since estimates suggest that over a million mines are stored worldwide (excluding United States weapons). In more than 30 countries, naval mines are in production, and most of them also do export. Improvised explosive devices are not accounted for in these numbers; however, they are affordable and the design can be relatively simple [12]. There is a major drive for defence industries to develop means of dealing with mines in both combat and peaceful territories.

Mine Countermeasures (MCM) based on AUVs is the application focus of this work. One of the reasons autonomous vehicles are considered as potential future technology for mine hunting operations is that they could provide a distributed system to deal with searching large areas and looking for targets from close proximity. Another significant advantage is that AUVs can be used for covert operations: they do not put human life at risk and have the potential of reducing the cost of such missions once their

Table 1.1: Comparison between underwater and ground wireless communications, [13], [14], [15]

Wireless communication	Acoustics underwater	RF on ground
Speed	1500 m/s	300 000 000 m/s
Frequency	kHz	GHz
Bandwidth	~kHz	~MHz
Data Rate	~kbit/s	~Mbit/s

maturity is reached. Introductory details on mine classification and countermeasures are provided in this chapter.

## 1.2 Physics of the environment - acoustic propagation

Reliable wireless communications is the basis for successful cooperation between robots, since they need to share status and location information as well as knowledge of the surrounding world. The major difficulties in the domain of underwater robotics can be explained by examining the harsh conditions presented by the physical environment. Signal transmission is severely impaired in sea water and the quality and rate of data transfer between the nodes in a system is seriously reduced compared to an equivalent network deployed on the ground. A brief comparison is presented in Table 1.1. A graph comparing the attenuation for acoustic and Radio Frequency (RF) signals in seawater is given in Figure 1.4 [13], [14], [15], showing nearly 1000 dB/km difference for ~kHz frequencies.

Due to the high absorption in seawater of Electromagnetic (EM) signals, both Radio Frequency (RF) and optical, they are not considered a feasible solution for underwater communications yet, as they only propagate for very short distances. However, applications such as docking and data transfer are driving the development of very short distance high speed communications; some recent studies are reporting data rates of about 10 Mbps for distances of less than a metre [16].

Optical transmissions can reach high data rates, allowing for image and video data to be sent between remote nodes, but their performance is very sensitive to the water quality. Reported data rates vary between technologies, tradeoffs and experimental environment, with recent results between 1.2 Mbps to 5 Mbps for distances of 30 to 200 m, respectively [17]. The intensive scattering of light in shallow regions restricts the usage of green-blue lasers to deep waters as means of reliable medium/long range communication. However, the threat of mines is most common in shallow waters, near shores and harbours.

Currently, the only practical solution for shallow water wireless communications is acoustics. Some of its most restrictive characteristics are slow propagation speed ( $\sim$ 1500 m/s), large and variable

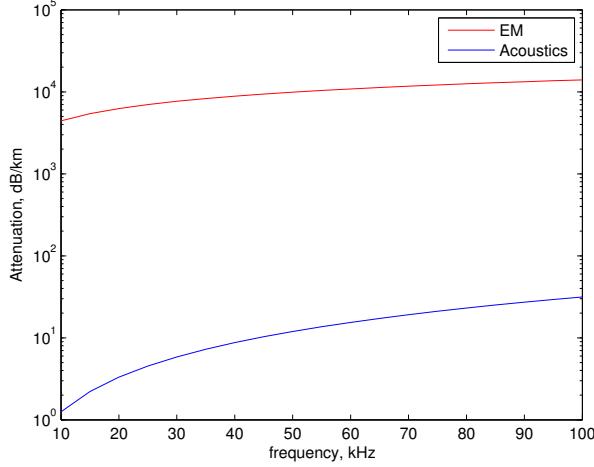


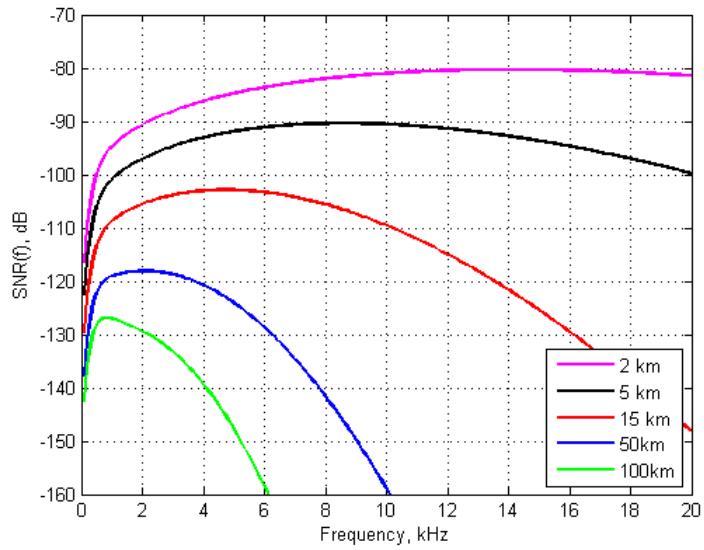
Figure 1.4: Comparison between acoustic and RF attenuation in seawater [15]

latency, low bandwidth ( $\sim$ kbytes) and available data rates ( $\sim$ kbytes). The bandwidth and data rate restrictions are due to available frequencies of propagation, where the lower limit is set by the various noise contributions and the attenuation imposes the high limit. Some other factors that affect the channel performance are multipath, Doppler spread, variable and long propagation delays.

Figure 1.5 [18], [15] shows how the Attenuation-Noise (AN) product affects the Signal-to-Noise Ratio (SNR). The attenuation, or path loss, is a function of both signal frequency and travelled distance, and the ambient noise power spectral density is a function of frequency. Taking into account these two functions, the effect of various distances and frequencies can be evaluated for the Signal-to-Noise Ratio (SNR) of an acoustic signal. The reason the  $y$  axis of the plot is measured in negative  $dB$  is because the power of the signal is assigned to 1, rather than using a specific modem power specifications ( $SNR = 1/AN, dB$ ). It can be seen that increasing the distance between nodes shifts the optimum SNR towards lower frequencies, giving the trade off between distance vs data rate.

Figure 1.6 [19], [20] separates the sources of the ambient noise, responsible for the low end attenuation in Figure 1.5. The graph shows that at low frequencies, turbulence and seismic effects have the highest noise spectral density level (the mean of a square root of background noise pressure within a specified bandwidth,  $dB$  re  $\mu Pa^2/Hz$ ). Then, at frequencies between  $\sim$ 10 and  $\sim$ 1000 Hz, the main noise contributions are shipping activities and marine life. The sea condition, waves and wind, is responsible for noise in the higher frequency spectrum. Despite the limitations of using acoustics for underwater communications, other types of physical signals, such as RF and optical, are strongly attenuated or scattered, and this prohibits their use for distances greater than a few metres [21].

The noise and attenuation are environmental conditions that impose hard boundaries on the acoustic



, [15]

Figure 1.5: Acoustic frequency limitation based on 1/Attenuation-Noise (AN) for various distances [18]

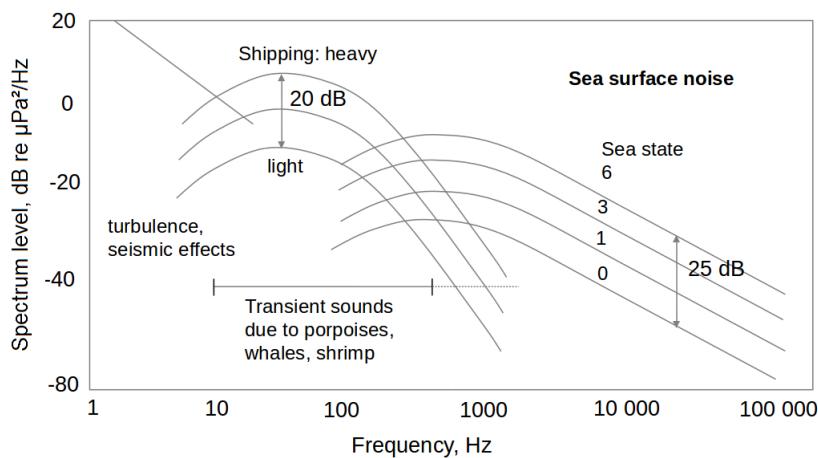


Figure 1.6: Propagation of noise sources in water<sup>2</sup>[19], [20]

signal propagation. However, effects like multipath, due to signal scattering from the sea surface and bottom, are affecting techniques for efficient exploitation of the bandwidth. Physical layer approaches, targeting multipath and increasing the available data rate, include better modulation techniques [22], as well as multiple-input multiple-output techniques, such as spatial multiplexing [23].

The high propagation delays, together with low bandwidth and a challenging acoustic channel make networking protocol solutions, such as Medium Access Control (MAC) and routing, critical to enable cooperative robotic systems [21]. Recently, the security of the network is being taken into account when designing underwater protocols [24].

### 1.2.1 Navigation

A direct consequence of the underwater physics properties makes the problem of navigation and localisation unique for AUVs. Good navigation information is necessary for safe vehicle operation, as well as accurate data collection. The Global Navigation Satellite System (GNSS) signal does not propagate in water. There is no easy way of eliminating positioning error without a vehicle surfacing or receiving corrections from a surface node or designated built-in structure. This causes issues with navigation especially when taking into account the ocean currents which can steer a vehicle in undesirable directions [25].

The methods used in underwater navigation can be divided in three categories and recent advances are summarised in surveys [26], [25]:

- Dead reckoning and Inertial Navigation System (INS). Dead reckoning is the simplest method for navigation - integrating the vehicle velocity over time to obtain the estimated position. However, it can be very imprecise due to water currents. INS is more precise, with an expectation of several metres to kilometres drift per hour, depending on the available aids and cost [27].
- Acoustic aid - using beacons, or external transducers, as guidance for the vehicles. The primary methods are Long Baseline (LBL) and Ultra-Short BaseLine (USBL). LBL relies on multiple deployed transponders. A ping from the vehicle is received by the LBL nodes and the position is usually estimated based on triangulation. Expected position accuracy is a few metres within a few kilometres distance. USBL relies on a receiver mounted on the vehicle that can measure the angle and range of an acoustic beacon.
- Terrain Based Navigation [28] - using a map of the environment to estimate the position by

---

<sup>2</sup>Figure adapted from a slide from Radar Systems lecture notes by Prof. Hugh Griffiths.

referencing to known measurements in the bathymetry, gravity or magnetic field. Problems with this method are lack of accurate maps, as well as computation cost [29].

### 1.2.2 Imaging sensors

Much as for navigation, there are performance implications for the sensor suite of an AUV due to the underwater environment. Optical sensors have limited application. Cameras can be used at a distance of a few metres, and often an artificial light is required. Hence, acoustics is also the basis of most sensors mounted on modern underwater vehicles. Those include sonars with various functions and their efficiency is strongly dependent on the conditions of the sea and terrain of operation. In the case of MCM, the area of interest is in the shallow zone of the water basin and the transmitted signals are often reflected from the sea bottom and surface resulting in blackout of the sensor or modem.

Typically Side-Scan Sonar (SSS), Synthetic Aperture Sonar (SAS) and multibeam are used as imaging sensors to search for objects on the sea bottom:

- SSS - mounted on both sides of the AUV hull, a fan-shaped beam is emitted. The intensity of the reflections are recorded and stitched together to form images in the direction of movement of the vehicle. Higher frequency increases the resolution. Good imagery relies on stable vehicle movement in a straight line. A well known limitation of this sensor is that the along-track resolution decreases as the range is increased. Detection and classification of shapes on the sea bottom is enhanced by the shadows created behind objects.
- SAS - improved and range-independent resolution compared to SSS. The enhanced performance is achieved by illuminating the same point with multiple pings and forming a multi-aspect image by coherent addition. This is achieved by transmitting and receiving larger bandwidth, which makes the aperture size smaller. However, to achieve a useful along-track sampling rate, an array of hydrophones is required, which makes the size of the sensor larger, and its complexity higher. Another limitation of the sensor is its sensitivity to motion, requiring a very stable AUV platform [30].
- Multibeam echosounder - emits multiple beams in a fan shape. A depth map, not image, is produced as the sensor collects only the range information from the scattered signals. Multibeam is used for bathymetry, but its object detection potential has also been explored [31].

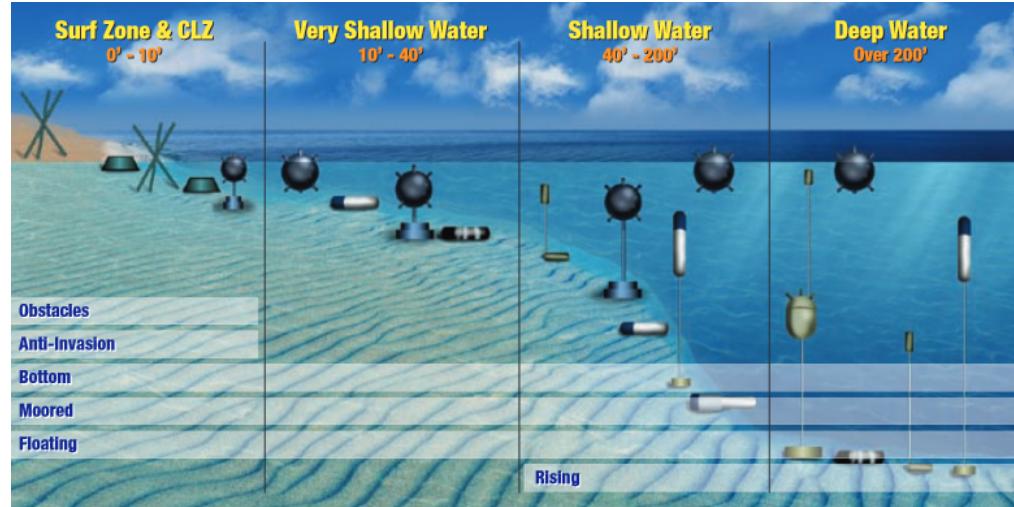


Figure 1.7: Mine classification for different warfare regions<sup>3</sup>

### 1.3 Naval mines and countermeasures

After considering the environment, the specifics of the application are the next most important consideration to take into account when designing an Underwater Sensor Network (UWSN). In this section, the threat of sea mines is explained, followed by some of the current means of tackling it.

#### 1.3.1 Mine Classification

The purpose of a naval mine is to stop or slow an enemy advance. It provides a relatively low-cost solution in terms of hardware and deployment, yet it can cause substantial damage to the opponent. Therefore, entering a minefield is considered a great risk and sometimes even dummy mines can be as effective as real ones.

To understand the challenges imposed to effective MCM, some common classification techniques are listed. This also gives an insight into why some countermeasures are becoming obsolete and can be used as an idea of future systems requirements.

#### Mine classification by type

Classification by type for most common mines in shallow waters include bottom, moored and floating, however there are other types and distinct warfare regions as seen in Figure 1.7.

---

<sup>3</sup>Photo credit: Wikimedia Commons



Figure 1.8: Bottom mine type Manta, partially buried due to ocean activity over time [32]

Bottom mines lie on the seabed. They are non-buoyant so greater quantities of explosive can be fitted, often making them more powerful than equivalent-sized buoyant mines. With time, the mine can become buried or covered with vegetation creating natural covertness and becoming harder to detect as shown in Figure 1.8 [32]. The threat of bottom mines is considerable since they are the ones most used by non-NATO countries [32]. Common triggers are magnetic and seismic.

Moored mines are buoyant — they have less explosives within their hull and so the operational radius is reduced. They are often attached to an anchor to keep the depth constant. Various types of triggers are typical - acoustic, magnetic, proximity, pressure. An example of a moored mine is shown in Figure 1.9 [32].

Floating mines are both buoyant and drifting freely with currents and tides. Since they are non-static, it is hard to map a minefield. Often such mines are inexpensive as they use a simplistic design. They are an easy solution when moored or bottom mines are hard or ineffective to deploy. However, their major disadvantage is endangering friendly units. An old drifting mine can be seen in Figure 1.10.

There are other distinctive types of mines but they are designed for more specific applications. The ones of interest for this thesis are the bottom mines which can often stay on the sea floor for decades and become harder to detect with time. Using AUVs for closer inspection of the ground could improve current mine hunting operations.

### Mine classification by actuation

Classification based on actuation includes common triggers, such as contact, influence (EM, pressure, seismic) and command and control. Such a distinction is important as it highlights when particular

---

<sup>4</sup>Photo credit: US Militia Forum page, URL: [www.usmilitariaforum.com](http://www.usmilitariaforum.com), Last Accessed: 04 September 2017



Figure 1.9: Moored mine being neutralised by divers [32]



Figure 1.10: Floating mine<sup>4</sup>

methods for clearing mines are effective.

Actuating a mine by contact requires a physical contact between mine and vehicle. This is the oldest and simplest approach. It was widely used during the First Gulf War [32].

There are various detectable signals created when a vehicle is moving through the water. Those are utilised in the influence triggering mechanisms. The acoustic signature is the most common influence trigger. Hull noises, engines and propellers can be detected by the hydrophones mounted on mines. Mine sweeping is an efficient approach to deal with acoustically actuated mines.

Another type of trigger senses electric and magnetic distortions. The magnetic trigger detects the slight change in the Earth magnetic field when the large metal hull of a ship is moving; the electric trigger is similar but it detects the small currents created by different metals immersed in water – normally the hull and propeller are made from different metals. Both are susceptible to sweeping.

Pressure signatures are often used in combination with other triggers as the sensor can be activated not only from the pressure waves created by the ship, but also from wave action. It is extensively used for countering mine sweeping. Seismic triggers are used for bottom mines as the seismic waves are transferred through the sediment. The mechanism is similar to acoustic triggers as vibrations causing seismic disruptions are made from sound waves.

Another common actuation mechanism is command and control - a signal is sent to the mine for detonation. The major advantage is that mine sweeping is ineffective as a deactivation method. However, cables and acoustic links are required.

### **Mine classification by deployment mechanism**

Distinguishing between the mine deployment mechanism defines the reaction time for countermeasures, but also can give a clue of the type of mines being deployed and their pattern. Classification by deployment techniques include ship, aircraft and submarine.

The advantage of using an aircraft to lay mines is the fast response time. It can also freely access minefields and the surface area. A downside is that the precision of placement is low and the amount of weapon load is smaller. It is, however, the preferred approach in offensive scenarios.

Submarines have the advantage of being covert. The disadvantage is their long response time since they need to return to base and load mines instead of torpedoes.



Figure 1.11: Mine layer of the Iraqi Navy<sup>5</sup>

Surface vessels are preferred in defensive operations, so needs to be in friendly waters. This approach can deploy mines with a large weapon load and accurate placement [32]. Figure 1.11 shows a mine layer from the Iraqi Navy.

Classification information can be used to define basic working parameters, such as required scanning area and path, size and amount of sonar images needed for classification and identification.

### 1.3.2 Countermeasures

As mentioned above, there is a great diversity of types of mines and ways that they can be actuated and deployed. This creates the need for trade-offs in designing MCM systems and optimisation based on priorities of parameters, such as area coverage, time of completion, probability of classification, covertness, robustness, etc. This section explores existing ways of tackling the naval mine problem.

Minesweeping is a technique for naval mine removal by detonating or capturing. It does not include detection prior to clearing an area. Usually a ship or aircraft has specific equipment for this task. A helicopter during mine sweeping exercise is shown in Figure 1.12. The process can be classified as contact sweep when moored mines are cut from their wires, or distance sweep - when a certain characteristic is mimicked that actuates a mine. However, modern mines rely on the influence of a combination of signatures and such methods are becoming inefficient.

---

<sup>5</sup>Figure adapted from a slide from Radar Systems lecture notes by Prof. Hugh Griffiths. Photo credit: United States Navy



Figure 1.12: An MH-53 helicopter, during a mine sweeping exercises in the Persian Gulf in 2014 - equipment that cuts the chains off sea mines is towed and the mines are revealed by floating to the surface<sup>6</sup>

Mine hunting includes detection and classification via Side-Scan Sonar (SSS) (usually), thus providing a controlled way of clearing a mine field by predefining a probability of detection. Currently, the common way is using a surface vessel that scans the area. The neutralisation is done by a Remotely Operated Vehicle (ROV) directed towards the identified target. The main difference between an ROV and AUV is that an ROV is connected to a surface vessel through cables, allowing remote navigation. The cabled connection provides a robust way for transmitting command and control messages to the ROV, as well as online imagery and other sensor data to the operator on the surface. In the case of MCM neutralisation, there is explosive placed in the ROV and it is critical that the operator has direct and instantaneous control over the vehicle. The downside of using an ROV is that the cable limits its operation. An example of a mine hunting ship and neutralisation ROV during an exercise are presented in Figure 1.13. A major disadvantage of using a minehunter is that the people on board operate in close proximity to a mine field. To reduce the threat, the ship's signatures are minimised but this makes the design of such vessels complex. An alternative to using a manned vessel is to use an autonomous boat operated wirelessly from a safe control station. Another drawback of using a surface ship is that it lacks flexibility when a target needs to be observed from close proximity in cases such as low sensor visibility or the need to explore an object from various angles. This is one of the reasons AUVs are gaining attention for use in MCM — they can form or be incorporated into a distributed network pattern.

---

<sup>6</sup>Photo credit: Stars and Stripes page, URL: [www.stripes.com](http://www.stripes.com), Last Accessed: 04 September 2017

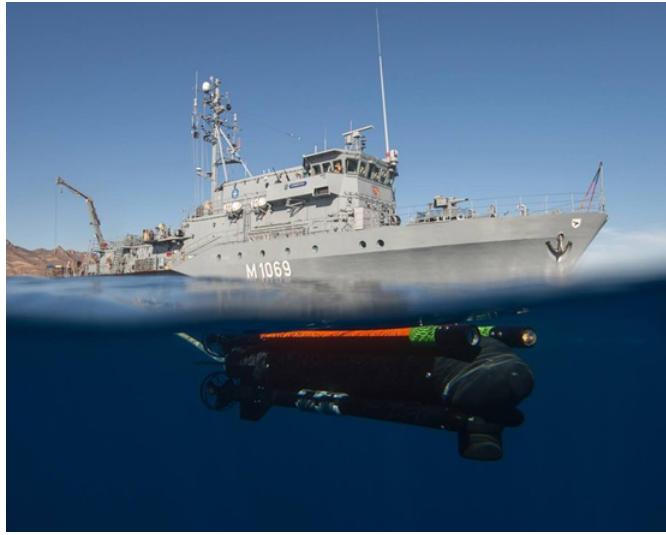


Figure 1.13: A Seafox mine neutralisation ROV launched from the German minehunter FGS Homburg<sup>7</sup>

Currently, the most reliable way of performing an overall mine clearing operation is using divers. The greatest disadvantage of this approach is the exposure of human personnel to the dangers of the minefield. They also have finite endurance and speed [32]. However, divers are still used for MCM operations, especially in the neutralisation phase.

Training dolphins and sea lions as part of the human team is another way of dealing with the naval mine threat. The advantages are that these animals are more agile and enduring than men, and more intelligent than current sensor-based systems. Their training is also less expensive than sending teams of divers or AUVs. However, disadvantages include ethical issues and the risk of misunderstanding between human and animal. Examples are the US Marine Mammal programmes that train the human-animal teams, or ‘mark’ numbers, MK 4, 7 and 8 using dolphins, MK 5 — sea lions, and MK 6 — both sea lions and dolphins [33]. This MCM approach is uncommon but it was used during the Gulf War and an example of a mammal currently in training can be seen in Figure 1.14.

### 1.3.3 Mine hunting phases

Replacing conventional mine hunting techniques with alternative technology requires a good understanding of the exact procedures that are currently in use. Equivalent capabilities need to be provided using a system based on AUVs. The mine hunting process consists of five Detect-to-Engage (DTE) phases [32].

1. Search — starts when the node is located at the starting point of the search area of interest. It

---

<sup>7</sup>Photo credit: Royal Navy page, URL: [www.royalnavy.mod.uk](http://www.royalnavy.mod.uk), Last Accessed: 04 September 2017



Figure 1.14: A sea lion from the Navy Marine Mammal Program salutes its handler before diving as part of an International Mine Countermeasures Exercise in the Middle East in 2014<sup>8</sup>

could include usage of a preprogrammed search pattern, adapting to and avoiding obstacles, and modifying the offline path based on dynamic requirements.

2. Detection — information from sensors to indicate a contact with Mine-Like Object (MLO). This phase deals with recording the MLO's location, environmental parameters and creating a message for post-processing.
3. Classification — a decision whether an object is a mine-like or non-mine like. In the case where it is not, the system ignores it.
4. Identification — determines if the MLO is bottom, moored or floating mine. A decision is made to mark if the type and location of the object is dangerous for the platform and needs to be avoided. Often, there is a very long delay between the Identify and Neutralise (Engage) functions, due to the system lacking the ability to autonomously Classify and Identify which means the data have to be recovered and processed first off-board.
5. Neutralisation — a mine is considered neutralised once its location is defined and it can be avoided. In case the platform cannot evade it, other measures are adopted. Those can include destroying or disabling the mine's detection or detonation ability. One problem is that the Engage function can be initiated some time after the Identification function, and the object has to be re-localised. Neutralisation is done in three ways: torpedo (mini) launched at target, jamming/disabling actuation mechanism or deflagration (heating and burning the mine's explosive).

<sup>8</sup>Photo credit: Stars and Stripes page, URL: [www.stripes.com](http://www.stripes.com), Last Accessed: 04 September 2017

Another relevant consideration is planning the phases immediately preceding and following the DTE process: transit — carrying and deploying vehicles to the drop point as well as engaging the equipment needed for the assets to be oriented and the required start location determined, and recovery — collecting nodes at the end location. Those are currently non-trivial problems for AUV operations. All the DTE phases need to be considered for a mine hunting system based on AUVs. This thesis focuses on providing adaptive online search pattern and mission planning techniques. Detection and Classification are assumed possible on-board which makes the work suitable for the near future.

## 1.4 Problem statement and key contributions

This thesis focuses on utilising multiple AUVs for mine hunting. The major problems of this approach are limited communication in the underwater environment and finite resources. This calls for development of effective and robust methods for cooperation between robots, and enabling the vehicles to autonomously prioritise mission goals based on their own sensory data and information communicated from other vehicles. Therefore, the problem this work addresses is:

Mine-hunting mission planning for multiple autonomous underwater vehicles in a  
communication and resource constrained environment

This thesis proposes the Synchronous Rendezvous (SR) solution - dynamically assigning a sequence of meeting points, where the AUVs can robustly share information and make decisions. From an operation perspective, this method gives the advantage to monitor the status of the network at predictable intervals of time. For a defence application, or when the assets are of high value, this is a vital step in adopting autonomous mission planning [34]. The other advantage comes from the ability to retask the existing assets based on the detected threats, thus using the system resources efficiently. The SR benefits come at a cost - the meeting point requires loss of resource due to the vehicles abandoning their mission tasks and travelling to the meeting point.

A successful outcome for this work is showing the feasibility of an AUV network completing a MCM operation by adapting autonomously to changing surroundings in a robust manner. Since experimental data is too costly to collect, this work relies mainly on simulation. The following chapters evaluate and develop the SR method and provide the key contributions of this thesis:

1. Chapter 2: Identifying the gaps in literature relevant to mission planning, MCM application and underwater robotic networks.

2. Chapter 3: Defining and evaluating the parameters of the SR method in a MCM scenario using AUVs. The contribution of this chapter is weighting the benefits and disadvantages of the SR method in a numerical simulation.
3. Chapter 4: Moving from adaptive to optimal SR - the vehicles can plan what is the best action sequence solution, given the final goal of their mission. For this, a Markov Decision Process (MDP) model is adapted to the multi-vehicle MCM problem and the SR method allows the definition of an optimal plan of where, when and how often the vehicles should meet, and how to manage the available resources. The SR-MDP method is evaluated in a numerical simulation.
4. Chapter 5: Validating the SR method by taking into account the physical constraints of platforms, sensors and environment. The Mission Oriented Operating Suite (MOOS)-Interval Programming (IvP) middleware is used for simulation. Partial validation of mission components is provided via trial data.

## 1.5 List of publications

1. Conference: V. Yordanova and H. Griffiths, "Synchronous Rendezvous Technique for Multi-Vehicle Mine Countermeasure Operations," in MTS/IEEE OCEANS Washington, October 2015.
2. Journal: V. Yordanova and H. Griffiths, "Rendezvous Point Technique for Multivehicle Mine Countermeasure Operations in Communication-constrained Environments," *Marine Technology Society Journal*, vol. 50, no. 2, pp. 5–16, 2016.
3. Journal: V. Yordanova, H. Griffiths and S. Hailes, "Rendezvous Planning for Multiple Autonomous Underwater Vehicles using a Markov Decision Process," *IET Radar, Sonar & Navigation, Special Issue: Sonar multi-sensor applications and techniques*, 2017

# Chapter 2

## Research Context

The field of Artificial Intelligence (AI) encompasses a very broad multi-disciplinary spectrum of methods and ideas. Different applications and fields often work in parallel on similar methods and goals, and, as a result, the terminology can vary significantly. This chapter introduces some terminology, giving the distinction between general AI definitions, commonly accepted interpretations and how they are adopted in marine applications. It continues with an overview of the recent marine autonomy development with focus on mine countermeasures. The aim is to summarise the common assumptions and trends appearing in current marine autonomy research. This chapter identifies gaps in the field and confirms that Synchronous Rendezvous (SR) method provides a promising solution to some of the common problems seen in the field.

Section 2.1 gives basic terminology and explains the field of AI planning from an uncertainty representation perspective. Section 2.2 examines the literature based on the method choice, while section 2.3 looks at recent work from an application perspective. Finally, section 2.4 presents the basis for the Synchronous Rendezvous (SR) method.

### 2.1 Autonomy and Artificial Intelligence

Artificial Intelligence (AI) studies the intelligent agent and solves problems related to learning, planning, knowledge, and perception, to name a few. Different applications try to solve similar problems and often terminology varies. For example, intelligence, cognition and autonomy can sometimes be used in the same context, or can be defined in a structured hierarchy. While the field of radar uses

‘cognitive’ for an intelligent system, in the underwater domain ‘autonomous’ is the preferred term. Autonomy in robotics might not specify if the robot is able to act by itself based on predefined commands or having a decision-making ability. In underwater robotics, an autonomous vehicle can also be preprogrammed; however, current work in autonomy includes adopting decision-making algorithms and the ability to act upon sensed and communicated data to reach a goal. Derived terms are often used, such as ‘autonomy engine’, which refers to a decision-making algorithm, tool or the middleware with which a vehicle is equipped. ‘Autonomy architecture’ in underwater robotics refers to the control architecture used in AI. Further descriptions and definitions used throughout the thesis follow.

### 2.1.1 Definitions

Starting from the title of the thesis:

- Intelligent (agent) — being able to perceive the environment, and act in a way to maximise the chance to achieving a goal. The definition varies and sometimes it can include the presence of knowledge or the ability to learn [35]. Some authors consider intelligent behaviour when an agent can select an optimal action or trajectory, and some consider it being able to select actions beyond the immediate, or being non-myopic. In this thesis, an intelligent agent is an agent that can define an optimal (or near optimal) non-myopic action policy, while dealing with uncertainty and resource management.
- Adaptive — the ability to adjust or react to change. ‘Adaptive’ is sometimes used interchangeably with intelligent. There are distinct definitions in control theory and psychology for this term but in the AI context the interpretation can vary. In the underwater field, application adaptive sampling is associated with the AUVs changing its trajectory in response to newly received sensory data. In this thesis, adaptive is used as the ability of the agent to take into account sensory data and to adjust its behaviour following predefined rules. The adaptive agent is considered myopic.
- Sensor Network — multiple autonomous sensors monitoring the environment that are able to communicate between each other, or to a data centre. This term usually assumes many small sensors, such as in wireless sensor networks or the Internet of Things concepts. In the underwater context, sensor networks are usually assumed to be multiple static nodes communicating via acoustic signals, but mobile nodes, such as AUVs and gliders can also be part of the network [21]. Sometimes a network comprised solely of autonomous vehicles is called a robotic network

[24]. In this thesis, both sensor network and robotic network will be used to address a network of AUVs.

Given the definitions above, the central problem of this thesis can be formulated as developing an intelligent network of autonomous underwater vehicles that can adapt to the changing environment, or update their world representation using available information. The application, mine countermeasures, provides the context of the goal and some of the system resource restrictions.

The thesis problem is central to fields such as planning, learning, control and decision theory, and has applications such as resource and sensor management, cooperative robotics, robotic networks, and task allocation. Some similarities between methods and applications are discussed here and give the basis for the Synchronous Rendezvous (SR) method development.

### 2.1.2 Classical Planning

Planning is a central topic in AI and aims at creating a plan of action to reach a goal. Classical planning techniques applied in robotics date back to the 1960s. They are defined for environments that are fully observable (all states are known, current state is known), static (change is observed only when the agent performs an action), deterministic (no uncertainty of the action outcome) and discrete (actions). The state representation is usually done using Situation Calculus, which defines classes based on specific properties of the world. This is used to avoid explicit state enumeration. The transition probabilities are defined using logic representation resulting in success or failure of an action [36]. These propositional algorithms lead to the extensive use of heuristics as solutions [35].

Classical planning has adopted different techniques over the years to relax some of the assumptions described before, that make it more suitable for real world problems. For example, taking into account the temporal and resource constraints. In [37], [38] a partial-order temporal mission planner is presented for an AUV inspection task. It aims at optimising the time to execute a mission by scheduling the vehicle's waypoints based on the time it would take to travel the distance between these waypoints. The planner orders the sequence of actions only partially and adapts to new data by changing the mesh of predefined waypoints.

Another example of relaxing the classical planning constraints is using goal-driven methods which facilitates the formulation and prioritisation of goals in event of unexpected failures or opportunities [7]. An example of a goal-driven agent applied to a United States Navy strategy simulator is ARTUE [39]. The task is to give orders to the available vessels (ships, planes and helicopters) while receiving

information about their varying position, fuel level, speed and heading. The simulation results showed that the agent can respond to unexpected events by reasoning which goals to follow or change. A similar agent was updated to work with continuous variables by setting operational boundaries. It managed to detect a discrepancy for an AUV trajectory and resource monitoring in simulation [40].

The disadvantage of temporal and goal-driven agents, or any agent based on an hierarchical ontology structure planner, is implementing the logic behind each knowledge element, as well as handling exceptions of the rules organising the knowledge. This includes specifying the expectations of each action, defining explanations for detected discrepancies and providing a variety of goals depending on the state.

### 2.1.3 Non-Classical Planning and Probability

Using logic to express uncertainty in classical planning has the limitation that the agent cannot quantify the difference between conditions, for example, how likely two outcomes that were not selected have been. Probability theory helps to quantify uncertainty by providing a measure for likelihood. Often, it is useful to be able to express a belief based on evidence, rather than be certain if a statement is true or not. The uncertain knowledge representation of variables' dependencies uses a Bayesian Networks structure. It is a directed acyclic graph that describes a joint probability distribution between variables. The Bayesian network is a direct equivalent of propositional logic representation in classical planning for definite knowledge.

The uncertainty usually comes from noisy perception or observation (uncertainty about the current state), or stochastic action outcome (uncertainty about the future state). These two concepts are referred to as a sensor model and transition model and they form the belief state of the agent - the possible states the agent can reach, or a temporal probability model.

Acting and reasoning under uncertainty is the study of decision theory. Probability is useful for estimating and updating the world model, and for projecting a control policy ahead in the future. Utility theory helps to define the preferences and goals of the agents. Combining utility and probability results in decision theory - the agent can weight and choose the best action based on the how desirable certain outcomes are, as well based on their likelihood. The Bayesian network representation is extended to include decision and utility nodes.

Often, being able to make the best choice for the immediate action is not enough to reach the best long-term utility. This is when the agent is shortsighted or myopic. The problem of sequential decision

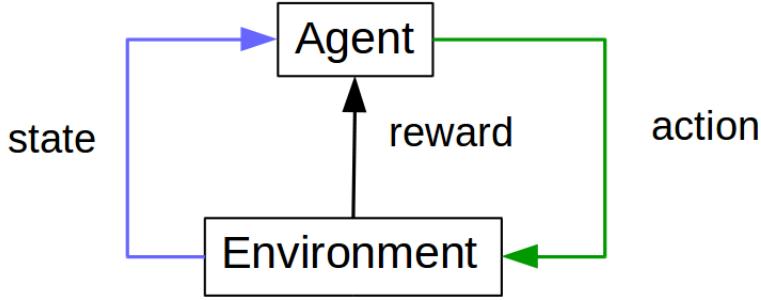


Figure 2.1: Markov Decision Process (MDP) framework

making deals with an agent that has foresight and can estimate what will be the best actions for multiple steps ahead. This is a non-myopic agent.

Stochastic decision making is modelled by a Markov Decision Process (MDP), which adds a specific conditional independence to the Bayesian network model (the Markov property). A MDP is defined by a transition probability model between states and actions, and a reward function, providing the reward the agent can achieve. A representation of this is shown in Figure 2.1. The solution of the MDP is an action policy — the sequence of actions, given the agent’s preference (reward function) and its belief of the world (transition probability). The major disadvantage of the approach is its high computation load even when dealing with simple problems. There are few representative examples of MDP in underwater applications, and in robotics in general. A more formal definition of the framework’s elements and relationship is given in Chapter 4.

An extension to MDP is Partially Observable Markov Decision Process (POMDP) - this encompasses uncertainty about the current state of the agent. This is the noisy sensory input for a robot. While POMDP is a more realistic model compared to a MDP, it adds substantial complexity and has more limited use for real-time planning systems.

A comparison between classical and non-classical planning methods and representations is given in Table 2.1 with the overview following ideas from [35].

## 2.2 Classification of Methods

In the literature, there are some common classifications where authors place their algorithms and methods. Rooted in the history of AI is the distinction between Reactive and Deliberative control architectures. This is the difference between an agent having an internal model of *Sense → Act* in

Table 2.1: Comparison between classical and non-classical planning

Classical Planning	Non-Classical Planning
Deterministic action outcome	Stochastic action outcome
Full state observability	Partial state observability
Static environment	Dynamic environment
Propositional representation (relation between actions and states)	Probabilistic representation (relation between actions and states)
Heuristic solutions	Decision theory based solutions
Knowledge-based world representation based on first order logic (hard to handle exceptions)	Learning a joint probability distribution (represents uncertain knowledge)
Ontology	Bayesian network
Planning Domain Definition Language (PDDL)	Markov Decision Process (MDP) or Partially Observable Markov Decision Process (POMDP)

a Reactive architecture, versus a  $Sense \rightarrow Plan \rightarrow Act$  in a Deliberative one. Another distinction often found in literature is the Model-driven versus Data-driven approaches. The difference is that a model-driven approach has an offline plan given as an input, while the data-driven one updates its plan during execution. This is similar to the problem of learning and planning, which is considered as a two-step sequential process in the Reinforcement learning (RL) literature, rather than an alternative choice.

### 2.2.1 Reactive vs Deliberative Control Architectures

Reactive or behavioural architecture is composed of a set of different robot behaviours. The robot reaction to the sensed environment emerges by continuously activating the suitable behaviours for the particular situation. This architecture is suitable for dynamic environments.

A notable example of a reactive architecture is subsumption. It decomposes the robot's behaviours into a hierarchy and gives them different priorities. The next action, given by a set of behaviours, is defined by the current situation, or the sensory information. One issue with this approach is that it might lead to conflicting goals - one behaviour's action might deviate another behaviour from its goal.

Deliberative architecture is one in which the robot maintains a representation of its internal state. The robot predicts the effects of its actions based on that internal model and creates a plan. Such control architectures are considered hard to adapt to a highly dynamic environments. Some ways to mitigate this effect are replanning or exception handling, as discussed in the Classical planning methods.

Finally, a hybrid architecture, which takes from both reactive and deliberative approaches, aiming to

Table 2.2: Comparison between reactive and deliberative control architecture

Reactive Control Architecture	Deliberative Control Architecture
no model	maintains model
quick response	slow response
suboptimal	optimal
computationally cheap	computationally expensive
low-level control	high-level control

minimise their limitations. An overview of the types of control architectures and examples of their application on AUVs in the 1990s is presented in [41]. Table 2.2 summarises some of the high-level control architecture distinctions.

### 2.2.2 Model-driven vs Data-driven Mission Planning

Despite the maturity of research on robotics control architectures, AUVs still rely on predefined missions with scripted sequential behaviours, usually a set of waypoints and constant depth. There is no flexibility in terms of dynamic and uncertain environment - the vehicle only collects data, rather than adapting its subsequent actions to it. One of the reasons is the inability to accurately simulate the underwater environment, and test the updated control of the AUV. Sometimes it is the reduced computation ability of the vehicle or the legacy software. And sometimes, it is lack of operators' trust in increasing the autonomy of the vehicles and granting them power of decision making. Fault detection is still an open research question but is vital to enable robust AUV autonomy. This inflexible offline planning approach is classified as model-driven [24].

The advantage of the offline planned model-driven approach is the predictability of the mission, as the human operator controls it, especially in challenging or unknown environments. Another advantage is the ability to define an optimal plan offline, which might not be possible online due to the limited resources on the platform. One example of using offline optimal planning coupled with classical methods, such as plan repair and contingency planning, is described in [42]. An MDP model for continuous resource management of the AUV's memory and battery is defined. Due to the large computational load of the MDP, multiple plans are predefined, and the vehicle can switch between them. A more typical example of a model-driven approach is multi-phase exploration missions, where further trials are planned based on the data collected in previous ones. This approach was implemented in [43] for exploration of hydrothermal vents using an AUV. Each phase of the trial used a different sensor, from chemical, to multibeam echosounder, to camera, in order to reduce the vent location uncertainty.

Similar is the approach to mine countermeasures - the mission is divided into phases that are executed by different robots with distinct capabilities. Offline planners define the task and area of operation of each AUV [24], making the mission inflexible.

The ability of the AUVs to use past and current data to decide their actions is a data-driven approach. The scope of how the data is utilised varies. The AUV could use the data in lower level control or could use a purely reactive approach, such as modifying its path in response to detected currents or sensor quality.

Adaptive sampling is an application that benefits from a data-driven approach. In [44], a glider fleet adapts its sampling pattern based on the changing position of the remaining vehicles in the fleet, while retaining a human supervisor in the loop for more complex situations. This mission uses simple reactive rules in order to optimise the area coverage.

Dynamic Programming (DP) is used to optimise the sampling location for an AUV team in [45]. The coordination between the vehicles assumes they need to move within constrained maximum radius from each other to maintain communication. Each vehicle's waypoint is selected to be at maximum distance from a previous sampling point, that also allows communication with adjacent vehicles. The algorithm gives a solution only for the next waypoint and does not explore possibilities with an optimal end-goal in mind. This makes the approach greedy in regards to the final mission objective, which is to uniformly reduce the uncertainty in the sampling area below a predefined threshold.

The heavy communication assumption of [45] was relaxed in [46] by utilising simple local rules when connection is lost. This developed an emergent behaviour, with simulation results approximating the optimal solution. One assumption the authors reported was that the environment did not change in time during the mission. As in [45], the vehicles planned only for the next waypoint, which made them aim for regions at the centre of the sampling area; consequently, the approach reports an uneven exploration. This could be avoided by planning for multiple waypoints ahead in time.

In a data-driven approach, the data can be used for higher level decision making, including changing the vehicle's task or mission phase, or optimising its future action plan. In [47], the author suggests that a data-driven approach that allows the vehicle to learn and update its internal environment model, rather than just to react to it, makes an intelligent AUV. Table 2.3 summarises some of the model-driven and data-driven mission planning differences.

Table 2.3: Comparison between model-driven and data-driven mission planning

Model-driven mission planning	Data-driven mission planning
offline	online
predictable outcome	unpredictable outcome
higher operator's trust	lower operator's trust
inflexible	flexible or adaptive
computationally cheap	computationally expensive

### 2.2.3 Planning and Learning

The importance of assimilating data and updating the agent's internal model was demonstrated by the examples in the previous sections. In a more general sense, this is the agent's ability to learn - augmenting the agent's knowledge of the environment with gaining new experience. Novel methods coming from machine learning can help an agent adapt to new circumstances by detecting and inferring patterns.

In [48], a review on the applicability of machine learning techniques based on neural networks for underwater vehicles' controls is presented. It suggests that combining the control and learning in the same loop gives the fastest adaptability rate to changing environment. The neural network learns by comparing the feedback from the output and the desired value. Some of the major issues the authors raise involve inconclusive results on whether neural networks can adapt to online learning and if such a controller will be sufficiently reliable for dealing with critical systems.

Some of the recent examples of AUV technology advancement is in terms of classification using deep learning [49] and fault detection [50]. These approaches are still in the research phase and have not been implemented for online use on the vehicle; however, they show promise.

Another machine learning approach is Reinforcement learning (RL), where sequential and non-independent and identically distributed data can be used as a training input. The idea of intelligent mission planning and autonomy developed in previous sections fits well with the four elements of an RL agent: 1) agent interacting with the environment, 2) seeking to achieve a goal, 3) while operating in uncertain environment and 4) considering the long term effects of its actions. The framework to model a RL agent is MDP, already introduced as the decision-theoretic planning framework. Learning and planning are the two central problems in RL. RL can be seen as solving the prediction (learning or updating the internal model from data) and control (defining an optimal action policy, can be greedy or with foresight) problem for a robot. In RL literature, the term model-based appears with the opposite meaning of the way to which it is referred in underwater robotics literature. The model-based method means

Table 2.4: Comparison between learning and planning steps in Reinforcement learning (RL)

Learning	Planning
executed first	executed second
solves the prediction problem	solves the control problem
estimation	optimal control
input: sensory data	output: policy

the agent has a model, or can build one, from which it can infer how the environment will behave. Such model-based planning approaches overlap with high-level and deliberative control architecture methods. However, building, or learning, these models is sometimes prohibitive as it requires a large amount of data. A recent survey on using RL in robotics is available in [51].

There are a limited number of reported cases in the literature where RL and neural networks are used for underwater robotics. In [52], a feasibility study is presented using a RL algorithm for high-level AUV control. It uses a neural network to represent the mapping between states and actions online. This new representation removes the need to build an ontology, as some of the planning methods from the previous sections suggest. The paper reports promising results for simple control tasks, such as tracking and recovery. Another RL approach is applied as a low-level reactive mechanism in Component Oriented Layer-based Architecture for Autonomy (COLA2) [53], a control architecture designed for the SPARUS II AUV. The vehicle has been trained in simulation to follow a cable and then this ability is tested in a real underwater environment.

The difference between data-driven and RL approaches is that often data-driven approaches accept a myopic or greedy control solution, while the RL prediction and control aim for a non-myopic strategy, or having foresight. This is the idea that some actions might not lead to a direct increase in the reward for the next action but could be good for the long run, or for the overall goal, that will be reached after taking multiple actions. The RL approach and the MDP framework provide an efficient solution for the planning problem, even when learning is not possible. For the application of mission planning for AUVs this is a promising alternative to existing algorithms. Recent research on adaptive sampling for multiple AUVs applies a decision-theoretic approach using an MDP to represent the control problem [54], [55]. The authors use a modified value iteration solution algorithm and report computation time suitable for real-time execution. The reason for this result with a non-myopic approach is that they discretise the state and action space very coarsely. However, these are some of the first simulation results using this approach successfully for underwater planning applications. Overall, RL is a two-step process, involving learning and planning (summarised in Figure 2.4).



Figure 2.2: Autonomous robots designs: top left: Sonobot surface vehicle, top right: Delivery car, bottom left: Mars Curiosity rover, bottom right: Raven drone<sup>1</sup>

## 2.3 Marine Autonomy Applications and Development

This section follows up on the advancement of different methods and considers the effect of the environment on how transferable the solutions are between robotic domains. A review of the available AUV mission planning capabilities is given so the gaps in the field can be identified.

### 2.3.1 Terrestrial, Space and Marine Environment

The different directions that terrestrial, marine and space robotics have taken can give an insight of the development pace and how applicable technology advancement is within the robotic fields. While general AI methods, such as planning and handling uncertainty are relevant for all autonomous systems, they are approached differently given the environment, application and operational constraints. Figure 2.2 shows a small sample of recent autonomous robot development.

---

<sup>1</sup>Photo credit: Sonobot - EvoLogics GmbH; Delivery car - Starship Technologies; Curiosity - NASA; Raven - United States Air Force

Space can be characterised as a vacuum medium, where the full EM spectrum propagates well. The issue is the distance or the occlusion of line of sight, which can delay communication and limit the available bandwidth. One unique issue is the presence of radiation, which limits the processing power as all electronics need to be radiation hardened. The cost, long development cycles and risk associated with space robotics can sometimes lead to slow adoption of new materials and algorithms.

In contrast to the space environment, the oceans are highly dynamic. The sea state, characterising wind speed and wave height, can be prohibitive for deployment of robots so even launching a mission is very dependent on meteorological day-to-day changes. The seawater chemical and physical properties, as well as dynamics, are varying temporally and spatially. There are changes in pressure, temperature, salinity, light and chemical composition which needs to be accommodated in the platform and sensor design. The poor propagation of light was already introduced, making communication, localisation and navigation non-trivial. Similar to a space environment, where the satellite can be very far and the connection unreliable and delayed, an underwater vehicle experiences similar problems due to using acoustics. Shallow water operations create even bigger communication challenges by the formation of shadow zones - an area where the speed of sound changes and signals are not able to pass through it. Multipath effect due to signal scattering from sea bottom and surface make the acoustic channel hard to model and predict. The seas are also very noisy, due to surface waves and wind, shipping and marine mammal activity, and seismic effects.

In terms of terrestrial and aerial robotics, there are common solutions and known limits for navigation, localisation and communication. This is the reason algorithms from these environments are sometimes not directly applicable and transferable to the underwater domain, despite the overlap in applications, such as collaborative robotics, networks and swarms.

### 2.3.2 Applications and Advances for Multi-Vehicle Systems

Recent advances in marine robotics include robust platform design, reducing size and cost of vehicles, docking, vision and perception algorithms, fault detection and building better sensory and motion models, navigation, control, to name a few. A lot of effort has also been put into studying and developing networked solutions and collaborative robotics. This section gives an overview of current problems and recent successes in the underwater autonomy field with a focus on multi-platform progress. An outlook of research challenges in underwater sensor networks is given in [21], [56], recent selected field experiments results are shown in [57], and a defence application perspective is presented in [24].

Using multiple platforms in a mission brings problems such as networking, coordination, collaboration, and integration between heterogeneous platforms.

## Coordination and Networking

Recent trial results from GREX, an international project between industry and academia, enabling coordination between multiple AUVs via acoustic link are reported in [58]. What was specific to the tests was that the two available vehicles were very different from each other. The most significant difference being that the minimum operational speed of one of the vehicles exceeded the maximum speed of the other. This called for scenario adjustments and coordination behaviour triggered by predefined threshold of positioning error. During the trial, the acoustic communication between the vehicles was the most serious challenge. It was reported that the successful data transmission never exceeded 60%. This was mainly due to the very limited bandwidth of 480 bps which required the usage of highly efficient and lossy compression schemes. Since there were only two vehicles in the formation, redundancy was possible and the overall coordination scheme worked successfully. However, the experiments confirmed the expected communications issues even with seemingly simple scenarios and goals.

In contrast, simulation results present advanced coordination algorithms, such as formation control for pipeline tracking [59], adaptive sampling application using dynamic space partitions [60], as well as optimising the sampling location based on other vehicles in the network [45], [46]. Coordination between platforms requires communication. [59] assumes only local communication between the leader and the follower is possible, and restricts the vehicles' position within a geometric configuration. [60] assumes that requesting a surfacing event from a vehicle is always possible and that this will get through to all other platforms in the network. [45] has a strong assumption that the network can handle and distribute all the messages for each vehicle announcing its next sampling point. [46] reduces this assumption by relying on sporadic links to be enough to coordinate between vehicles, and if a vehicle is left disconnected for a period, it can adapt its behaviour based on the last information it has about the network.

The coordination requirements based on network and communication restrictions are a central question addressed in sensor networks and swarm approaches. Common concerns are scaling up a network by adding and removing nodes, or designing connectivity rules and geometry constraints for vehicle movements. The complexity arises from allowing all platforms to access the common medium and the desire for them to share the limited bandwidth fairly. Since the underwater channel is half-

duplex, where simultaneous communication is not possible, most widespread networking techniques do not apply. Having very few nodes transferring messages between each other allows scheduling the transmissions, or using Time Division Multiple Access (TDMA). Increasing the number of participants in the network means less time for transmissions and strict time synchronisation. The complexity of this approach rapidly increases with the size of the network or when the traffic is aperiodic.

There are comparative studies to evaluate how suitable common underwater protocols are for sensor networks. In [61], variations of three protocols are examined: ALOHA-CS, based on simple collision avoidance mechanism to sense the channel before data packets are transmitted, MACA, relying on Request To Send (RTS) and Clear To Send (CTS) handshake between sender and receiver to allow communication, and Floor Acquisition Multiple Access (FAMA), using RTS/CTS of length based on the propagation delay of the channel. Simulation results showed the simple ALOHA-CS had the highest throughput. Furthermore, using an acknowledgement packet to verify correctly received packets reduces throughput for all protocols. Finally, the energy evaluation points out that the sink for ALOHA-CS uses the least energy as it does not send CTS; however, the nodes have very high energy consumption.

There are also studies considering the effects a moving platform has on the MAC scheme and how this can be utilised. One such example is UW-Polling [62] where a simulation of an AUV collecting data from static nodes is testing an auction-based algorithm. The vehicle sends a broadcast to discover its neighbours and assigns a priority list to sequence data uploads from the nodes. The comparison with random access protocols, such as ALOHA, showed that UW-Polling has better throughput and packet delivery ratio.

The issue with underwater protocol design is that most of the work remains in simulation and is not further tested in real environment. Although there are simulators that account for some of the challenges of the channel [14], such as sound speed profiling and path loss, there are practical issues [63] which can be tested and evaluated only in water, such as delay variance [64] and multipath. The other issue with simulations is that there is no available protocol standard to follow, which makes it hard to evaluate system performance. The first underwater communication standard, JANUS, was only recently accepted [65].

One example of deployed underwater sensor network published the results from trials performed in Portland Harbour, United Kingdom in December 2004 [66]. It evaluates two MAC schemes, TDMA and Random Access (based on a RTS packet sent to the destination and after receiving a CTS message it transmits the required data until a confirmation is sent back from the destination node), and a

routing ad hoc protocol (assumed that most nodes would be stationary and therefore a reactive routing approach was implemented: discovering the routes upon packet request). The trial set up included six nodes and water depth of 15 m. The primary issues noted were signal reflections from the harbour walls, ambient noise and local shipping activities. It was reported that close by ships have a major impact. Another significant disruption was caused by AUV transmission tests performed in close proximity. The results from the TDMA scheme, using 5 stationary and 1 drifting node, evaluated the effect of the drifting node on the network. Overall, no packets were lost and no collisions experienced. This was due to the low traffic rate and number of access points. The TDMA protocol had a guard interval that accounted for the drifting node of up to 3000 m and therefore this did not affect the operation. Secondly, trials were performed using multi-hop transmissions with Random Access and ad hoc routing. The purpose was to asses collision between packets and test route rediscovery and node replacement. The presence of noise interference and the inability to perform broadcast efficiently and react to the need to establish new routes, resulted in large latency penalties. Another heavy delay was induced by the presence of asymmetrical link, which resulted in long delays to the establishment of routes. During the trial, the only time packets were dropped, reducing the channel efficiency to 90 %, was with the presence of noise interference over a long period. This led to inaccessible channel links and the need for rerouting, which resulted in the highest reported latency. Overall, the trial demonstrates a successful ad hoc underwater network; however, it assumes only static (or slowly drifting) nodes and, at times, experienced high delays. Finally, low traffic was assumed, and a busier channel will result in higher message collisions and latency. Both MAC and routing are challenging tasks underwater, resulting in significantly poorer performance compared to terrestrial setup.

## Cooperation

There are coordination requirements of communication and networking where nodes need to synchronise their behaviour. If, in addition, the robots need to cooperate in terms of sharing and distributing tasks between each other during mission execution, this will add additional message overhead and robustness requirements.

One recent example of a European project building a cooperative network of vehicles that can reallocate its resource is MORPH [4]. This is inspired by the need to develop a tool able to map underwater environments with attention to complex structures. Since most high-resolution sensors operate in close proximity to the sea bottom, the architecture involves several nodes linked wirelessly into a supra-vehicle that allows coordinated reallocation of parts of the robot to selected areas. The main advantage of such a topology is that it allows for optimal sensor placement based on surface restrictions

or required illumination angle. To enable reliable links between all vehicles, there is a node designated for relaying communication messages between the AUVs and a surface node responsible for keeping a human operator in the loop.

Another recent cooperative robotics project is Cognitive Autonomous Diving Budd (CADDY) [67]. It aims at developing robot companions for divers. The project involves an AUV that understands commands from the diver and monitors the diver's health status so it can intervene in case of an emergency. There is also a surface node that can relay messages to a ground station and can be used for reliable navigation of the underwater units.

In [68], the authors look into the benefit of using cooperative AUVs for search applications. The addressed issues include the ability to reconfigure nodes online, heterogeneous vehicle and sensor capabilities, and introducing multiple objectives. The simulation results show that multiple vehicles are able to adapt to the mission objectives and environment changes. However, the conclusion is that it is difficult to evaluate how effective a cooperative mission is, as there are multiple simultaneous objectives that can form the scoring result. This evaluation difficulty is due to the tradeoff that needs to be incorporated in the cost-function, such as search time, battery efficiency, number of vehicles, and detection threshold. The weight of these competing goals is defined by the overall mission objective.

Additional challenges come from integrating heterogeneous platforms and multi-environment robotic networks, such as AUVs, surface vessels, air drones and ground vehicles. In [69], a demonstration of coordination and operational challenges is given. Another example is the yearly student competition, where teams show the capabilities of their multi-domain robotic networks emergency response in an unstructured environment [70].

## Resource and Sensor Management

The ability of agents to cooperate under resource or operational constraints brings the issue of resource and sensor management. Sensor management refers to the control of an agile sensor according to the objectives and limitations in a dynamic environment. Resource management is a more broad term focusing on the best way to allocate resources. In [24], the authors consider the data-driven autonomy approach can be treated as an instance of sensor management since the AUV, equipped with multiple sensors, can be seen as a single complex mobile sensor. In [71], the authors consider the sensor management problem to be a decision process and cast it as an instance of optimal planning, since the optimal sensor configuration can be achieved if an action policy is defined. Finally, in [72], the resource management problem is described as the ability to allocate resources optimally by reconfiguring a

system, while sensor management is limited to the resource allocation problem particularly for sensors. It is clear that resource and sensor management overlap and methods developed in both fields can be applicable for multi-AUV cooperation.

The work available in both sensor and resource management fields is applicable to many AUV mission planning applications as it provides extensive theoretical background and methods for selecting the objective function. The objective function is the way to express and compare different solutions. It can be based on task-specific or information theoretic objectives. The first, for instance, can measure the detection probability of a target or the probability of false alarm. Some authors prefer this as it can be easier to put together multiple objectives across many tasks, such as the case of multi-function radar [73]. The second objective function takes into account information gain (measures difference between probability distributions), and has the advantage of being invariant to the data, making it more robust to dynamic changes [71].

Despite the choice of an objective function, optimal resource and sensor management relies on long term planning. Greedy actions in the present might be costly and suboptimal for the overall objective of the mission. This is why the decision theoretic approach is often applied to such problems and using a MDP or POMDP framework for representation [71].

## Task Allocation

Enabling cooperation requires multiple nodes to follow a mechanism to distribute tasks between each other. In robotics, this problem is commonly referred to as Multi-Robot Task Allocation (MRTA) [3]. This gives grounds for successful network coordination and cooperation.

An early implementation of MRTA follows an approach built upon motivating certain behaviours that allow robots to act only if they have the ability to fulfil a task [74]. ALLIANCE is a behaviour-based, distributed architecture aimed at fault tolerant cooperation. The robots rely on communicating regularly the tasks they are performing within the network. This early work does not look into task negotiation and the decision for one robot to take over another robot's task is based on a function defining the level of behaviour motivation for each node.

The economically-inspired concept of market-based control involves features, such as decentralised decision-making, interaction between agents and a limited resource that need to be allocated. The notable element of this approach is that it can control a complex system globally relying on simple interaction and limited information. MAC for UWSN and task allocation for multi-AUV networks are

problems that can benefit from adapting solutions based on this framework. In the Coordination and Networking subsection, the UW-Polling MAC protocol that is described applies auction-like mechanism for the AUV data collection. It enables a fair sharing of the medium between the multiple static nodes in the network.

The classical notion of an auction involves buying and selling goods based on a set of exchange rules. Currently, this idea is especially applicable to fields like robotics and AI, where multiple nodes follow a mechanism to distribute tasks between each other. An example of auction designed for a robotic system will typically include the following steps:

- A task is announced by an auctioneer
- Calculating utility (sometimes referred to as a fitness function) of each robot. This combines all the characteristics into a single value that could be compared to other robots' fitness for completing the task
- Bidding - robots announce their fitness
- Winner announcement by the auctioneer

Recently, the concept of auctions has been applied to resource management in a multifunction radar system to continuously allocate sensor parameters for optimising target tracking [73]. The simulation results provide evidence of performance improvement over rule-based approaches. The framework allows multiple agents to interact locally, ensuring global optimal resource utilisation.

A major disadvantage of such an economically-inspired approach is that it relies on robust communication between the nodes. This is particularly inefficient in the underwater environment where each auction could be repeated multiple times until all the messages manage to get through. Despite the assumption of heavy reliance on communication between nodes, there are some examples that suggest market-based control does not add a large overhead. In [75], an experiment based on a first-price one-round auction (the highest bidder wins over one round of bidding) suggests that the message exchange involves packets of small average size (bytes), medium-sized network (8 participants) and frequent data exchange ( $\approx 20$  messages per minute). A challenge for modern underwater acoustic modems will be the congestion of the network as signals travel much slower in water and a single message sent in the wrong time slot could damage the whole auction process. Repeating all transmissions would be required.

There is available work utilising market-based control in the underwater environment. One example of

a framework offering a solution to mine hunting for the search and classify phases is DEMiR-CF [76], [77]. This addresses the MRTA in two stages: firstly the area is divided optimally between all available vehicles and, secondly in case of changes to the topology, the tasks are reassigned according to the result of auctions. This approach was tested via simulation and successful results reported. However, some simplifications deserve attention. Firstly, the framework assumes 100% success in executing the task of classification - the crawler either recognises a mine or, after a time-out, it is considered a false alarm. Another simplification is the communication overhead analysis — this is made by comparing the performance of the proposed approach by degrading the channel availability by considering 25, 50, 75 and 100 % message loss. This analysis does not account for the use of an environmental model as an input, nor does it evaluate the amount of data transfer required.

Another recent example for applying a market-based approach when dealing with an underwater cooperative AUV for MCM is presented in [78]. The author considers the high probability of a message loss due to bad channel conditions or large distance and therefore a vehicle's inability to participate in the auction process. To reduce the communication load, the approach assumes that each node has knowledge of the network topology. A decision to accept or deny a task is then calculated based on a utility function with input of distance to target, location of other vehicles and intercepted communication from neighbouring nodes. Each vehicle broadcasts its results. The disadvantage of this method is that it relies on accurate and updated topology information and same tasks can be accepted by multiple vehicles if this condition is not satisfied.

An auction procedure is implemented in simulation for a high-level mission planner in [79]. This work involves multiple platforms that divide the available tasks between each other based on the distance to the target and time to completion. Although successful results are reported, the author calls for a mission planner that has such an auction approach as an option rather than as default. This is consistent with the varying quality of the communication channel, which at times can prohibit such heavy message transfer.

### 2.3.3 AUV Development in Defence and Surveillance

After considering the general intelligent multi-agent design drives, such as choice of control architecture, planning and learning prerequisites, environment dynamics, communication and networking limitations, choice of objective function and task allocations, this section looks into how this is put together for defence and surveillance applications and, more specifically, for mine countermeasures.

Underwater surveillance refers to the task of monitoring an area for threats. In the case of Anti-Submarine Warfare (ASW) [80], this is looking for submarines, which are moving targets and the mission requires tracking, once a detection is present. Harbour protection uses sonar for protecting infrastructure from underwater attacks [81]. Mine countermeasures relies on searching an area for static objects [82].

ASW is usually performed using a passive sonar sensor; however, recent work reports using an active sonar source and multiple passive receivers on AUVs [80]. The paper evaluates the feasibility of using a cooperative robotic network within ASW operational requirements in simulation and trials. A more recent study looks into an adaptive non-myopic tracking capability for ASW using AUVs [83]. The biggest challenge is the computational load of the look-ahead algorithm and the low processing power on the vehicle. The algorithm defines the AUV control in real time and was tested in simulation and demonstrated in a trial. The authors expect online planning algorithms significantly to outperform missions with predefined tracks.

The Centre for Maritime Research and Experimentation report on Autonomy Experimentation from 2014 [84] describes some of the current objectives and capabilities in mine countermeasures. In terms of autonomous search for mines, the AUV is able to adapt the track width of the lawnmower pattern, and it can detect and adapt to sand ripples. The exercises focused on target revisit strategies in terms of target selection (how likely is it that the detected target is a mine) and observation. In addition to search patterns, the mine reacquisition and identification, and the performance of the mission, in terms of data quality and path planning, were studied. There was no multi-vehicle autonomous trial.

The requirements and advantages of using multiple AUVs for MCM operations was analysed in [85]. The authors advocate the need for a distributed knowledge representation in the form of an ontology, a data-driven adaptive decision making framework, and an automatic target recognition capability. The report suggests promising results from a cooperative mission in a MCM setting; however, no details are provided about how the required mission elements were implemented.

Another MCM scenario is defined in [86]. The work uses an ontology-based representation planner and builds an executive layer (mid-level control), DELPHIS, aiming at reduced replanning from the top deliberative module. The suggested mission controller is responsible for vehicle communication, goal selection and action prediction of the other vehicles (in case of communication loss). The authors compare their approach to a stoplight system - AUVs meeting at prescribed meeting points. They consider this approach state of the art in multi-AUV coordination (year 2010) and limited in terms of being too demanding on the user for predefining the synchronised rendezvous points or high demand

on communication, if they are made dynamic. The evaluation metrics to compare the stoplight and DELPHIS are mission speed (how quickly the mission is executed), accuracy (how many goals are missed or made redundant) and target acquisition. The two scenarios the approaches were tested on were pipeline tracking and MCM. It is not clear how the stoplight meeting point location and frequency are defined in the test case. It is reported that the DELPHIS approach outperforms the stoplight, however in multiple cases the stoplight reaches efficiency above 100% (despite the fact that the authors themselves designed the performance metrics). The communication in the DELPHIS system relies on broadcast without acknowledgements. The authors claim that, despite collisions, the messages reach their destination, however no formal proof of data or simulation was provided, neither were the limits of length, frequency and number of messages and vehicles that the medium would support were shown.

### 2.3.4 Summary of Common Assumptions

This section summarises the common assumptions and open questions found in the literature of marine autonomy and robotics.

- Communication: often quoted as a constraint due to variable channel and low bandwidth [21]. It is a bottleneck for most multi-AUV applications. Sometimes authors assume that keeping the vehicles in close proximity would be enough to relay messages [45], others introduce contingencies in case vehicles lose connection [46], [86]. There are cases in which authors assume small sporadic messages can be handled by the network and reach all nodes [60].
- Networking: closely related to the communication problems, networking becomes an important issue when it comes to shared Medium Access Control (MAC) when many nodes are deployed, or when messages need to reach distant nodes over multiple hops.
- Mission objectives: defining metrics to evaluate mission success is hard as many objectives can be taken into account [68]. It is common practice for authors to select an application and set the boundaries and limitations of the system. This makes it difficult to compare methods as no common benchmark or set of goals is followed. A similar issue comes from the fact that most trials are performed using different platforms with varying capabilities.
- State of the art in Mine Countermeasures (MCM): relating to the previous point, there is no widely-adopted benchmark for the application of MCM. Common assumptions include uniformly distributed targets [68], predefined classification boundaries [76] and multi-phase missions [24].

- Processing power: platforms have different processing power capabilities; however, due to limited battery resources on the vehicles, it is usually low. Most authors take this into account when designing computation-heavy algorithms by reporting the complexity or running time. However, very few trials are available testing such algorithms in a real environment. This is especially applicable to non-myopic algorithms, known for their high computation load [52], [83].
- Vehicle type: capabilities between vehicles can differ substantially and this makes it critical to use a suitable task allocation algorithm (which brings the communication issue), or adapt the mission operations accordingly [58].
- Vehicle's internal model: the assumption that a vehicle can keep track of the other nodes in the network and can operate based on this information, in case it gets disconnected, is a way of reducing the communication reliance [46], [78].

## 2.4 Synchronous Rendezvous

As seen in this chapter, multi-vehicle applications often adapt to the communication-constrained environment by relying on methods such as formation flying or setting path constraints. Often throughput and channel variations are acknowledged but unrealistic assumptions about communication and networking limitations are common. This thesis proposes the Synchronous Rendezvous (SR) method as a solution for utilising multiple Autonomous Underwater Vehicle (AUV) for Mine Countermeasures (MCM). It defines dynamically allocated meeting points for the vehicles so they can exchange mission and status information in a robust way. The SR points adapt to the workload presented by detected targets in a reactive manner (Chapter 3) and then it is extended to an optimal non-myopic planning strategy (Chapter 4). The SR allows for an operator to monitor the progress of the mission execution at predictable and comparatively regular intervals, which is critical for defence applications using autonomous assets. This can also be extended to periodically collecting the available data by a surface vessel.

The classical rendezvous problem deals with defining a strategy for a multi-agent system that maximises the probability of agents meeting each other [87]. The Synchronous Rendezvous (SR) method uses some of the principles from the original rendezvous problem, such as communication-constrained environment, limited sensor range, collective behaviours, and not having the meeting location and time scripted [88]. For example, [87] suggests an algorithm where the robots select a point without knowing if the rest of the nodes will also consider the same spot as likely for a meeting. There is even the

possibility that they will not recognise each other once they meet, due to noisy sensors. The reported experimental results showed that the robots chose four unsuccessful locations on average before they met. The SR method differs from the classical rendezvous problem as it does not require for the nodes to ‘discover’ the meeting point, but instead to ‘agree’ on it. The location and time to meet are unknown at the start of the mission, but the agents schedule them dynamically. Thus, the SR method differs from previous work, and techniques suggested for the original rendezvous problem are not explored further.

In robotics, using a meeting point, or a rendezvous, for multi-vehicle coordination in a communication-restricted environment is not uncommon. In [89], a survey for autonomous path planning considers the ability for the vehicles to rendezvous as a requirement for successful multi-vehicle coordination. This approach is often used for air drone cooperative missions [90], [91].

Synchronous Rendezvous (SR) was already mentioned in this chapter when reviewing [86]. The paper refers to the stoplight method and considers it as the benchmark scenario. The authors assume a predefined sequence of points and discard dynamic allocation of points as impractical due to communication constraints. The paper does not provide any sources or further studies into testing this assumption. This thesis develops the SR method with the assumption that close proximity of the nodes allows robust communication. Even if increased message exchange is needed to negotiate a new meeting point time and location, this can be done using a simple TDMA scheme, with short travelling distance for messages and the allocated time slots. This assumption was tested and proven feasible in simulation environment using MOOS-IvP in Chapter 5. The SR provides an approach to remove, or reduce, the communication and networking problem for a multi-AUV mission.

The expected disadvantage of the method is the resource loss of the system due to vehicles meeting, instead of doing mission-related tasks. This effect is studied in a MCM scenario, using parameter sweep to define acceptable loss boundaries. The analysis is available in Chapter 3. The resource loss is compared to the gain of utilising the AUV network in a flexible way, having a robust means of communication and coordination. Chapter 3 suggests a reactive heuristic-based algorithm that adapts the available resource and defines mission objectives and constraints in an MCM context.

Once the limitations of the SR are presented in Chapter 3 and suitable scenario parameters selected, the myopic reactive SR algorithm is updated into an intelligent non-myopic and optimal solution of the MCM scenario. The issue of computation load is central for this approach and the method introduced in Chapter 4 suggests approximations of the state and action space in order to handle the MCM scenario mission requirements. The results give a proof of concept for building an intelligent mission

planner for multi-AUV MCM mission.

# Chapter 3

## Synchronous Rendezvous Method

Given the communication limitations presented in the previous chapters, and the need for the Autonomous Underwater Vehicle (AUV)<sup>1</sup> network to exchange information, this thesis develops the Synchronous Rendezvous (SR) method — bringing the vehicles together throughout the mission by autonomously scheduling meeting points. Outside of the SRs, the agents can assume complete lack of communication. Thus, severe channel disruptions and variability are no longer an issue for the system operation.

In order to examine closely the benefits and losses the SR brings, and which parameters are most relevant in optimising the scheduling, a specific scenario is developed. The aim of this work is providing an alternative method for mine hunting using AUVs. Therefore, the introduced scenario follows exclusively a Mine Countermeasures (MCM) application and introduces its constraints.

Following the MCM phases described in Section 1.3.3, a distinction should be made between the aspects of the mine hunting procedure that are relevant to the SR method, and those which are considered out of scope for this work. The focus of the SR is to facilitate cooperation between multiple platforms on a mission planning level. The relevant MCM phases are Search and gathering information for Classification and Identification. This work is not concerned with processing the sonar data (Detection), however it assumes that some online detection capabilities are available on board the underwater platform. Once a detection is made based on the coarse data collected during Search, often more detailed data is required for Classification and Identification. Thus, approaching the contact and collecting additional high-resolution sensory data, as well as different distance and aspect angles of the

---

<sup>1</sup>For the remaining of the thesis, the terms Autonomous Underwater Vehicle (AUV), vehicle, platform, asset, agent and node will be used interchangeably, meaning an AUV in a network performing a MCM mission.

target, facilitate the decision making in the subsequent phases [92], [93]. While the SR method is not involved directly in the Classification or Identification process, it assists the vehicles in collecting the data to enable these phases. This distinction is also mentioned in [82], where two operational tasks are defined: Search-Classify-Map (SCM) - allows faster search while collecting coarse sonar data, and Reacquire-Identify (RI) - high-resolution data collection for final decision making. For this work, the tasks will be referred to as ‘Search’ and ‘RI’. Neutralisation often assumes that either a vehicle, or parts of its payload, are disposable. Often an ROV is preferred for this stage, which also makes it irrelevant for the multi-AUV SR scheduling problem.

This chapter proposes a scenario used throughout the rest of the thesis. It focuses on developing a proof of concept by defining parameters and a benchmark to evaluate the loss and gain from the SR method. The results are assessed based on numerical simulations in MATLAB. The tradeoff the SR method introduces is the ability to retask vehicles dynamically versus the loss of resource due to the vehicles travelling to meet and communicate. This chapter’s method and results are published in [94] and [95].

### 3.1 Synchronous Rendezvous Method Scenario

The simulation scenario is aimed at representing a typical MCM mission - demining a strip of seabed near the shore. Figure 3.1 gives an example of the mission setting and SR method operation.

The area that requires clearing is a rectangle with dimensions  $x$  and  $y$ . The ‘start’ point defines the launching site of the AUVs. On the left graph in Figure 3.1, the search phase is depicted. Three vehicles are moving in a ‘lawnmower’ pattern, with each having a separate search area. The AUVs meet at the SR to share status and data gathered during the search phase.

The graph on the right shows an example of decision making and retasking of the vehicles. At the SR, they have shared locations of potential targets that require further inspection. Based on this information, the vehicles decide to allocate one node for RI task and two nodes to continue searching new area — ‘veh 1’ and ‘veh 2’ have a new search pattern and ‘veh 3’ is assigned with the RI task. The RI vehicle revisits the detections following the shortest path. The time required for this task defines the time of the next SR. The search vehicles adapt their search paths based on their new number, now two instead of three, as well as the new time window — the SR is adaptive to the RI workload. This is the advantage of the SR method — the robotic network can utilise its resources optimally. The vehicles dynamically adapt their tasks based on the new knowledge developed during the search phase.

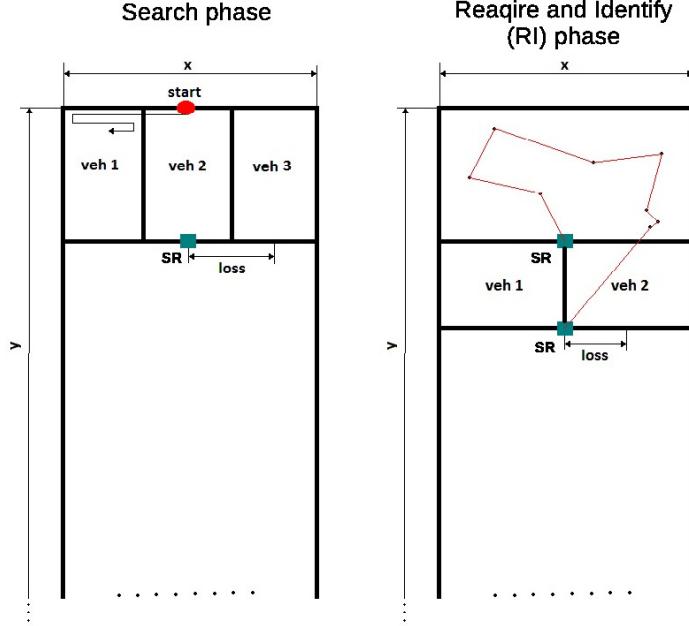


Figure 3.1: Synchronous Rendezvous (SR) approach: Left - search phase, Right - Reacquire-Identify (RI) phase. The area that requires clearing is a rectangle with dimensions  $x$  and  $y$ . The ‘start’ point defines the launching site of the AUVs. On the left graph, three vehicles are moving in a lawnmower pattern. Each node has a separate search area. The vehicles meet at the SR to share status and data gathered during the search phase. On the right, is an example of decision making and retasking of the vehicles. At the SR, they share locations of potential targets that require further inspection.

The mission goal defined for this scenario is maximising the search area, while revisiting all contacts. The decisions the vehicles need to make are when and where to meet and how to distribute the search and RI tasks between them for the next SR window. This is expanded further in the chapter.

The SR method removes reliance on preplanned missions and predefined tasks. It also solves the communication issue by bringing the nodes together to enable a reliable communication link. At SR, an operator can monitor the system status and the sensory data can be downloaded and sent to a mission centre at intervals during the mission. However, the SR method brings an inevitable loss to the system - the vehicles need to periodically cease their MCM tasks and travel to the meeting location. The more SRs are scheduled during the mission, the more loss is introduced. Therefore, reducing the number of SRs during the mission is desirable. On the other hand, the system can distribute its resources better if the vehicles have regular chances to meet and adjust future strategy. This tradeoff is analysed by a sensitivity analysis. It evaluates the parameters that affect this loss and makes it possible to define boundaries for the SR time window.

### 3.1.1 Synchronous Rendezvous Loss in Mine Countermeasure Scenario

The Synchronous Rendezvous (SR) loss is defined as the time spent by each vehicle to travel to a meeting point instead of doing a mission-related task. The following equation gives the loss,  $L$  — the fraction of time for the SR interval lost due to travelling to the meeting point:

$$L = \begin{cases} \frac{1}{2}(1 - \frac{1}{n_S}) \frac{x}{v \times t_{SR}}, & \text{if } n_S > 1 \\ \frac{1}{2n_S} \frac{x}{v \times t_{SR}}, & \text{if } n_S = 1 \end{cases} \quad (3.1)$$

where  $x$  is the width of the search area,  $n$  is the number of search vehicles (the loss dependence on number of vehicles is explained further),  $v$  is speed and  $t_{SR}$  the SR time interval (time between two SRs).

The loss,  $L$ , is evaluated per single vehicle, as it is related only to the search task (the RI defines the SR interval and no loss is introduced for this task), and at each SR interval a variable number of vehicles are tasked to search. The loss is also calculated per single SR interval, as opposed to total mission time, as each SR interval varies in time. Therefore, the overall SR loss for the mission duration will be additive, but nevertheless it will be unique for each time interval.

Figure 3.1 gives an example of the loss. During the search phase, the AUV search follows a lawnmower pattern. At the time the vehicle needs to stop searching and aim for the SR, the AUV could be at the far or close corner of its search box, or anywhere in between. The distance from this point to the meeting point will define the resource loss. However, the simulation is not optimised to position the nodes at the near corner. Equation 3.1 is adjusted so the calculation always assumes the distance between the middle point of the platform's search area (on the  $y$  axis, same as where the SR is) and the meeting point. The special case when  $n=1$  is required as otherwise it would result in  $L=0$ . The current penalty for this case resembles the loss of  $n=2$ .

The parameters that affect the loss,  $L$ , are:

1. Number of search vehicles — loss increases by increasing the number of search vehicles, as their search perimeter moves farther from the SR in the middle.
2. Speed,  $v$  — the faster the platforms, the quicker they reach the SR so  $L$  is reduced.
3. Synchronous Rendezvous (SR) time interval,  $t_{SR}$  — defines what fraction of the mission time will be used for search and how much will be needed for travelling to the meeting point.

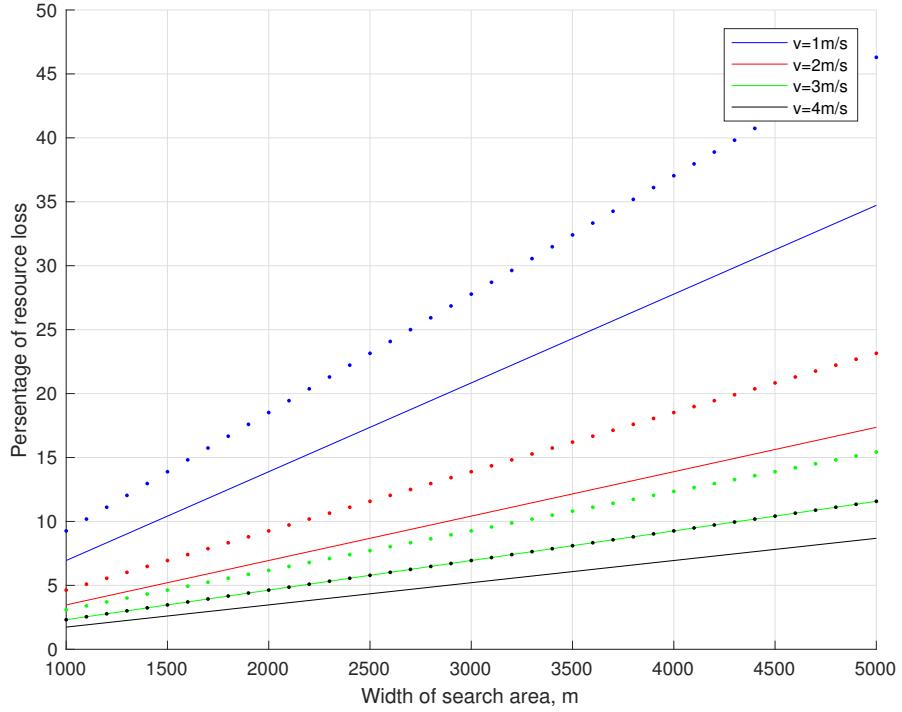


Figure 3.2: Loss calculation for two vehicles (lines) and three vehicles (dots): different speed values and width of search area ( $x$ ); fixed RP time (1 hour), and fixed length of the  $y$  dimension of searched space (5000 m).

Parametrising Equation 3.1 allows to define limits for the SR in order to keep the loss at acceptable boundaries. Figure 3.2 presents the loss for  $n=2$  and  $n=3$ , for different speed values,  $v = [1 : 4]$  m/s, and width of the searched area,  $x = [1000 : 5000]$  m, while the SR time interval,  $t_{SR}$  and the length of the search area,  $y$  were kept constant ( $t_{SR}=1$  hour,  $y=5000$  m).

In order to select suitable boundaries for the SR interval, based on Figure 3.2, the simulation parameters were set to:  $n=3$ ,  $v=2$  m/s and  $x=2500$  m. Then, Figure 3.3 gives the percentage of time loss as a function of the length of the SR interval. It is used as a guidance for the minimum time limit between scheduling SRs. For example, if  $t_{SR} \in [1 : 2]$  hours results in a loss,  $L$ , of 12 to 5 %, respectively. For the simulations in this chapter,  $t_{SR}=1$  hour is selected, however if the loss is considered unacceptable for a specific mission, a longer time window can be selected.

Some of the limitations and assumptions introduced by Equation 3.1, that are valid for the simulations in this chapter, include:

- The system is resource-constrained - not enough battery life to search the whole area and revisit all targets.

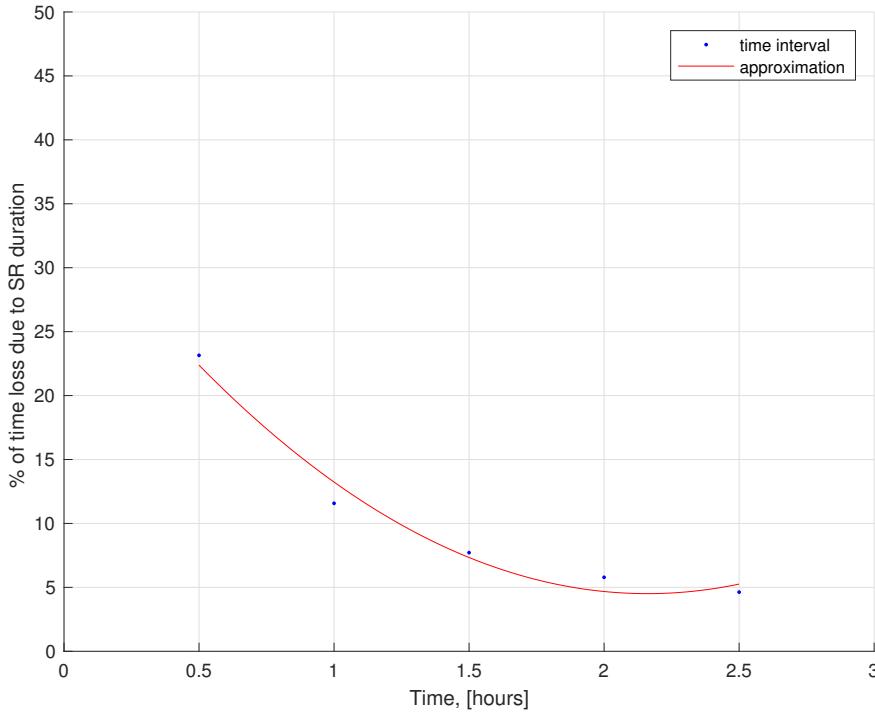


Figure 3.3: Loss of time resource (y axis) vs total time between SR (x axis) – calculation based on Equation 3.1 (parameters used:  $n = 3$ ;  $x = 2500$  m;  $v = 2$  m/s).

- Sequential area search.
- Homogeneous vehicles equipped with sensors for both search and RI.
- Constant speed,  $v=2$  m/s, total number of AUVs used is three, search area size 2500 by 5000 metres, SR interval lower bound  $t_{SR(min)} = 60$  min, based on Figure 3.2, and SR interval upper bound  $t_{SR(max)} = 3t_{SR(min)}$ , to account for the maximum resource of three vehicles in the network.
- Environment effects are ignored (eg. currents).
- All vehicles arrive simultaneously at the SR.
- The scenario used for simulations is the one from Figure 3.1. Given Equation 3.1, the results derived in this chapter are dependent on the choice of scenario parameters.

### 3.1.2 The Baseline Mission

Reconfigurability of the assets during the mission and their ability to adapt to the emerging workload is the basis of the SR approach. Such capability is required as it is difficult to correctly estimate the number of targets a priori, and it is also non-trivial to distribute the AUV time resource efficiently at the start of the mission.

The mission plan for a standard AUV mine hunting mission [85] includes an offline search plan in a lawnmower pattern for coarse sonar imagery collection. Then, the vehicle is recovered, data is downloaded and analysed by an operator. Based on the possible targets flagged by the operator, a reacquisition pattern is assigned for the vehicle to collect additional imagery. The vehicle is launched for a second time. For this simulation, the baseline mission is adapted to a multi-vehicle scenario. In addition, Automatic Target Recognition (ATR) and unlimited communication was assumed. Due to the ATR assumption, the vehicles do not need recovery for their data to be assessed by an operator as this is done automatically on board. Even though unlimited communication is an unfair advantage, compared to the SR mission (where communication is available only at the meeting point), this simulation aims at identifying the benefit of reconfigurability. The baseline mission assumes two vehicles are tasked with search and one vehicle is tasked with RI. The search vehicles communicate any target location to the RI vehicle without restriction. The RI vehicle schedules a path to revisit the targets. While the RI vehicle plans an optimal path between the targets, it is still not utilised optimally, given multiple available vehicles. Either the targets will be too many for a single vehicle and at the end of the mission there will be non-visited targets, or there will be too little targets and the RI vehicle will be idle for some time.

An alternative benchmark scenario that was considered, but not implemented, is using all vehicles for both search and RI - once a target is detected with a side scan sonar, the vehicle can stop the search phase and inspect the contact. The main consideration to have a designated vehicle for search and RI in the selected benchmark is to reduce the disruption of the vehicle track. Data from a sea trial is presented in Chapter 5 and it shows the extent to which a vehicle track can deviate from a desirable position. In addition, the Side-Scan Sonar (SSS) sensor requires a steady platform moving in straight lines to acquire imagery. Frequent disruptions, due to a vehicle diving to observe a contact, brings distortions and additional requirement to reposition the vehicle.

## 3.2 Synchronous Rendezvous Scheduling Algorithm

There are two parameters defining the Synchronous Rendezvous (SR) scheduling mechanism: selecting the *time* and *location* for meeting.

### 3.2.1 Synchronous Rendezvous Time Interval

The next SR time interval is based on the shortest path to travel between all previously detected targets. Therefore, it depends on the number and location of the targets found during the previous SR interval. The target distribution number and location is generated as a uniform distribution. These dependencies are explained further and then the sequence of a mission is shown in Algorithm 1.

First, the time to travel between the known targets,  $t_{RI}$ , is calculated:

$$t_{RI} = f(t_G, t_H) \propto U\{a, b\} \quad (3.2)$$

where  $t_G$  is the time required for a single vehicle to travel the greedy shortest path and revisit all detected Mine-Like Objects (MLOs). The number of the targets is a uniform distribution and multiple averages are tested further in the results section, as no reliable literature source was found to base that assumption on. The location of the targets follows a uniform distribution  $U\{a, b\}$  based on the defined search area,  $A_S$ , from Equation 3.4. The second term in Equation 3.2 is  $t_H$  — the additional time needed for the vehicle to reacquire the target and collect additional high-resolution sensory information. This parameter can be adjusted based on the mission and sensor requirements. In the simulations for this chapter, a constant value is selected,  $t_H = 150$  s per target. Estimating and quantifying the parameters characterising  $t_H$  in non-trivial. A sea trial experiment was designed to study an AUV's performance of the RI task for a single target. The results are described in Chapter 5, however no significant evidence can be derived and the assumption of  $t_H = 150$  s per target is arbitrary for this chapter's simulations.

Once  $t_{RI}$  is known, the next meeting point depends on how many vehicles are committed to the RI task,  $n_{RI}$ . This depends on the lower and upper limit of  $t_{SR}$ :

$$t_{SR} = f(t_{RI}, n_{RI}) \in [t_{SR(min)}, t_{SR(max)}] \quad (3.3)$$

$t_{SR}$  is bounded by Figure 3.3, based on Equation 3.1, on the lower end (for this work,  $t_{SR(min)} = 60$  min), and for the upper limit, a hard boundary is defined by the vehicles' resource limit for the duration of the SR interval ( $3 \times t_{SR(min)} = 180$  min).

### 3.2.2 Synchronous Rendezvous Location

The location of the next SR is based on an estimation of the search vehicles' position after they finish their task. The AUVs' end position can be calculated by the search area they cover:

$$A_S = v \times w \times t_{SR} \times n \quad (3.4)$$

where  $v$  is AUV speed in m/s,  $w$  is sensor swath width (Side-Scan Sonar (SSS), assumed  $w = 100$  m),  $t_{SR}$  is the time until the next meeting point in seconds from Equation 3.3, and  $n$  is the number of search vehicles. Following the assumptions from the previous subsection, all platforms are of the same type and have the same sensors, and all nodes reach the same  $y$  coordinate at time  $t_{SR}$ , since they are assigned with equally sized search task. Based on these assumptions, the SR location is the middle of the search area width,  $x$ , at the  $y$  coordinate, to which the vehicles have advanced following Equation 3.4. Figure 3.1 gives an illustration.

### 3.2.3 Synchronous Rendezvous Algorithm

The algorithm used in this chapter's simulations is summarised in pseudocode in Algorithm 1. The first step is to define the input parameters:

- Line 2 defines the lower bound of the Synchronous Rendezvous (SR) time interval based on the loss,  $L$ , discussed in the previous section. For the simulations,  $t_{SR(min)}=60$  minutes, is assumed.
- Line 3 assigns the first SR time interval the  $t_{SR(min)}$ . This is the only predefined SR in the whole mission.
- Line 4 initialises the target number parameter,  $N_{MLO}$ . At the beginning of the mission, no targets are detected, so  $N_{MLO} = 0$ .
- Line 5 keeps track of the mission time that has passed with  $mission\_time$ . The first SR is predefined and the scheduling procedure starts from there.

---

**Algorithm 1** Dynamic allocation of vehicle tasks and SR scheduling

---

```
1: procedure SCHEDULE SRs
2:    $t_{SR(min)} \leftarrow$  based on limiting the loss,  $L$ , from Figure 3.3 and Equation 3.1
3:    $t_{SR} = t_{SR(min)} \leftarrow$  initialise first SR
4:    $N_{MLO} = 0 \leftarrow$  initialise number of targets
5:    $mission\_time = t_{SR}$ 
6:   while  $mission\_time < B$  do
7:      $n = 3 \leftarrow$  number search vehicles
8:     Calculate search area  $A_S$  using Equation 3.4
9:      $N_{MLO} = N_{MLO} + U\{a, b\}$ 
10:     $t_G \leftarrow$  calculate RI time (Nearest Neighbour)  $\propto A_S$ 
11:     $t_{RI} \leftarrow t_G + N_{MLO} \times t_H$ 
12:    if ( $t_{RI} > t_{SR(max)}$ ) then
13:      break
14:    else
15:      if  $t_{RI} \in [t_{SR(min)}, 2t_{SR(min)}]$  then
16:         $n_{RI} = 1$ 
17:         $t_{SR} = t_{RI}/n_{RI}$ 
18:         $n = n - n_{RI}$ 
19:         $N_{MLO} = 0$ 
20:      else
21:        if  $t_{RI} \in [2t_{SR(min)}, 3t_{SR(min)}]$  then
22:           $n_{RI} = 2$ 
23:           $t_{SR} = t_{RI}/n_{RI}$ 
24:           $n = n - n_{RI}$ 
25:           $N_{MLO} = 0$ 
26:        else
27:           $t_{SR} = t_{SR(min)}$ 
28:           $N_{MLO} \leftarrow$  transfer all targets for next iteration
29:        end if
30:      end if
31:       $mission\_time = +t_{SR}$ 
32:    end if
33:  end while
34: end procedure
```

---

Once all the input is assigned, the SR algorithm is run:

- Line 6 gives the termination condition of the algorithm. The *mission\_time* is additive for each SR cycle (Line 31). The algorithm runs until a battery threshold,  $B$ , is reached. This parameter is based on the battery life of the AUVs and for the simulations,  $B = 7$  hours is assumed. The mission duration parameter depends on the vehicle's capabilities and the operator's choice. Once the *mission\_time* parameter reaches the battery threshold, the end of the mission is triggered and the final SR executed. This means that each simulation will run for different time, as the agents adapt to the uniformly distributed workload, and the SR that triggers the end of the mission is reached at a different time.
- Line 7 assigns the number of search vehicles. The mission is initialised with all three available vehicles performing the search.
- Line 8 calculates the size of the search area, following Equation 3.4 and assumes constant input for  $v = 2 \text{ m/s}$ ,  $x = 2500 \text{ m}$ ,  $y = 5000 \text{ m}$ ,  $w = 100 \text{ m}$ .
- Line 9 generates targets based on a uniform distribution  $U\{a, b\}$ . The simulation tests the effect of distributions with three different average number of targets per  $t_{SR}$  interval. The new targets are added to any targets left from a previous SR cycle. The target information is available to the vehicles at every SR (new targets are generated at every SR).
- Line 10 calculates the shortest path to visit all known targets,  $t_G$ . It is based on a greedy Nearest Neighbour algorithm. The target locations are uniformly distributed within the dimensions of the search area,  $A_S$ .
- Line 11 calculates the time required for the Reacquire-Identify (RI) task,  $t_{RI}$ . It takes into account  $t_G$  and the additional time to acquire high-resolution information for every known target,  $t_H$ . This follows Equation 3.2.

Lines 12 to 32 give the heuristics of the approach. Here, the vehicles decide when to schedule the next SR and how many vehicles to be tasked with search and how many with RI.

- The first If-statement, lines 12 and 13, makes sure the algorithm is not out of bounds - it breaks if the workload is too high. The condition checks the time for RI does not exceed the time resource for the robotic network,  $t_{SR(max)}$ . In the current simulation case, if  $t_{RI}$  requires more than three vehicles to be tasked with RI, then the break would be triggered.
- The first task allocation case is given in Lines 15 to 19 when  $t_{RI}$  is between  $t_{SR(min)}$  and  $2t_{SR(min)}$

(line 15). This means a single vehicle will be able to perform the RI task for all the known targets. The SR time is based on the RI time (Line 17). The search vehicles are reduced by one (Line 18), as one AUV is tasked with RI (Line 16). All known targets are revisited, so the total count is set to zero (Line 19).

- The second case is between Lines 21 and 25. By increasing the number of targets,  $t_{RI}$  grows. When  $t_{RI}$  is between  $2t_{SR(min)}$  and  $3t_{SR(min)}$ , then two vehicles are tasked with RI (Line 22). On Line 23, the time of the next SR is half the time for RI (as two vehicles are dealing with the workload,  $n_{RI} = 2$ ). The number of search vehicles is reduced to one (Line 24). The target number is assigned to zero as they are all revisited (Line 25).
- Lines 27 and 28 give the last task allocation case, when the RI time is below the minimum limit that is assigned for  $t_{SR(min)}$  on Line 2. In this case, the simulation assumes it is better to ignore the few detected targets and postpone gathering additional data on them until the next SR cycle. This way, the RI vehicle is never idle, but instead is retasked to do a search. In this case, the number of vehicles doing the search stays as three,  $n = 3$ , as assigned on Line 7, and no vehicles are tasked with RI. The next SR is scheduled based on the requirement set on Line 2. The number of targets here is not set to zero, but instead carries to the next SR, where the newly discovered targets are added to the old ones. The disadvantage is that the targets will be further away so the RI vehicle will have to travel additional distance to reach them.

### 3.2.4 Target Number Distribution Model

The SR algorithm adapts the system resources to the targets found during the mission. This property does not require a prior estimation of the number and distribution of the minefield.

The assumption of no prior knowledge of the minefield distribution makes the uniform distribution  $U\{a, b\}$  the most suitable model. The parameters  $a$  and  $b$  give the minimum and maximum value of the distribution's elements. Let  $N$  be the number of elements for the set  $S = \{a, b\}$ . The probability of observing each element,  $n$ , of  $S$  is equally likely:

$$P(n) = 1/N \quad (3.5)$$

In order to see the effect of the target number on the SR algorithm, the simulations were repeated for three different sets with increasing number of elements:  $U_1\{0, 10\}$ ,  $U_2\{0, 20\}$  and  $U_3\{0, 30\}$ . The

choice of the three target distribution models is somewhat arbitrary, as it is not based on literature or experiments. Since this is hard to determine, the target intervals were selected in order to test the SR's performance with changing the variance of the workload. This choice represents low target number, medium target number and cluttered environment. The mean  $((a + b)/2)$  gives an expectation of the average number of targets that are simulated for each SR interval. Further in the text, the simulations corresponding to each distribution are referred as SR(5), SR(10) and SR(15), respectively.

### 3.3 Results and Analysis

#### 3.3.1 Simulation Results

Figure 3.4 shows three plots, each corresponding to a different target model - on top is  $U_1$  with average of 5 targets per SR interval, middle is  $U_2$  with average of 10 and bottom is  $U_3$  with average of 15. Each graph gives a scatter plot of the search area (y axis) as a function of the mission time (x axis). Each dot is a full mission simulation - the blue dots are where SR is applied, the red dots are where the benchmark is applied. It is important to note that the mission time for each simulation is defined by the SR algorithm (Line 6) and varies for each repetition. This mission time limit, as well as the same target model number and locations, are also applied to the base case. Using the same parameters input, each simulation creates a pair of SR and benchmark with a comparable output. Each of the three plots on Figure 3.4 has 1000 full mission pair simulations.

The benchmark simulations (red) follow a linear relationship - the search area increases when more mission time is available. On the other hand, the SR simulations (black) can be seen above the red line, showing higher resource utilisation compared to the base case, as well as below the red line, resulting in worse performance. On the top plot, the majority of the SR simulations show better performance. This is due to the RI vehicle from the base case being idle for a large part of the mission as there are not enough targets to fill its schedule. The SR algorithm retasks the vehicle from RI to search so no idle time is present in the system. With increasing the number of targets in the middle and bottom plots, the SR method has a reduced searched area gain compared to the base case. This is due to more resource (sometimes more than one vehicle) being used for the RI task compared to searching a new area. Since the mission goal is not only to gain search area, but also to reallocate all detections, some of the benchmark simulations fail, as the RI resource is not sufficient. This is not visible on this figure, but is analysed further in this chapter.

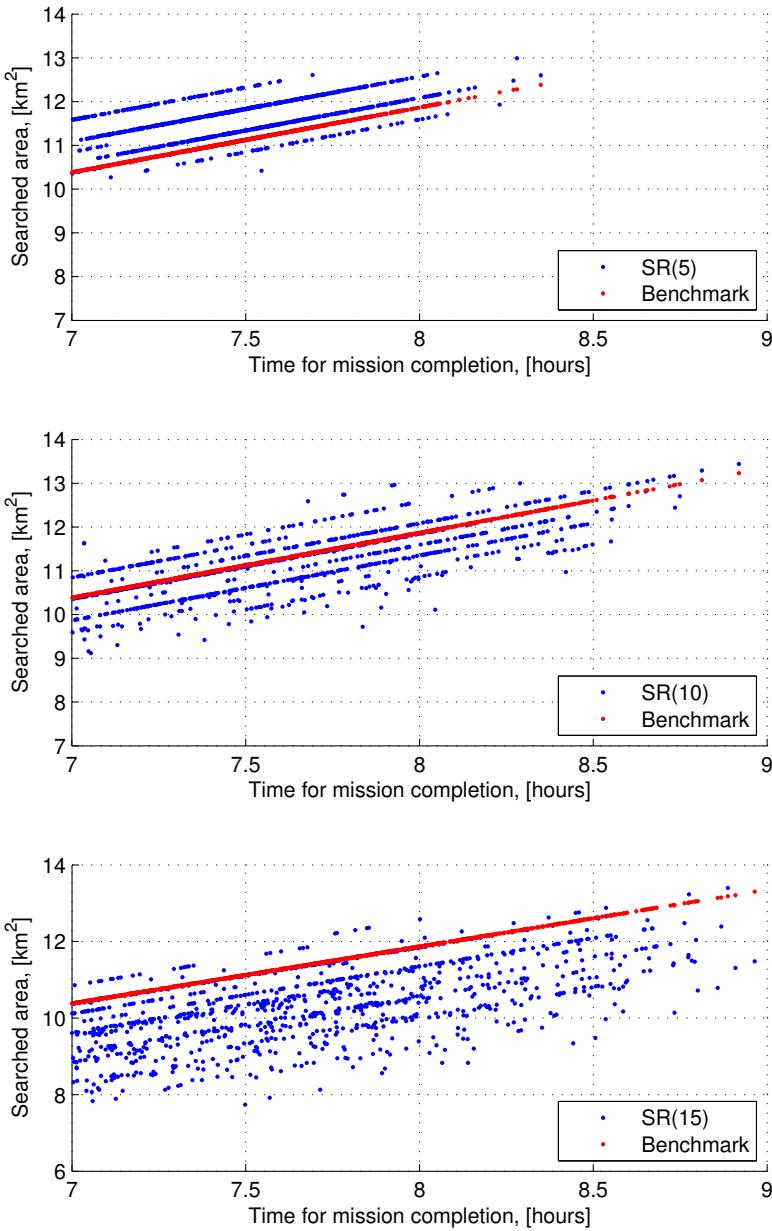


Figure 3.4: Comparing the search area gain (y axis) for the Synchronous Rendezvous (SR) method and the benchmark scenario given variable mission time (x axis) and different number of targets (increasing the number of targets with each model from the top plot towards the bottom one.) The higher on the y axis the result, the better the approach performance. On the top plot, on average, the SR method is outperforming the benchmark. Increasing the number of targets allows for the benchmark to gain more area and to outperform the SR method (bottom plot).

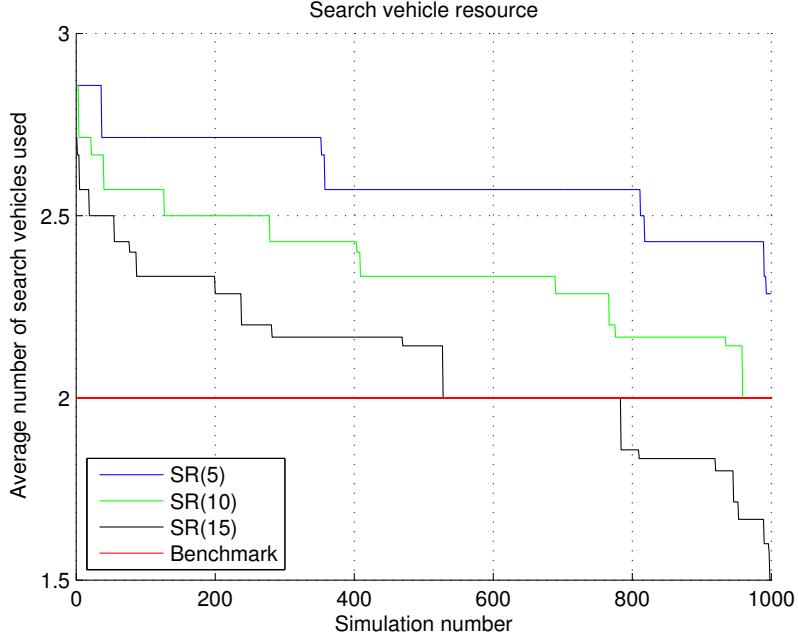


Figure 3.5: Vehicle task allocation - average number of search vehicles (y axis) per mission (x axis) normalised over the number of SRs in the particular mission. The benchmark is set to a constant of two vehicles for all missions as the vehicles' tasks are fixed. The SR simulations corresponding to the three target models show the dependency of task allocation in response to workload.

### 3.3.2 Vehicle Task Allocation

The search area gain for the SR method is variable and is related to how the search and RI tasks are divided between the vehicles. Figure 3.5 shows how many vehicles are tasked with search during a missions. Since this number can change at every SR, the search vehicles are normalised over the number of SR points scheduled during the mission. The red line, with a constant  $y = 2$ , is the base case - since vehicle tasks are predefined, two vehicles are doing a search in all simulation instances, regardless of the mission circumstances. For the SR case, the search vehicle number varies at every SR window (multiple during the mission duration), and it is dependent on the target model - if more detections are found, more vehicles will be tasked with RI, hence the search average will drop. This relation explains why the black line (SR(15)), corresponding to the most cluttered environment and has the lowest search vehicle average within the SR models. It also explains the lower search area gain performance in the bottom graph of Figure 3.4. The blue line (SR(5)) shows most of the vehicles were doing search since a small number of targets required the RI resource. This relates to the top plot in Figure 3.4, where the majority of the blue SR dots have moved above the red benchmark dots, leading to a better search area gain performance, despite the resource loss due to platforms meeting at the SR.

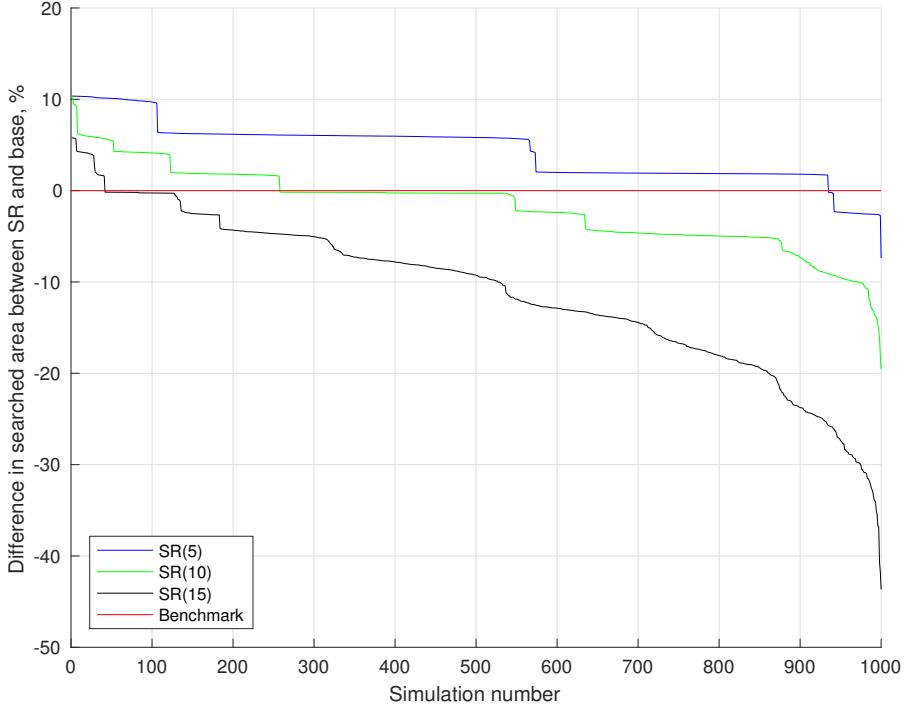


Figure 3.6: Search area comparison

### 3.3.3 Search Resource

After seeing the raw data in a scatter plot in Figure 3.4, and the task allocation of the vehicles in Figure 3.5, the SR performance can be evaluated based on the search area gain parameter. Figure 3.6 shows the difference in percentage between the base case and SR search area gain. All results are normalised to the benchmark, which is the red line at  $y = 0$ . Results above the red line are outperforming the benchmark. The blue line, SR(5), shows that for the majority of the simulation, and despite the loss due to meeting, the SR method has a search gain between 2 to 10 % for more than 95% of the simulations. This gain reduces significantly when increasing the number of targets, and the black line (SR(15)) is below the red benchmark for the majority of the simulations, leading to reduced search area gain performance. Figure 3.6 visualisation emphasises the discrepancy in overall area gain between the SR and base case pairs. It shows that the search area gain is higher for the SR when low number of targets is present and lower for cluttered environment.

### 3.3.4 Reacquire-Identify (RI) Resource

The Reacquire-Identify (RI) resource is the other performance metric on which the SR method is evaluated. Figure 3.7 shows how much RI resource is spent for the SR method and the base case as percentage of the total mission time. The three plots from top to bottom show results for different target distribution models with ascending average. All results were normalised to the overall mission time (based on battery limit,  $B$ ) for each simulation - the green line in the plots shows this time limit. Below it, the RI task will be completed, above it, there will not be enough resource in the system to accomplish the RI task with a single vehicle. The blue line is the SR implementation and the red line is the benchmark.

One important point is that the benchmark always uses less RI resource compared to the SR method. This is due to the RI vehicle's route being defined at the meeting point, so additional time is spent to travel back to the target locations. On the other hand, the base case schedules its target waypoints immediately, as no communication loss is introduced.

The RI resource is not sufficient if the line goes above  $y = 0$  (green line). This is most evident in the bottom plot, where many targets are simulated, and almost 60% of the simulations are lacking RI resource to complete the mission successfully. While this is true for the benchmark, the SR method is adaptive and retasks the nodes so more than one vehicle can perform RI in case the number of targets requires it. At the end of the mission, all contacts are identified for the SR simulations, despite the blue line being above zero in Figure 3.7. This flexibility gives an advantage to the SR approach that can surpass the overall meeting time loss and the reduced total search area.

One disadvantage of the SR method is that the detections are revisited after a meeting point. As a result, the contacts from the final SR are not revisited in this chapter's numerical simulations, as the algorithm terminates once the mission time reaches the selected threshold,  $B$ . However, in terms of evaluating the performance of the SR method, the analysis for both the SR method and the base case disregard the contacts found during the last SR cycle. This simulation oversight and purely reactive behaviour is revisited in Chapter 4, where the optimal SR-MDP algorithm takes into account the remaining time of the mission and plans the vehicles' actions in a way that all detections are revisited by the termination of the mission. A simpler solution for a reactive approach, such as the heuristic SR algorithm, is to task all vehicles with search and immediately perform an RI task once a target is detected during the final SR time interval. This is the approach described as an alternative benchmark in Section 3.1.2. This, however, does not change the evaluation of the method and is not included in the simulations.

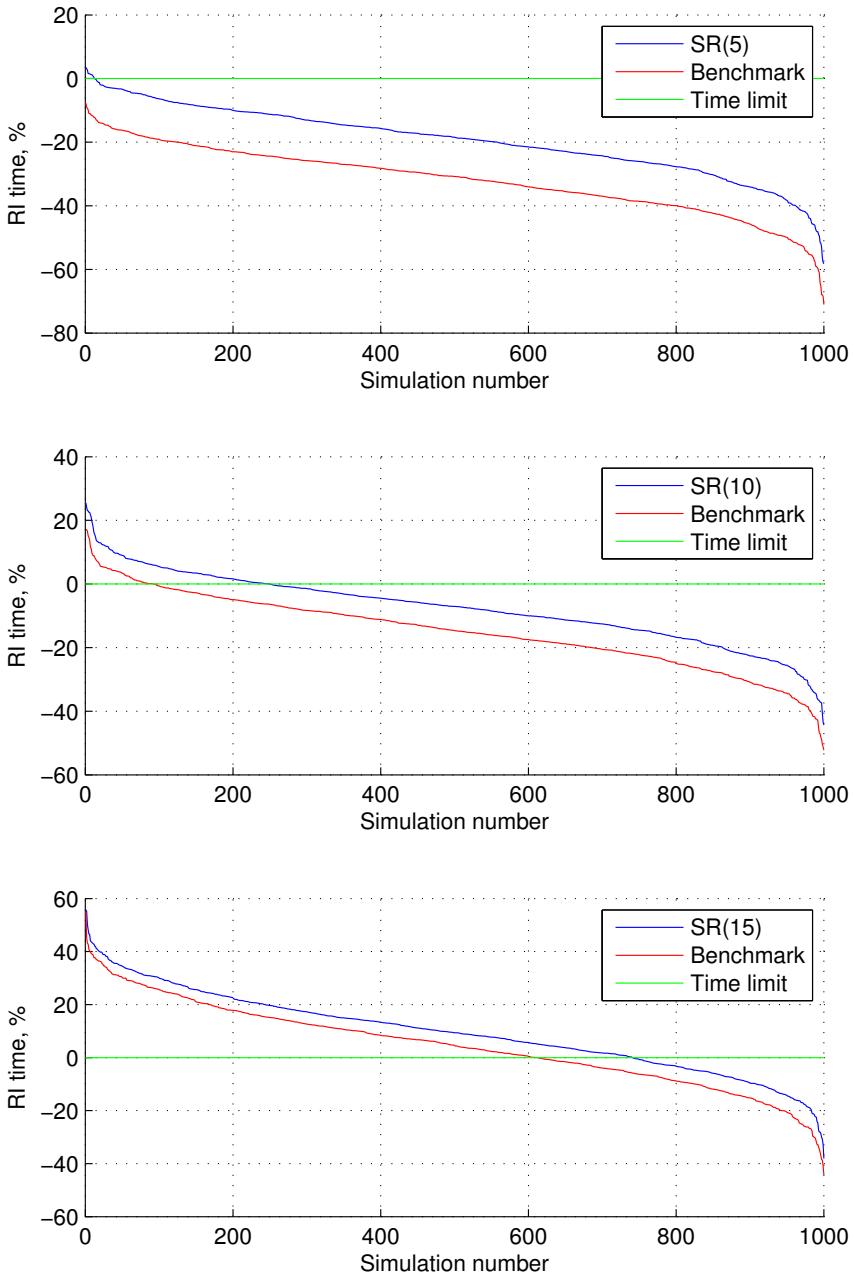


Figure 3.7: Reacquire-Identify (RI) resource allocation comparison between the Synchronous Rendezvous (SR) and benchmark methods. The RI analysis shows the ability of each method to satisfy the condition that all targets need to be revisited. The green line shows the battery life limit of a vehicle. If the simulation result is above the time limit, this would be a mission failure. Since the SR method can retask the vehicles during the mission, it can swap search resource for RI in order to achieve a successful mission at a cost of reduced area size. The benchmark method has set vehicle tasks and cannot adapt to the workload, resulting in multiple failed missions.

### 3.3.5 Time and Search Area Loss as a Function of Number of Targets

In addition to the search and RI resources, analysing the overall search time and search area losses gives an insight of the SR method's performance. This is closely related to understanding how the SR time interval depends on the amount of targets the vehicles encounter. Finally, the analysis concludes by identifying when the SR method is more suitable, compared to the base case, and when it is not efficient.

Figure 3.8 gives the percentage of time resource loss for the system. The top two and the bottom left plots represent each of the target models, with 1000 simulation repetitions. They compare the SR (blue) and the base case (red) missions. The bottom right graph shows a joint plot that compares the SR loss for the three target model instances by plotting them together. There are two interesting trends shown: first, by increasing the number of targets, the base case loss changes its curve shape, and secondly, the SR time loss increases by reducing the targets (joint plot).

In the top left plot, where the least number of targets were simulated (5 on average per SR window), the benchmark curve loss increases monotonically. This is due to the RI vehicle being idle, as it cannot be reassigned to another task dynamically. The SR method adapts the network's resource to the emerging workload, so the loss is only due to the vehicles meeting. In the top right and bottom left graphs (SR(10) and SR(15)), the benchmark loss reduces for the first 10% and 60% of the simulations, respectively, reaches zero loss, and then increases. This pattern is due to the loss coming from two different sources - the left-hand side, before reaching zero, the loss is again due to the vehicle being idle. A loss of zero means all three vehicles are utilised optimally. The right-hand side, or where the benchmark loss slope grows, the loss is due to the RI resource not being sufficient to cover all detected targets. This trend was already mentioned in the previous section and observed in Figure 3.7. These are two very distinct ways to measure loss. The first case, when a vehicle is idle, means the system is being inefficient. The second case, when the targets are not revisited, results in a mission failure, as defined by the scenario mission objectives. Using the base case inflexible task allocation approach, the vehicles are unable to adapt to the emerging workload, and it is unlikely for a predefined mission to reach its optimal resource utilisation.

The bottom right plot shows the SR time loss and how it changes in relation to the target model. After analysing the search resource, the intuition might be that the loss increases with increasing the target number. However, for the SR method simulations, the search and RI resource complement each other and this is not considered a loss for the system. The loss comes from the time spent on the vehicles travelling to the meeting points during the mission. The way the SR interval is defined in Equation

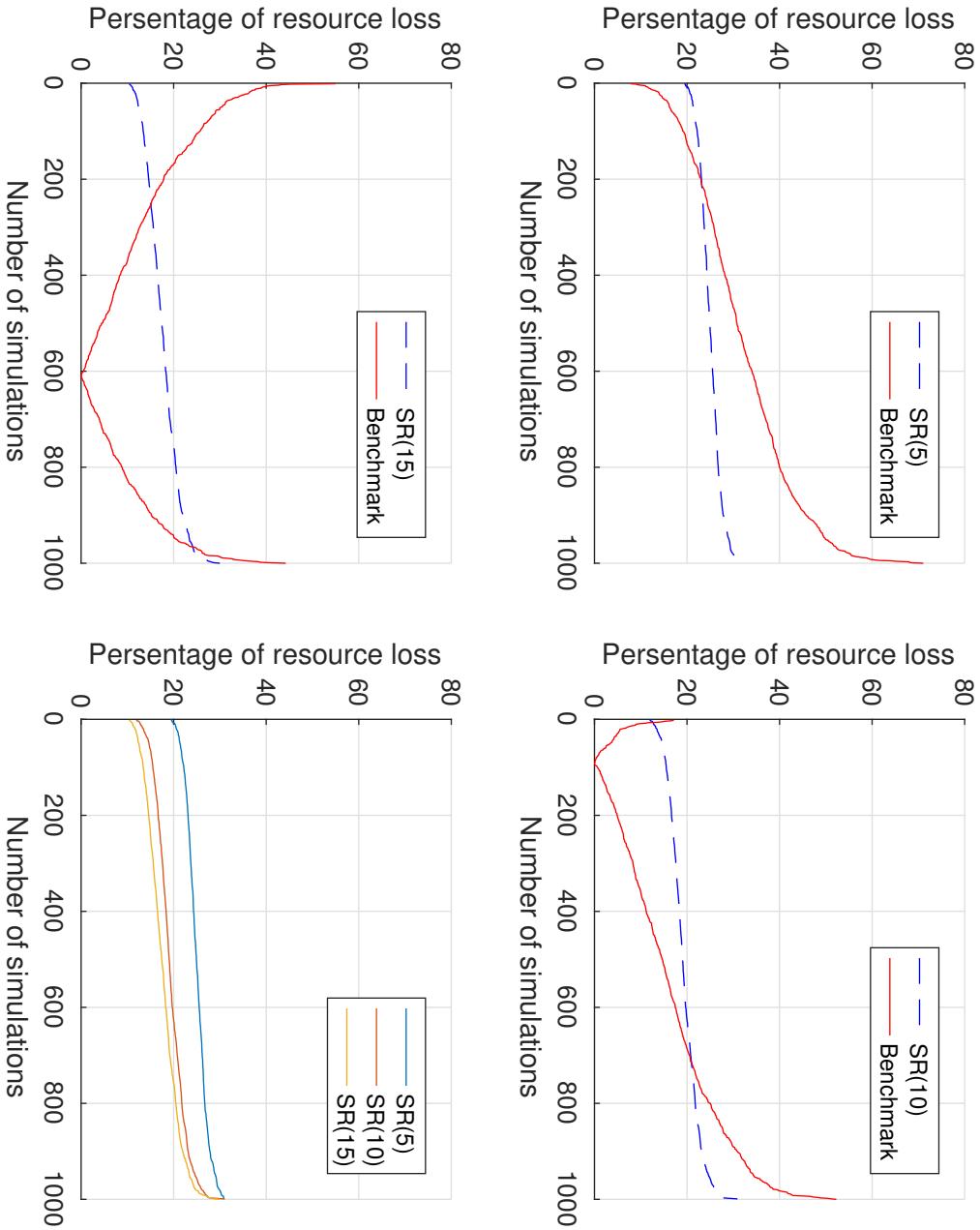


Figure 3.8: Percentage of resource loss (y axis) for a number of repeated simulations (x axis). Blue dashed line shows the SR loss. Red continuous line shows the base case loss. First three plots show the loss for different number of simulated targets - 5, 10, 15 on average. The last plot gives the SR loss from the previous graphs together so the loss can be compared as a function of changing the number of targets.

3.3, and implemented in Algorithm 1, shows that, if there are a small number of targets, the SR will be set to the minimum — every 60 minutes for this MCM scenario. If there are more targets, the SR interval is adapted based on the time to RI - the window will be larger. This leads to scheduling less SR points throughout the mission, hence less overall loss. Figure 3.9 gives a histogram with the number of SRs per mission (iterations). The blue bar corresponds to SR(5) target model, red bar is SR(10) and yellow bar is the SR(15). For all target models, the variation of scheduled number of SRs is 5, 6 or 7 (considering the minimum interval  $t_{SR(min)}$  of 60 minutes and the overall mission time limit of 420 minutes + last SR time interval for the studied scenario). The y axis of the histogram gives the number of simulations resulting with either of the SR counts per mission. The blue bar peaks at 7 SRs scheduled per mission, and corresponds to the target model with least average number of targets, SR(5), being applied. This leads to shorter SR intervals and more of them being scheduled during the mission, and hence increasing the overall time loss in Figure 3.8. On the other hand, the SR(15) target model, simulating cluttered environment, presented by the yellow bars on the histogram, shows it has the highest number of simulations resulting with the least number of meeting points — 5. This drives the overall loss in Figure 3.8 slightly lower compared to the SR(10) model. Intuitively, the less SRs are scheduled during the mission, the less loss is brought to the system. Detecting more targets brings the SR interval up, and reducing the number of meetings.

This intuition is further illustrated in Figures 3.10 and 3.11. Figure 3.10 gives the time loss (y axis) as a function of number of targets (x axis), generated per SR interval. The results following SR(5) target distribution are in blue, in black are simulations based on SR(10), and the green ones follow SR(15). Figure 3.11 uses the same colours for the corresponding simulations and target models, showing how the average SR time interval (y axis) changes as a function of the number of targets (x axis). The results on the figures are sorted in an ascending order in respect to the time loss. Both graphs show the relationship that increasing the number of targets makes the SR time interval to grow, thus scheduling less SRs per mission and reducing the overall time loss.

### 3.3.6 Evaluation — Synchronous Rendezvous (SR) or Benchmark

Figure 3.8 shows that the benchmark reaches zero loss in some simulation instances. Therefore, it can be expected that the SR method is inferior in some cases, since it always brings loss to the system due the vehicles dropping their mission related tasks and aim for the meetings points multiple times during the mission.

Figure 3.12 gives the SR method (blue) and benchmark (black) loss (y axis) as a function of number

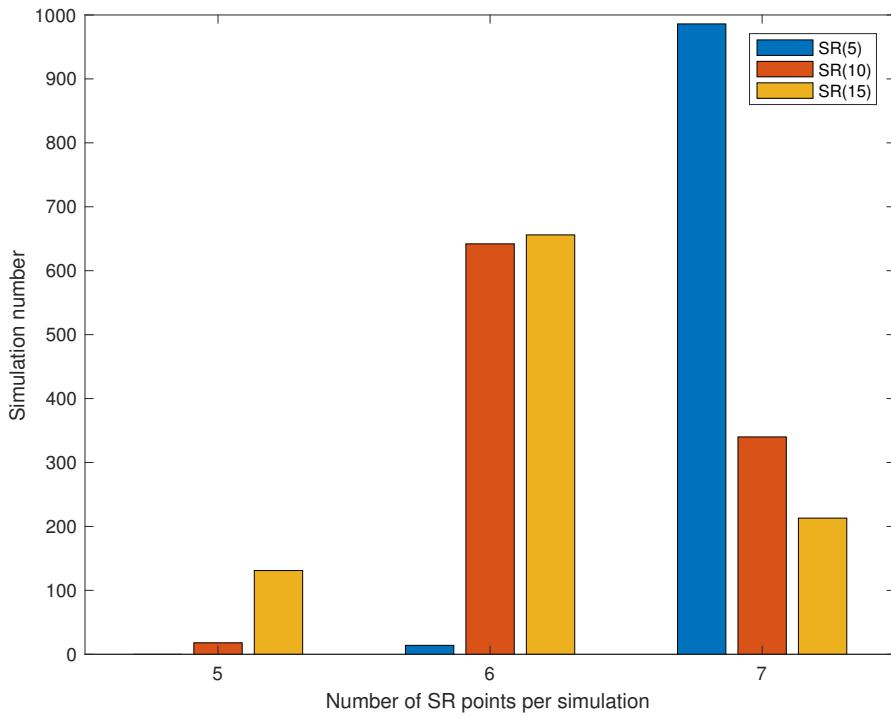


Figure 3.9: Histogram showing number of Synchronous Rendezvous (SRs) scheduled per mission ( $x$  axis) for all the simulations ( $y$  axis). The three bars correspond to simulations with 5,10 and 15 simulated targets on average per SR respectively.

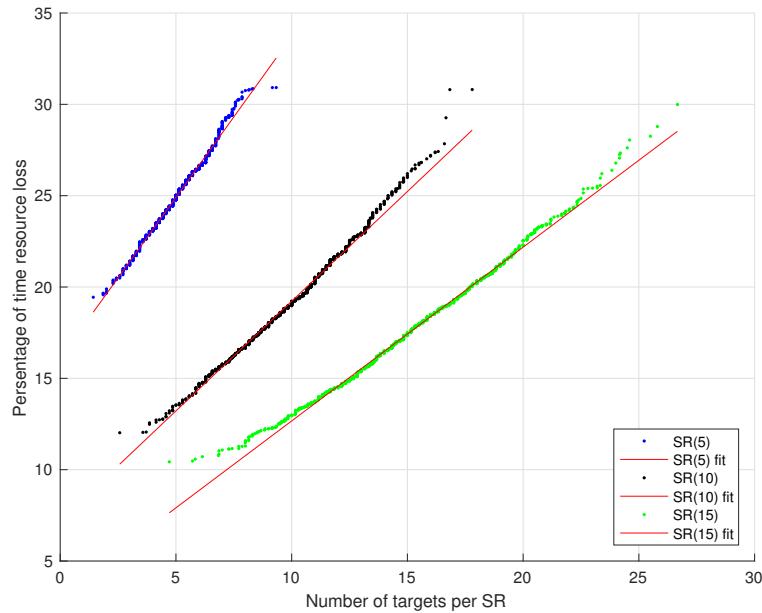


Figure 3.10: Resource loss due to time spent on Synchronous Rendezvous (SR) in relation to the number of targets. Results are sorted based on increasing loss and are colour coded based on the used target model.

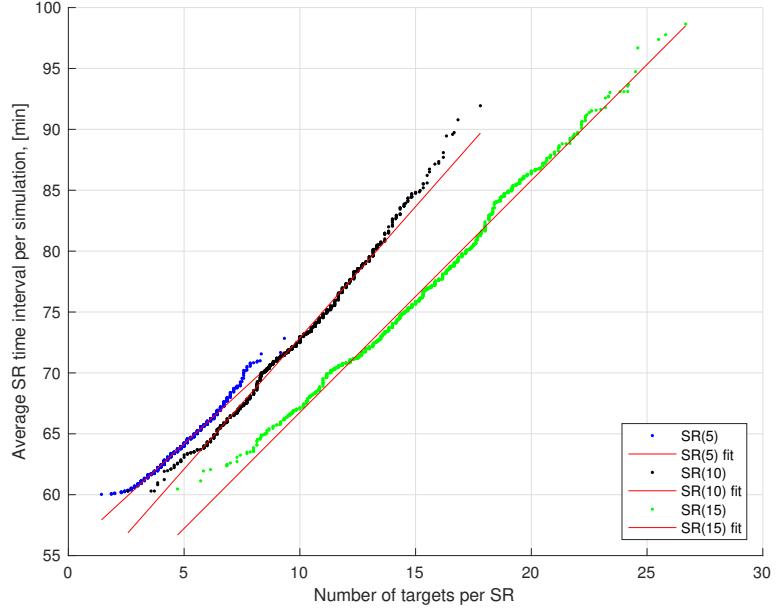


Figure 3.11: Average Synchronous Rendezvous (SR) time interval as a function of number of targets simulated per SR

of targets. It brings together all 3000 simulations that have been presented separately in this section, 1000 simulations for each target model. If a curve is fitted through the resulting SR and base case unsorted scatters, an evaluation can be made in which target regions the SR method outperforms the benchmark and where it is inferior. Figure 3.12 shows that the SR method has a better performance for both low and high target load. If low number of targets per mission is expected, between 0 and 7 on average per SR window, or cluttered environment, with 19 and more targets per SR window, then the SR method outperforms the benchmark. Otherwise, for a target distribution with an average of 8 to 18 per SR window, the base case should be preferred.

However, as already mentioned in Figure 3.8, the benchmark loss comes from two different sources — vehicle being idle or mission failure. Figure 3.13 gives a better illustration of the loss, setting the simulations resulting in mission failure for the benchmark, at 100 % loss. The blue are the SR simulations, the same as in Figure 3.12. The base case scatters are divided into two groups. Going through the list of all 3000 simulations sequentially, arranged by putting the SR(5), SR(10) and SR(15) results back-to-back, the division threshold is when the first mission failure appears. The black in the figure are simulations where no failures are observed. They are 1000 in total. The green are a mix of successful missions and failures. A decreasing loss trend is observed in the green scatters when the target number grows, until a point is reached where the RI resource is not sufficient and the mission fails — this is when a maximum loss of 100 % is given. Since the target number and location are

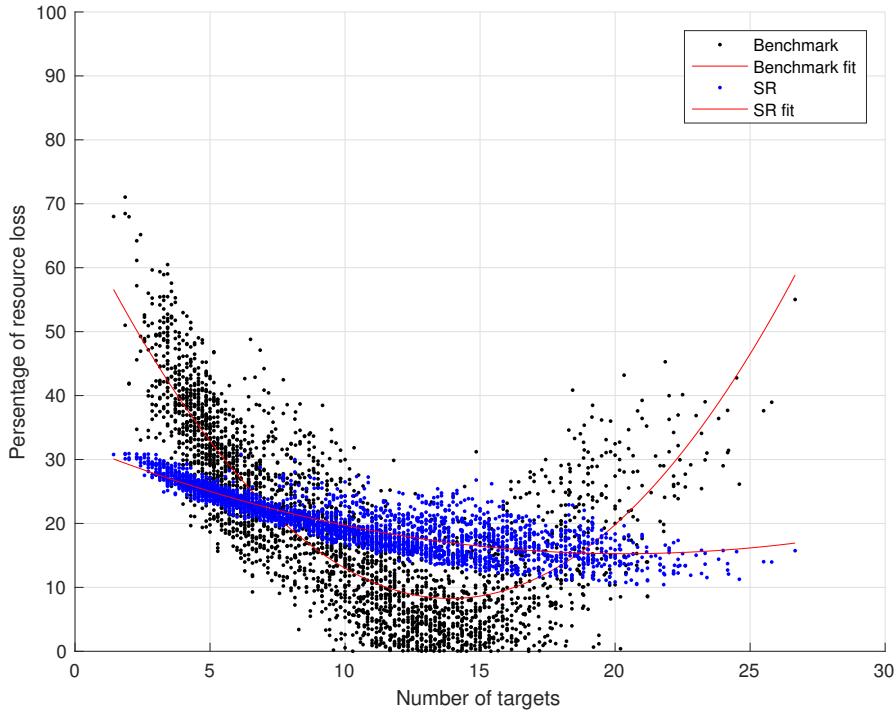


Figure 3.12: Comparison for the time resource loss between the Synchronous Rendezvous (SR) method and the benchmark. No adjustment to address the benchmark loss source.

uniformly distributed, there is no linear relationship how to identify this tipping point between success and failure.

Therefore, in order to analyse the SR and benchmark results, different functions must be used to approximate their behaviour. The SR distribution is fitted with a second order polynomial. The black distribution, the benchmark with no failed mission simulations, is fitted with a first polynomial line. Despite the high variance, the trend is sufficient to observe the trend in the performance.

The green scatters, where failed and successful missions are observed, are treated as a binary classification problem — failure or success. If the base case mission is successful in this region, the result will likely outperform the SR. In contrast, if the base case mission fails, this is a situation where the SR is the preferred method. This leads to a binary representation of the outcomes from this distribution. All the missions failures were already set to loss of 100% in Figure 3.13. In Figure 3.14, all the successes are set to zero loss. Now a logistic regression can be implemented. This statistical method is useful when there are two possible outcomes and it models the relationship between the dependent variables (input) and the binary outcome (output).

The way the green data points are classified follows the binary logistic regression based on the logit

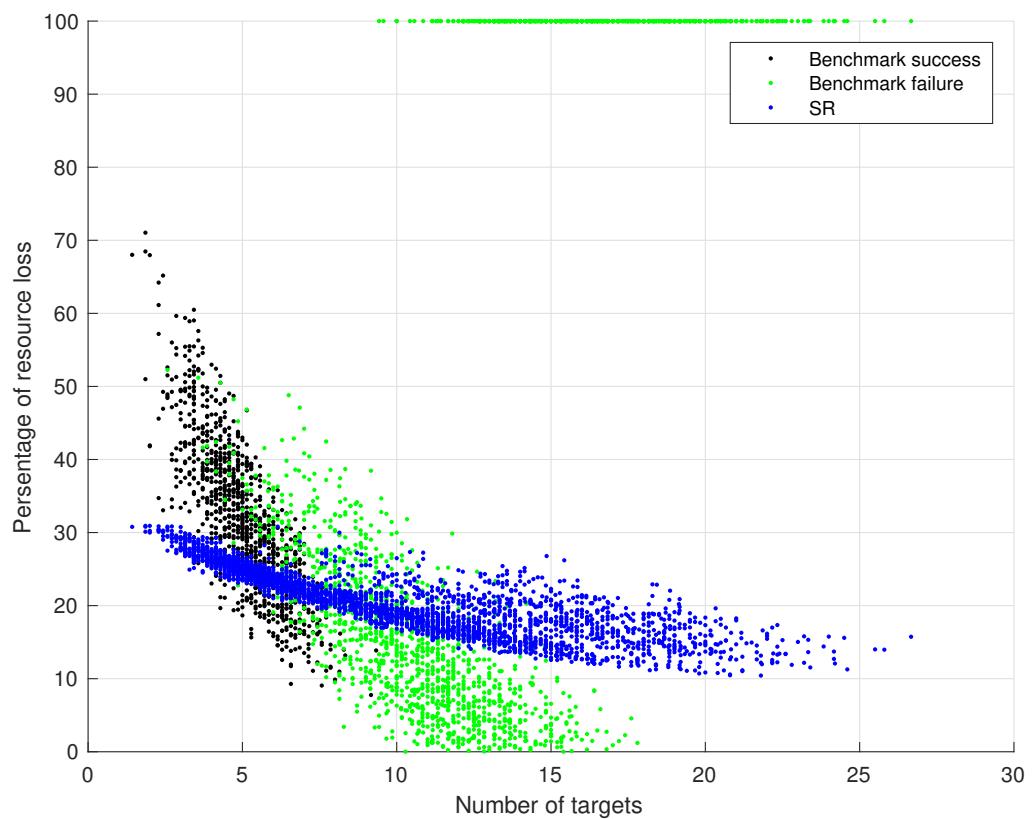


Figure 3.13: Comparison for the time resource loss between the Synchronous Rendezvous (SR) method and the benchmark. The mission failure for the benchmark case is accounted for by setting the resulting simulations to 100 % loss.

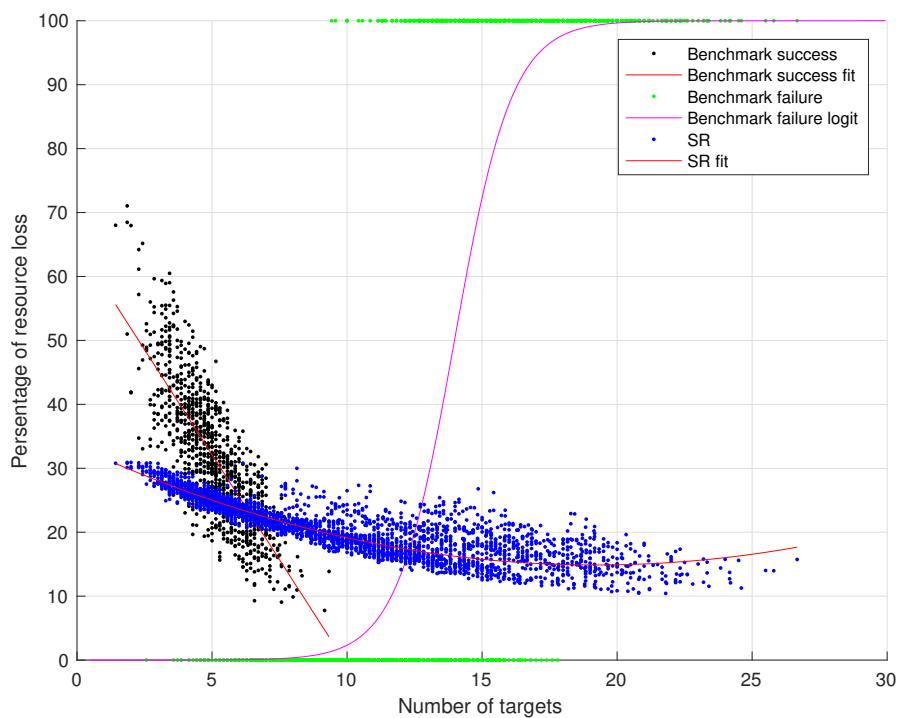


Figure 3.14: Evaluation of Synchronous Rendezvous (SR) and base case performance. The SR method and the successful benchmark results are fitted using linear regression. The mixed results from the base case are evaluated using logistic regression. The operational regions of both approaches can be evaluated following the approximated curves.

function. MATLAB general linear model library was used to calculate the logistic regression parameters. The resulting sigmoid function has a domain between zero and one. On the graph, this is translated to 0 and 100 % loss. By choosing a limit for probability of mission failure on the sigmoid function, the upper limit of the benchmark operational interval is defined. If mission failure rate of 30% (y axis) is selected, interpolating this towards the x axis, the crossing point will be at 14 targets. This makes the benchmark a better choice than the SR method when the average number of simulated targets is between 7 and 13. The SR method is the superior approach for all other target distributions. The threshold limit on the sigmoid function can be changed depending on how many mission failure results are acceptable for the analysis.

Figure 3.14 summarises the results from this chapter. It provides a straightforward comparison between the Synchronous Rendezvous (SR) and benchmark methods. It can be seen that for low and high number of targets, the SR method outperforms the benchmark, while for a medium number of targets, the benchmark is better. This result can be traced to finding out the favourable conditions for which the benchmark vehicle task formation would be best suited - the mission scenario operational parameters and average workload of 7 to 13 targets per SR interval. If this information is available and an estimation of the targets can be made, a predefined mission would be more efficient. However, in the case of uncertainty of defining the target model, the SR method is likely to adapt better and be more efficient overall.

The presented analysis is tightly related to the choice of parameters defining the simulation scenario. However, the analysis steps can be repeated to factor in various parameter choices and adjust the SR method loss based on the mission and operation needs. This can give an indication when the SR method might not be suitable.

## 3.4 Summary

This chapter defines a scenario following a Mine Countermeasures (MCM) mission constraints and requirements. The adaptive Synchronous Rendezvous (SR) method for multi-AUV MCM mission has been presented. The advantage of the SR is that by dynamically retasking the vehicles, they are utilised at a constant rate throughout the mission, independent of the number and location of detected targets. In contrast, following a typical mission planning approach of allocating the function of the vehicles at the start of the mission, the mission efficiency and success are dependent on the target workload. The SR method gains and losses have been demonstrated in a numerical simulation evaluating the time

resource loss for the robotic system due to the vehicles travelling to meeting points, as well as the flexibility of the system to adapt to emerging workload. Based on the results analysis, the SR method outperforms the benchmark case when there are a low or high number of targets in the minefield. The predefined allocation mechanism is more efficient when the workload can be predicted and divided well between the available robotic assets. The simulation results and analysis is highly dependent on the initial conditions of the system — defining scenario and operational parameters. However, this chapter suggests how and to what extent the SR method can be used for a typical MCM setting.

## Chapter 4

# Synchronous Rendezvous Planning with Markov Decision Process (SR-MDP)

The Synchronous Rendezvous (SR) method enables managing of the vehicles' resource during the Mine Countermeasures (MCM) mission. The previous chapter showed an adaptive setting of the algorithm, where the system revised the tasks of the Autonomous Underwater Vehicles (AUVs) based on the emerging workload. It also showed the advantage of the method that by utilising the vehicles at constant rate, the effect of the loss incurred by meeting, is reduced. The disadvantage of the adaptive SR is that it uses heuristics for myopic decision making — the vehicles adapt to the current workload without making an estimate, or build an expectation, for the future. If looking into the future is possible, then taking this into account when planning the next vehicle action can lead to optimal resource distribution in the system. In this chapter, the SR method is explored when moving from adaptive to intelligent planning using a Markov Decision Process (MDP). Parts of this chapter's method and results are published in [96].

### 4.1 Markov Decision Process Definition

The Markov process is a random process where the current values (states) determine the future probabilities. The information of the past probabilities, or the sequence of past events, is discarded. Formal

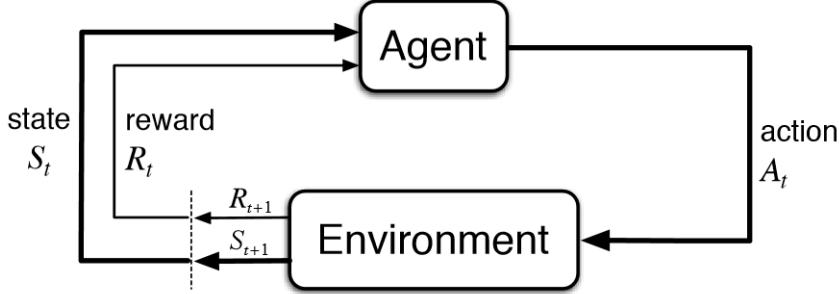


Figure 4.1: Reinforcement learning: agent-environment interaction [1].

definition of the Markov property can be found in many texts, such as [1]. The notation in this chapter follows the updated draft of [1] from 2016.

One type of a Markov process is the Markov chain. It can be described as a Markov process with discrete and countable state space, where the state space is the set of all values that a process can take. A discrete-time Markov chain is when the process evolves through discrete time steps, as opposed to a continuous-time Markov chain, where the process changes over a continuous time interval. The Markov Decision Process is a Markov chain where an agent can make decisions by taking actions. The Synchronous Rendezvous (SR)-Markov Decision Process (MDP) model is a discrete-time discrete-space MDP. The MDP can formulate a problem by encompassing all relevant information by defining the agent's state, action and goal.

#### 4.1.1 Markov Decision Process Elements

The elements defining an MDP are state, action, transition between states and reward. The interaction between agent and environment can be seen in Figure 4.1, [1], showing a common representation of the Reinforcement learning (RL) problem, which also applies to the planning problem.

The agent and the environment interact at every time step, or episode,  $t \in [0, T]$ , with  $T$  being the last time interval of the available period.  $t$  can be discrete and countable, as in the case of the SR-MDP problem, but it can also be modelled with infinite or continuous interval. At each time step  $t$ , the environment sends a signal to the agent representing its state —  $S_t \in \mathcal{S}$ , where  $\mathcal{S}$  is the set of all possible states. The agent interacts with the environment by selecting an action, available from a set of actions at state  $S_t$ ,  $A_t \in \mathcal{A}(S_t)$ . The probability that the agent will transition from a state  $s$  to another state  $s'$ , given action  $a$  is given by:

$$p(s'|s, a) = Pr\{S_{t+1} = s' | S_t = s, A_t = a\} \quad (4.1)$$

During the next time step,  $t + 1$ , as a result of the agent's action, the environment sends a reward signal,  $R_{t+1} \in \mathcal{R}$ , and an indication of the agent's new state  $S_{t+1}$ . The equation defining the expected reward for the next time step,  $R_{t+1}$ , given the current state  $s$ , the action  $a$  and the expected resulting state  $s'$  is:

$$r(s', a, s) = E\{R_{t+1} | S_t = s, A_t = a, S_{t+1} = s'\} \quad (4.2)$$

The reward represents the mission's goal encoded in the agent's preference - it returns a positive reward for behaviour that will advance the mission and we want to encourage, and penalises deviations or undesired mission results. The goal of the agent is to maximise the sum of the rewards over the future. When the process is stochastic, there is uncertainty in the future rewards. To model this uncertainty, the future rewards are discounted and the aim is to maximise the expected return function,  $G_t$ , defining a specific reward sequence:

$$G_t = \sum_{k=0}^T \gamma^k R_{t+k+1} \quad (4.3)$$

$\gamma$  is the discount rate and can take values between  $[0, 1]$ . A reward received  $k$  time steps ahead in the future is  $\gamma^{k-1}$  times less valuable compared to a reward received immediately. If  $\gamma=0$ , the agent ignores the future reward and only maximised each immediate reward. This is a myopic agent. If  $\gamma$  approaches 1, the agent puts nearly the same weight to all future rewards, no matter how far they are in the future. This is the case when uncertainty in the system is low and the agent benefits from being farsighted.

The output of an MDP is a policy,  $\pi$  — a sequence of actions the agent should take from a given state. This chapter concludes with finding an optimal policy for the MCM problem within the scenario setting from Chapter 3.

## 4.2 Markov Decision Process Formulation for MCM with SR

To formulate an Markov Decision Process (MDP) planning problem, the domain knowledge from the Mine Countermeasures (MCM) application needs to be exploited. More specifically, the abstractions of agent and environment need to be defined, as well as the boundary between them.

The scenario described in the previous chapter is reiterated here in order to redefine the assumptions to fit the use of the MDP model. As before, focusing on the mission planning problem, some parameters and capabilities of the MCM application are treated as a blackbox. Following the structure of an MCM mission, here are the assumptions made for the SR-MDP approach:

- Search: 1) Side-Scan Sonar (SSS) data is collected in a lawnmower pattern, aiming for complete area coverage; 2) the sensor collects data within its swath width,  $w = 50$  m; the speed of the vehicles is  $v = 2$  m/s. The problems of optimal coverage [97], localisation and navigation [29] and image processing [49] are still open research questions, however they are not considered further, and are assumed solved up to a sufficient level for the purpose of the mission planning problem setting.
- Detection: The probability of detection is a function of the imaging sensor, which comes as a continuous input data stream throughout the mission. It depends on the quality of the collected data, the position of the vehicle and the algorithms used to distinguish objects from the background. This work considers the sensor as a blackbox and no real or simulated data is provided as input. Instead, a deterministic detection output is assumed, and is passed to the MDP planner as target locations that need to be revisited.
- Classification: In order to increase the probability of classification, additional data collection is required. This is done by the AUV reallocating the detected targets during the search. Increasing the resolution of the imagery by collecting data from closer proximity, and the ability to observe a target from different angles, gives significant performance increase to the classification algorithms, relying on distinguishing object features [92]. The processing of the sensory data, the accuracy of target positioning and data collection, are out of the scope of this work. What is taken to the SR-MDP planner from the MCM classification phase, is creating the second task, apart from search, — the Reacquire-Identify (RI) task, which was already introduced in the previous chapter.
- Identification and Neutralisation: These two phases are out of the scope for an autonomous MCM mission as they would require human input to decide how to dispose of an identified target.

#### 4.2.1 SR-MDP Goal

The overall goal of the AUVs during the MCM mission is to maximise the search area while revisiting all detections. Two tasks are defined based on the mission goal: search and RI (reacquire and collect

additional data for classification and identification). The SR method, evaluated in the previous chapter, introduces multiple meeting points throughout the mission for data exchange and task allocation. The SR method gives an additional goal - vehicles need to meet at, or around, a predefined time interval in order to bring predictability to the mission. An operator could monitor the mission at these meeting points, and possibly collect data at known intervals. As a result, the vehicles need to weight the benefit between not deviating from scheduling meeting points at predefined intervals and optimally using the network's resources.

## 4.3 Modelling Multiple Vehicle MCM Planning with MDP

The elements of the MDP were listed in subsection 4.1.1. This section describes in detail each of them as modelled for the Synchronous Rendezvous (SR)-Markov Decision Process (MDP) application.

### 4.3.1 State Space Model and Discretisation

The battery level of the vehicles and workload, based on detected targets, are used to represent the MDP state space for the SR-MDP problem. This makes the state space a 2D matrix.

The battery level, or the endurance of the vehicle, can be expressed in time units. By knowing how much operational time the vehicle has left, it can decide whether and by how much a task can be postponed.

The workload is based on the shortest path to visit known detections and an example is given in the right-hand side of Figure 3.1 in Chapter 3. The workload depends on the number and location of the detections. The initial point of the path is the current SR and the final point is the next SR. This assignment is called RI and corresponds to the same task from Chapter 3. Once a target is visited, it is removed from the list of detections.

Both the battery ( $S_1$ ) and workload, or RI, ( $S_2$ ) state spaces are defined in time and are continuous. In order to make an online planner, the MDP needs to be solved for the full 2D state space of positive real integers,  $S = \mathbb{R}_+^2$ , in a computationally efficient way. It is an open research question how to deal with continuous state spaces and this decision depends on the application and how much approximation it allows. Value function approximation methods are a current research topic, such as neural networks and regression trees representation of the value function output [36]. Another solution is using factored

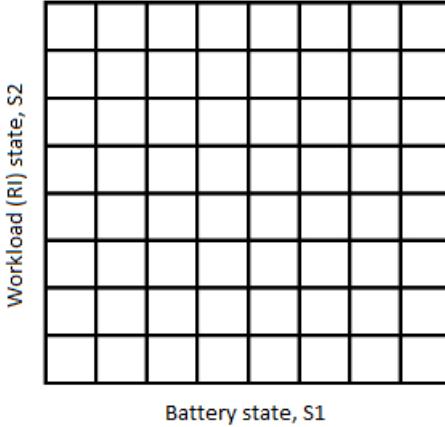


Figure 4.2: Discretised state space

representation of the transition probably function by a Bayesian network, where states are grouped together and assigned the same value function [98], [99].

Another consideration for the MDP framework, regarding the state space representation, is the assumption of observability — the agent knows its current and surrounding states, or has full observability. Real world problems involve the agent receiving information through noisy sensors so there is uncertainty about the current state, and often do not see the full extent of the state space. This is the case of partial observability. The POMDP model considers this, however this is not taken into account in the development of the SR-MDP method.

The focus is on simplification of the continuous state space in order to reduce the computation of the planner and make it applicable for real time operations. The straightforward method to solve the continuous state space problem is to discretise it, such as shown in Figure 4.2.

There are two disadvantages to applying discretisation. First, an assumption is made that the evaluation within each cell is constant. The larger the cell discretisation intervals are made, the stronger this assumption becomes. However, if the discretisation interval is reduced, this increases the computation load. The second disadvantage is the ‘curse of dimensionality’, a common problem for MDP frameworks [100]. This refers to an exponential growth of computational complexity with regards to the number of state spaces and does not scale for large problems. For example, if we have  $n$  number of state space dimensions and we discretise them in  $k$  intervals, the resulting state space will be  $k^n$ . For a 2D problem with 100 discrete values, this results in  $100^2 = 10000$ . However, for a 10D problem, this will be  $10^{10}$ , which is not computationally feasible at the moment. The discretisation state space method is recommended for small 1D or 2D problems [101], while additional proofs and other approaches are suggested for higher dimentionality [102], [103].

In order to take advantage of the state discretisation method, the MCM problem is represented in the time domain, as described earlier. The location information of the vehicles is contained in the battery state variable,  $\mathbf{S}_1$ . The assumption is that their location is defined by the time the vehicles spend doing a search task in a lawnmower pattern, with predefined area parameters and constant speed (all environmental disruptions are ignored). The target locations are contained in the workload state variable,  $\mathbf{S}_2$ , by calculating the time to reach the detections based on shortest path calculation. These state representations enable the use of time-to-track vehicles and targets locations, instead of x and y coordinates, and each state space is transferred from 2D to 1D.

The selected interval discretisation of one minute for both  $\mathbf{S}_1$  and  $\mathbf{S}_2$ , makes the state space discrete and finite ordered sequence. Moving from continuous to discrete space approximation, the notation for state changes to  $\bar{s}$  for a specific state and  $\bar{S}$  for the full set.

$$\bar{S}_1 \in [s_0, s_1, \dots, s_i]^T, i \in [0, 600] \quad (4.4)$$

$$\bar{S}_2 \in [s_0, s_1, \dots, s_j]^T, j \in [0, 300] \quad (4.5)$$

$$\bar{S} = \bar{s}_{i,j} \in \mathbb{N}^{l \times m}, l = 600, m = 300 \quad (4.6)$$

where  $\bar{s}_{i,j}$  is the corresponding element from  $\bar{S}_1$  and  $\bar{S}_2$  respectively.  $\bar{S}$  is a set of non-negative integers, represented as a matrix<sup>1</sup>.

The choice of interval discretisation, one minute, is based on the assumption that change for the underwater system will not be significant within that time, due to the slow speed of AUVs.

The choice of limiting the battery state,  $\mathbf{S}_1$ , to 600 minutes means giving the vehicles 10 hours endurance, until they need to finish the mission (specifications based on the SeaCat vehicle, further details given in Chapter 5).

The workload state,  $\mathbf{S}_2$ , limit was selected to keep the state space low. This also gives a practical limit to the number of targets the system can handle. If all three vehicles with combined operational time of  $3 \times 600$  minutes need to do RI task for more than half of this time ( $3 \times 300$ ), then it will be better

---

<sup>1</sup>For the remaining of the thesis, the state space is discretised, however the bar is removed to ease readability.

to do another coarse search of the area, as there might be too many false alarms and reallocating all of them will have limited benefit for the mission.

The states in an MDP can be terminal and non-terminal. A terminal state triggers termination of the mission. In the simulation, terminal states are all states that have reached the battery state limit:  $s(-1,:)$ .

### 4.3.2 Action Space Model and Discretisation

The action space represents how the Autonomous Underwater Vehicle (AUV) agents interact with the environment in the Mine Countermeasures (MCM) scenario. In this work, all simulations assume three available vehicles. Due to the low number of agents, the action model can be simplified by having a look-up table for all task combinations divided between the agents. That way the problem considers a model for a single agent instead of multi-agent system. The set size of the action space for the SR-MDP model grows linearly with the number of vehicles  $n$ :

$$|\mathcal{A}| = 2n + 1 \quad (4.7)$$

The available actions are listed in Table 4.1 and an example of the AUV paths are shown in Figure 4.3. There are seven actions defined for the Synchronous Rendezvous (SR)-Markov Decision Process (MDP) model. Each action sequence number (ID) is seen in the first column of Table 4.1. The actions are based on:

1. How many vehicles are tasked with search and how many are tasked with Reacquire-Identify (RI). This is shown in the second column of Table 4.1, where  $n$  gives the number of search vehicles, out of three vehicles in the network.
2. What is the duration of the SR time interval. It can be the predefined duration,  $t_{SR(min)} = 60$  minutes, following Chapter 3, like actions 4,5,6 and 7. The other option is adapting the SR duration to the current workload (time to visit the shortest path between detections - encoded in the  $\mathbf{S}_2$  state space), and divide  $t_{RI}$  by the number of vehicles tasked with RI. This is done for actions 1,2 and 3. The SR time duration,  $t_{SR}$ , is shown in the third columns in Table 4.1.
3. The last column in the table is the overall RI time spent by the vehicles during the SR cycle. It is based on the SR duration and the number of RI vehicles. The SR interval choice is a tradeoff

Table 4.1: Action space

ID	$n$	$t_{SR}$	$t_{RI}$ (completed)
1	0	$t_{RI}/3$	$t_{RI}$
2	1	$t_{RI}/2$	$t_{RI}$
3	2	$t_{RI}$	$t_{RI}$
4	3	$t_{SR(min)}$	0
5	2	$t_{SR(min)}$	$t_{SR(min)}$
6	1	$t_{SR(min)}$	$2t_{SR(min)}$
7	0	$t_{SR(min)}$	$3t_{SR(min)}$

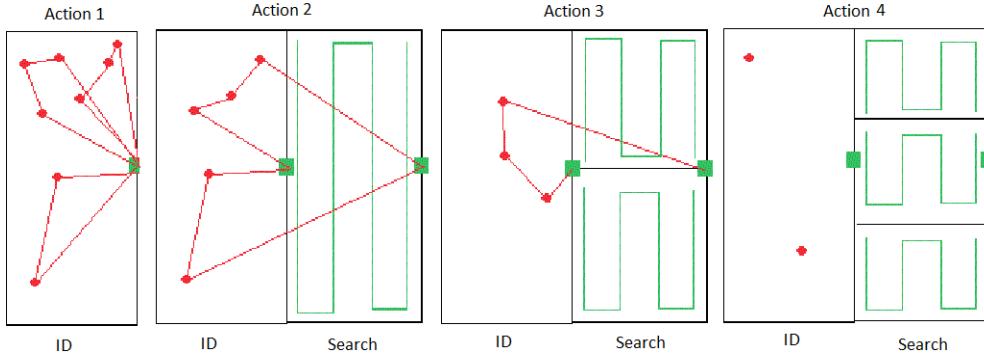


Figure 4.3: AUV path examples illustrating Actions 1,2,3 and 4 from Table 1. Actions 5,6 and 7 are similar to Actions 3,2 and 1, respectively, but the SR duration is based on the predefined  $t_{SR}$  time, not the adaptive RI time interval.

between distributing the resource more efficiently (length based on RI time, or actions 1, 2 and 3), or following a strict procedure of regular meeting intervals ( $t_{SR(min)}$ , or actions 4, 5, 6 or 7).

Limitations are set for some actions in order to comply with the index limits of the state space. Firstly, the target model is a function of search area (explained in the next section), and causes the index for Action 3 in the numerical simulation to go out of bounds. This is due to setting the SR duration to the maximum workload time, 299 minutes, and sending two vehicles to search for that duration, resulting in scheduling too much workload for the system. Hence, a limit of 200 minutes is set only for Actions 1, 2 and 3 (although only Action 3 was usually affected by it), and the excess of RI is handled as credit for the next time period. This limitation is due to the workload,  $\mathbf{S}_2$ , dimension of the state space.

The battery state space,  $\mathbf{S}_1$  brings another exception to the action space. This is due to the condition that actions cannot lead the system beyond a termination state. To keep the mission duration limit of 600 minutes, or always reach  $s[-1,:]$ , all actions adopt the behaviour of Action 1, once the sum of the current value of  $\mathbf{S}_1$  with the expected SR duration, exceeds the termination state index. For example, if we are in battery state  $s(580,:)$ , then the maximum SR duration can be  $599 - 580 = 19$  minutes, and all vehicles are set to do an RI task.

The action limitations are consistent and no loss is introduced to the model. If part of the workload is left unfinished during the current time step, it is transferred to the next, which results in reduced reward. If the agents select actions that overload the system and not enough resource is present to handle it by the end of the mission, this results in a large penalty to show that this is undesirable behaviour. The reward function sections gives further details.

### 4.3.3 World Model Approximation

The world model is the agent's perception of the environment's dynamics. Equation 4.1 shows how any state and action define the probability of reaching a possible future state  $s'$ .

The main goal of a reinforcement learning agent is to learn the transition dynamics of the environment. Once this is known, the agent can calculate its optimal action sequence, or solve the planning problem. The transition function is assumed known for the MCM scenario and this work is concerned with solving the planning problem in near real time. The transition function for the SR-MDP method links the search area, the number of detections and the shortest path to revisit them.

### Target Model

The target model follows partially the parameter selection from Chapter 3. The location of the targets are uniformly distributed over the x and y coordinates of a unit area,  $A_U$  of a  $0.5 \text{ km}^2$ : for example a square  $U_x\{0, 707\}$ ,  $U_y = \{0, 707\}$ . This is different in comparison to Chapter 3, where the location distribution was also uniform over x and y, but the size of the area was defined by the SR duration, outlining the search area, which varied. For the SR-MDP, the target model scales with a unit area,  $A_U$ .

The three target models tested with the SR-MDP approach have a mean number of targets per unit area of 5, 10 and 15, the same as the target average from Chapter 3. The difference here is that instead of a uniform distribution, a Gaussian filter is applied to model the number of targets expectation. The Gaussian filter reduces the strong uncertainty assumption of the uniform target model from Chapter 3. This aims at making a more gradual change in target number when the vehicles are doing a search in adjacent areas, moving from one SR interval to another. The three model distributions that are tested are  $G_1(\mu = 5, \sigma^2 = 1)$ ,  $G_2(\mu = 10, \sigma^2 = 1)$ ,  $G_3(\mu = 15, \sigma^2 = 1)$ .

Figure 4.4 gives the Gaussian filter distributions for the three target models,  $G_1$ ,  $G_2$  and  $G_3$ , for a

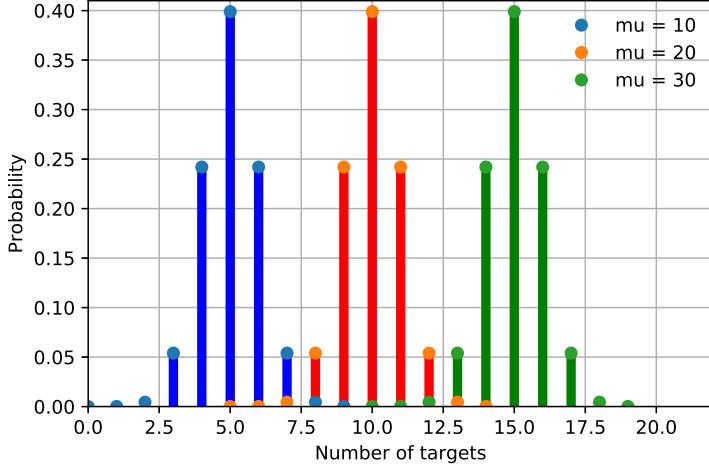


Figure 4.4: Target models based on sampled Gaussian kernel per unit area

unit area of  $A_U = 0.5 \text{ km}^2$ . It shows the stochastic nature of the transition dynamics in the SR-MDP model.

In order to reduce the computation, the continuous Gaussian distribution is approximated using a Gaussian filter. This is the process of convolving the discrete target space with a sampled Gaussian kernel:

$$L(x, u) = \sum_{q=1}^{10} f(x - q) G(q, u) \quad (4.8)$$

where  $L$  is the Gaussian filter,  $f$  is the discrete target space.  $L$  is truncated at the ends to confine the Gaussian function between a finite interval, defined by  $q = [1, 10]$ , so there are 10 probabilities to propagate further in the MDP model.  $G(q, t)$  is the sampled Gaussian kernel:

$$G(q, u) = \frac{1}{\sqrt{2\pi u}} e^{\frac{(q-\mu)^2}{2u}} \quad (4.9)$$

where  $u = \sigma^2$  is the variance and  $\mu$  the mean. Once the unit area, and target number and location distributions are defined, the search time for each SR time interval scales the detections found based on the search area, given in Equation 3.4. Then, the number of targets in the searched area,  $A_S$ , is proportional to the unit area ( $A_U, [\text{km}^2]$ ):

$$N_{MLO} = \frac{\mu \times A_S}{A_U} = \frac{\mu \times v \times t_{SR} \times w}{A_U} \quad (4.10)$$

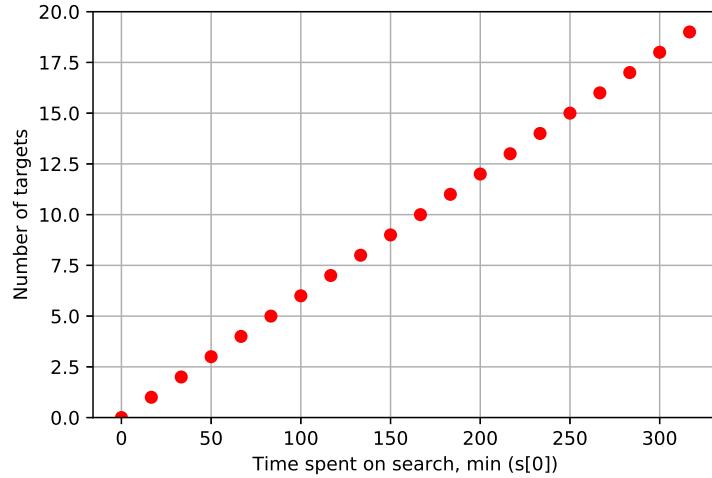


Figure 4.5: Expected average number of targets as a function of time spent on search task.

Figure 4.5 shows the linear relationship described in Equation 4.10 by giving the expected average number of targets for  $G_2(10, 1)$  as a function of time spent on search. Figure 4.4 gives a sample of the target models used to calculate the transition probability. The figure shows the probability of expected number of targets. A total of seven such target samples are calculated for each state  $s$ , depending on the action,  $a$ .

This target model and scaling are unrealistic, but modelling of the mine field is outside of the scope of this thesis, and details of such distributions are rarely disclosed in open literature. The SR-MDP algorithm will be valid, if an updated environment model is applied.

### Transition Probability Model - Shortest Path between Targets

From the described target model, a prediction of the number of detections for the next SR interval is obtained. Then, this needs to be translated into an expectation of a future state space,  $s'$ . To do this, the location of the targets has to be factored in. A uniform distribution is used to model the x and y coordinates of the targets within the search area. The problems becomes an instance of the Travelling Salesmen Problem (TSP) — finding the shortest path to visit the targets, and then calculating the time required for a vehicle to do that. Since there are numerous variations for x and y locations, an approximation of the shortest path is used, based on a model, suggested in [104]:

$$d = K \sqrt{N_{MLO} \times A_S} \quad (4.11)$$

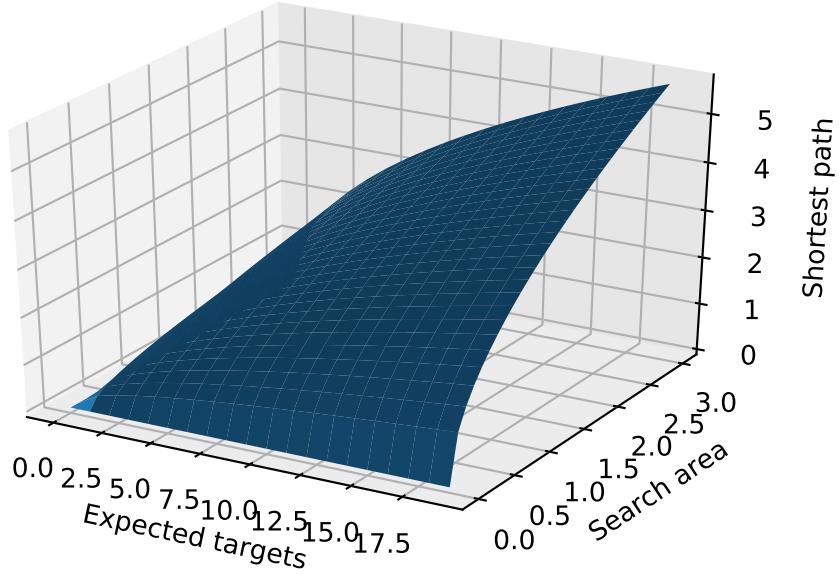


Figure 4.6: Average expected shortest path in  $km$  (z axis) as a function of number of targets (x axis), uniformly distributed within a search area in  $km^2$  (y axis). Calculations follow Equation 4.11.

where  $d$  is the average distance travelled between the targets,  $K$  is an empirical coefficient ( $K = 0.765$ ),  $N_{MLO}$  is the number of targets, given from the predictive model presented earlier, and  $A_S$  is the size of the plane where the targets are confined to (the searched area). In case a target is left from a previous SR interval, this area size,  $A_S$  will grow for the next SR interval to include the target locations that were not reacquired before.

The final step to getting  $s'$  is to transfer the average path length,  $d$ , from distance to time by dividing it by the speed of the vehicles. Constant speed of  $2\text{ m/s}$  is assumed:

$$t_{RI} = \frac{d}{v} = \frac{d}{2} \quad (4.12)$$

Figure 4.6 brings equations 4.11 and 4.12 by showing the shortest path ( $km$ ) on the z axis (Equation 4.12) as a function of expected target number and search area ( $km^2$ ) (Equation 4.11).

One limitation of the shortest path approximation method from [104], given in Equation 4.11, is that it gives an average distance. This means that each of the ten probabilities propagated by the transition function to reach  $s'$  from any state  $s$  and action  $a$ , needs to be multiplied by a normal distribution in

Table 4.2: Reward function

Name	Unit	Function
Area ( $R_a$ )	1 min time on search	$y = x$
SR deviation ( $D_{SR}$ )	1 min deviation from SR	$y = 40 \times e^{(1 - \frac{t_{SR}}{t_{SR(min)}})} - 40$
Workload Efficiency ( $E_{RI}$ )	1 min RI mismanaged	if $x > 0, y = 0.01x^2$ , if $x < 0, y = 0.1x^2$
Mission failure	1+ unidentified targets	$y = -300$

order to account for the unlimited options the approximation can deviate from in reality. This aspect is ignored for the SR-MDP approach. The disadvantage is that the precision of the shortest path time expectation is reduced. However, since the transition dynamics is not relying on a real target model, this precision is already lost. In case a real target distribution is applied, the shortest path expectation should be given accurate boundaries and this will add value to the solution.

#### 4.3.4 Reward Function Model

The reward represents the agent's preference and together with the transition probability function, it models the environment's dynamics. The Mine Countermeasures (MCM) scenario goals are introduced to the agents by rewarding them for desirable behaviour and penalising them for deviations or unwanted mission results. Equations 4.2 and 4.3, linking the expected return to the current state, action and possible future state, as well as its propagation over the MDP time steps, was explained earlier.

For the SR-MDP model, the goal is to increase the search area, while revisiting all contacts. Having a target that is not visited after the end of the mission, results in a mission failure and brings a very high negative reward (penalty or cost). This makes the agent avoid ending the mission in such a state. Another objective is not to deviate much from the predefined SR interval,  $t_{SR(min)}$ . This is a soft condition, and the deviation from  $t_{SR(min)}$  is weighted against the benefit of efficiently distributing the vehicle's resources. The reward model and the relationship between its elements is given with equation 4.13 and Table 4.2.

$$R = R_a - |D_{SR}| + E_{RI} \quad (4.13)$$

where the  $R_a$  is the positive reward associated with gaining search area during the SR interval. It gives  $R = 1$  for each minute the agent spends on search (area is translated to time assuming constant speed and search pattern of the vehicles). The  $D_{SR}$  is negative reward for deviating from the set SR interval:

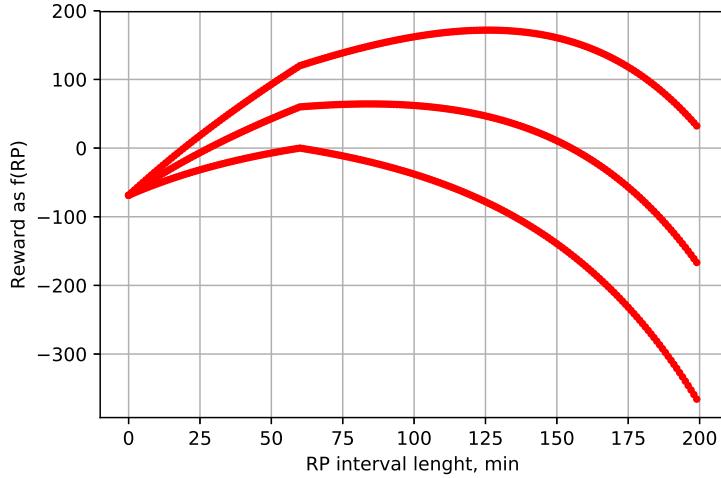


Figure 4.7: Reward as a function of SR duration — area gain  $R_a$  and penalty for exceeding the expected SR,  $D_{SR}$ .

$$D_{SR} = 40 * e^{(1 - \frac{t_{SR}}{t_{SR(min)}})} - 40 \quad (4.14)$$

where  $t_{SR}$  is the choice of SR interval for the next time step and  $t_{SR(min)}$  is the preferred value. Deviation from  $t_{SR(min)}$  gives penalty. The equation is heuristically weighted by multiplying it and subtracting by 40 to make the penalty with a desirable ratio in comparison to the other reward elements. The relationship between the current and desired SR interval is exponential so that further difference is penalised more.

Finally, the  $E_{RI}$  measures how efficiently the RI time is managed. If targets are left for a later stage and less RI resource is spent than is required, then there will be workload credit carried to the next SR. If the RI resource is over-committed, then the vehicle will be idle, and this brings a tenfold increase in the penalty, compared to carrying a credit. This happens when the SR window is larger than the time required to RI all detections. The  $E_{RI}$  reward function is given in Table 4.2.

Figures 4.7 and 4.8 show a sensitivity analysis of the reward function. Both  $R_a$  and  $D_{SR}$  depend on the SR interval length and their relationship is shown in Figure 4.7. The  $E_{RI}$  penalty depends on the difference between required and committed RI resource. In Figure 4.8 the penalty is ten times higher when a vehicle is idle (over-committed RI resource,  $x < 0$ ) compared to when there is a credit workload left for later (under-committed RI resource,  $x > 0$ ).

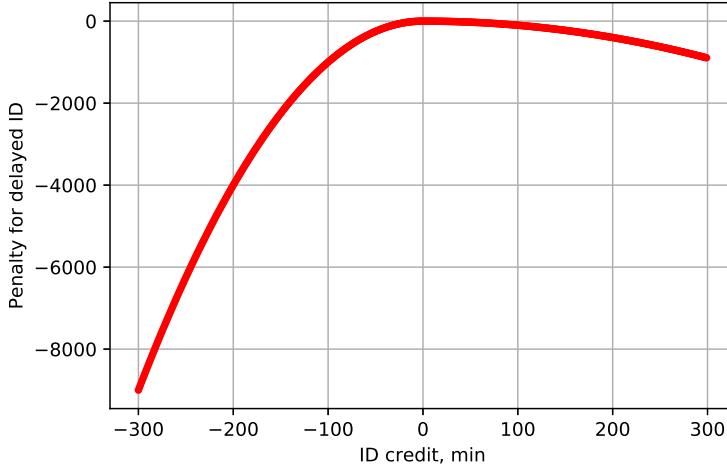


Figure 4.8: Reward as a function of RI state and action — penalty for carrying over workload between SRs ( $x > 0$ ) or being idle ( $x < 0$ ).

#### 4.3.5 Horizon Space

The horizon defines how many time steps will be executed until the termination of the algorithm. Often, MDP models are defined with an infinite horizon space and a discount factor,  $\gamma$ , is introduced to compute an expected return. For the SR-MDP problem, there is a finite number of time steps, defined by the number of scheduled SRs throughout the mission. They are limited by the battery resource. Therefore, the problem is with unbounded horizon space — finite, but unknown number.

The SRs allows discretisation of the horizon space in a way that the agents need to make very few decisions over the whole mission, reducing the problem significantly. For example, if the minimum SR time interval was always selected,  $t_{SR(min)}$ , then the maximum time steps, or number of decisions the agents need to make, is ten (600 minuets of operation and 60 minutes minimum SR time interval). This is the reason the computation can be executed in near-real time.

### 4.4 MDP Solution Method

Different solution methods exist for solving an MDP. They can be classified into Dynamic Programming (DP), Temporal Difference (TD) and Monte Carlo (MC) methods [1].

DP methods have a well developed theoretical background but require an existing model of the environment (transition probability and reward). The output is an optimal policy. They have limited application due to the assumption of world model and high computation load. The DP algorithms are

based on different update rules of the Bellman equation. The DP algorithms are Policy Evaluation (computing the value function iteratively for an arbitrary policy), Policy Improvement (finding a better policy than the original by a greedy search through the value function), Policy Iteration (repeating Policy Evaluation and Policy Improvement until finding an optimal policy), Value Iteration (Policy Improvement and truncated Policy Evaluation (after one sweep)).

Monte Carlo (MC) computes full roll-outs so it is not suitable for estimating the reward step-by-step, but rather episode-by-episode (an episode is executing the whole mission or algorithm until a terminal state is reached). One major advantage of the approach is that it does not require a model and learns an optimal behaviour by direct interaction with the environment. The Monte Carlo method averages its experience based on the sample sequences it collects, or on complete returns. This makes it a useful approach if a simulation environment is available. The ideas of policy evaluation and improvement are extended to the Monte Carlo case.

Temporal Difference (TD) methods combine features from both Dynamic Programming and Monte Carlo. Like MC, no model is required and the environment dynamics can be learnt from experience. Similar to DP, TD updates its estimates based on other estimates (bootstrapping).

#### 4.4.1 Value Iteration

The solution method used for the Synchronous Rendezvous (SR)-Markov Decision Process (MDP) model is Value Iteration, shown in Algorithm 2. It is a Dynamic Programming (DP) algorithm that recursively computes the value function,  $V(s)$ , working backwards from the goal states to the initial states.  $V(s)$  gives an evaluation of the goodness of a state, or how desirable each state is. Value Iteration uses the Bellman Optimality Equation [100] as an update rule to find the optimal value function  $V^*(s)$ :

$$V^*(s) = \max_a \sum_{s'} p(s'|s, a)[r(s, a, s') + \gamma V^*(s')] \quad (4.15)$$

Algorithm 2 starts by initialising the value function for all states to 0. Then it iterates over each state  $s$  and action  $a$  to estimate the expected reward,  $p(s'|s, a)[r(s, a, s') + \gamma V(s')]$ . The maximum sum over the actions is selected and the value assigned to the new  $V(s)$ . This step is repeated until the difference between the value function in the current and the previous cycle,  $V(s)$  and  $v$  respectively, is negligible. This threshold depends on the application. For the SR-MDP model, four iterations are enough for the

algorithm to converge. The policy,  $\pi$ , is the output of the algorithm and its equation is shown in Step 4. It gives the optimal action sequence the agent can select from each state  $s$ . The equation to extract the policy is the same as the value function, but the assignment uses the argument, rather than the value. This classical dynamic programming (DP) algorithm is optimal and the proof is available in [100].

---

#### **Algorithm 2** Value Iteration

- 
1. Initialise  $V(s) = 0$ , for all  $s \in S^+$
  2. Repeat until converged ( $v - V(s) \leftarrow 0$ ):
    - (a) For each  $s \in S$ :
      - i.  $v \leftarrow V(s)$
      - ii.  $V(s) \leftarrow \max_a \sum_{s'} p(s'|s, a)[r(s, a, s') + \gamma V(s')]$
    3. Output: policy  $\pi$
    4.  $\pi(s) = \arg \max_a \sum_{s'} p(s'|s, a)[r(s, a, s') + \gamma V(s')]$
- 

#### 4.4.2 Results

Figures 4.9 and 4.10 show the output of Algorithm 2 for the SR-MDP — the value function and the optimal policy. The  $x$  and  $y$  axis for both figures show the state space, the Reacquire-Identify (RI) workload time,  $S_1$ , and the remaining battery time  $S_2$ . The  $z$  axis on Figure 4.9 is the overall amount of reward the agent is expected to accumulate until the end of the mission, starting from the current state. This is the value of the state. The  $z$  axis on Figure 4.10 is the optimal action for the agent for the given state. Figure 4.10 is the solution of the SR-MDP problem.

#### 4.4.3 Discussion

The 3D graph from Figure 4.10, showing the optimal action policy, is sliced into 2D plots in Figure 4.11, giving each individual action selection. The white space on each plot is that action being selected for the corresponding state. Each action can be selected only once out of the whole seven plots so if the white spaces from all seven plots are put together, they will form a single white plot without an overlap. The label on top of each plot tells the action it represents. It is useful to consult Table 4.1 to follow the discussion on this graph.

In order to analyse the benefits of an optimal decision making as opposed to a myopic solution, the converged solution from Figure 4.11 is compared to the solution given after the first iteration of the algorithm in Figure 4.12.

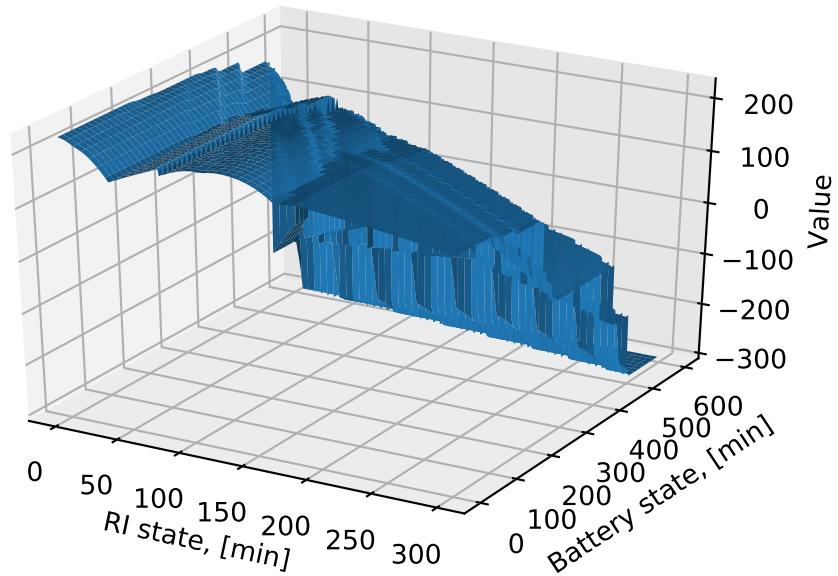


Figure 4.9: Value function solution.

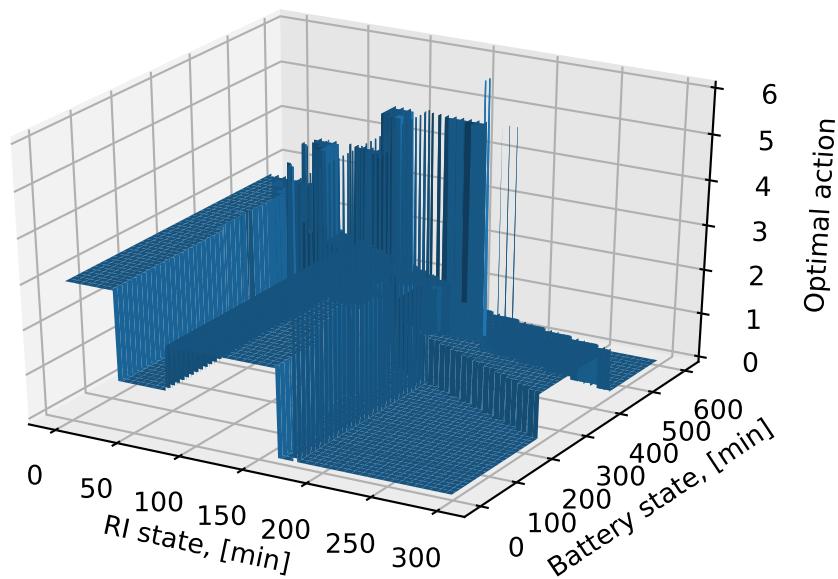


Figure 4.10: Optimal policy solution of the multi-AUV MCM problem.

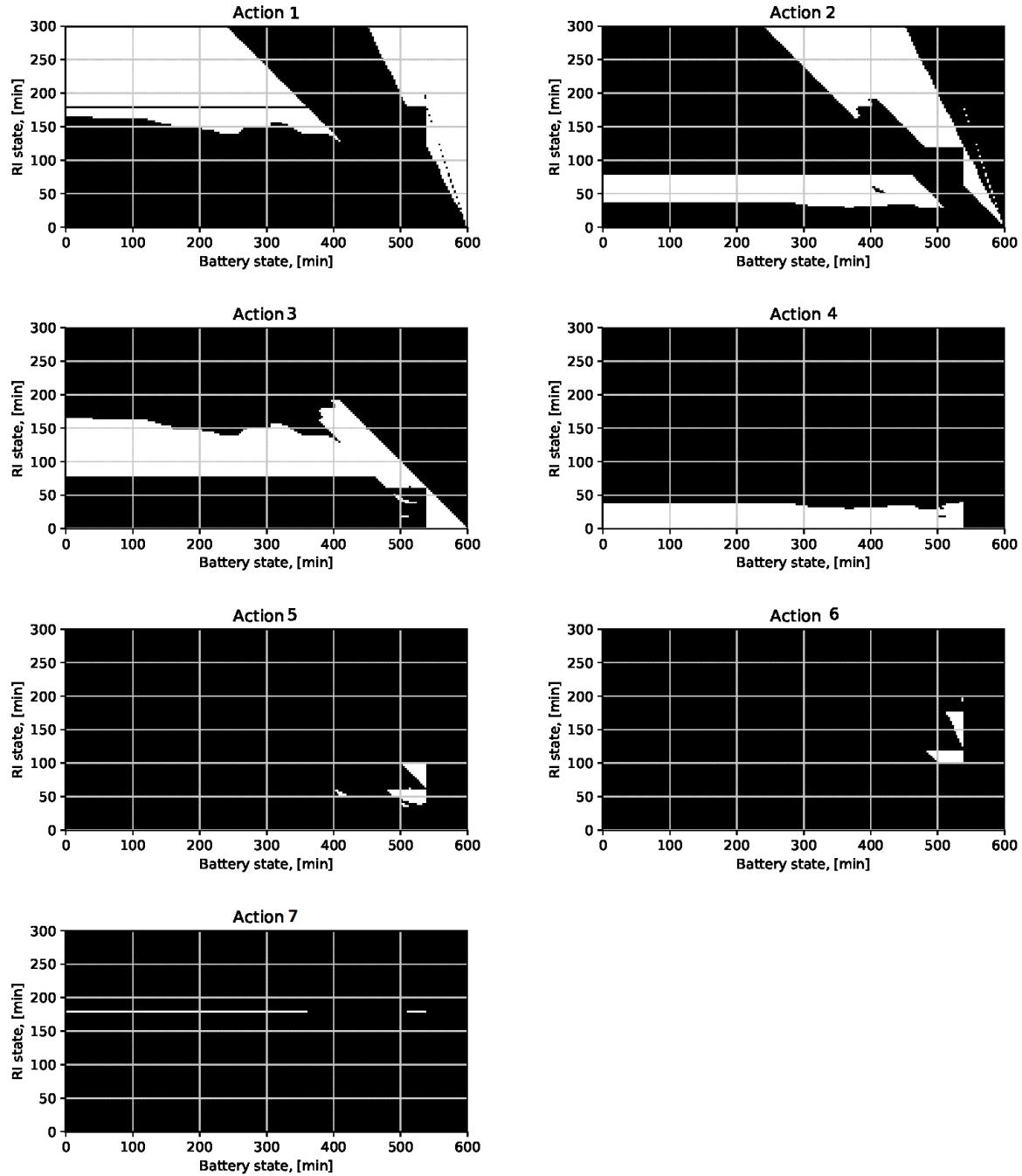


Figure 4.11: 2D slices for each action from the 3D policy solution after convergence.

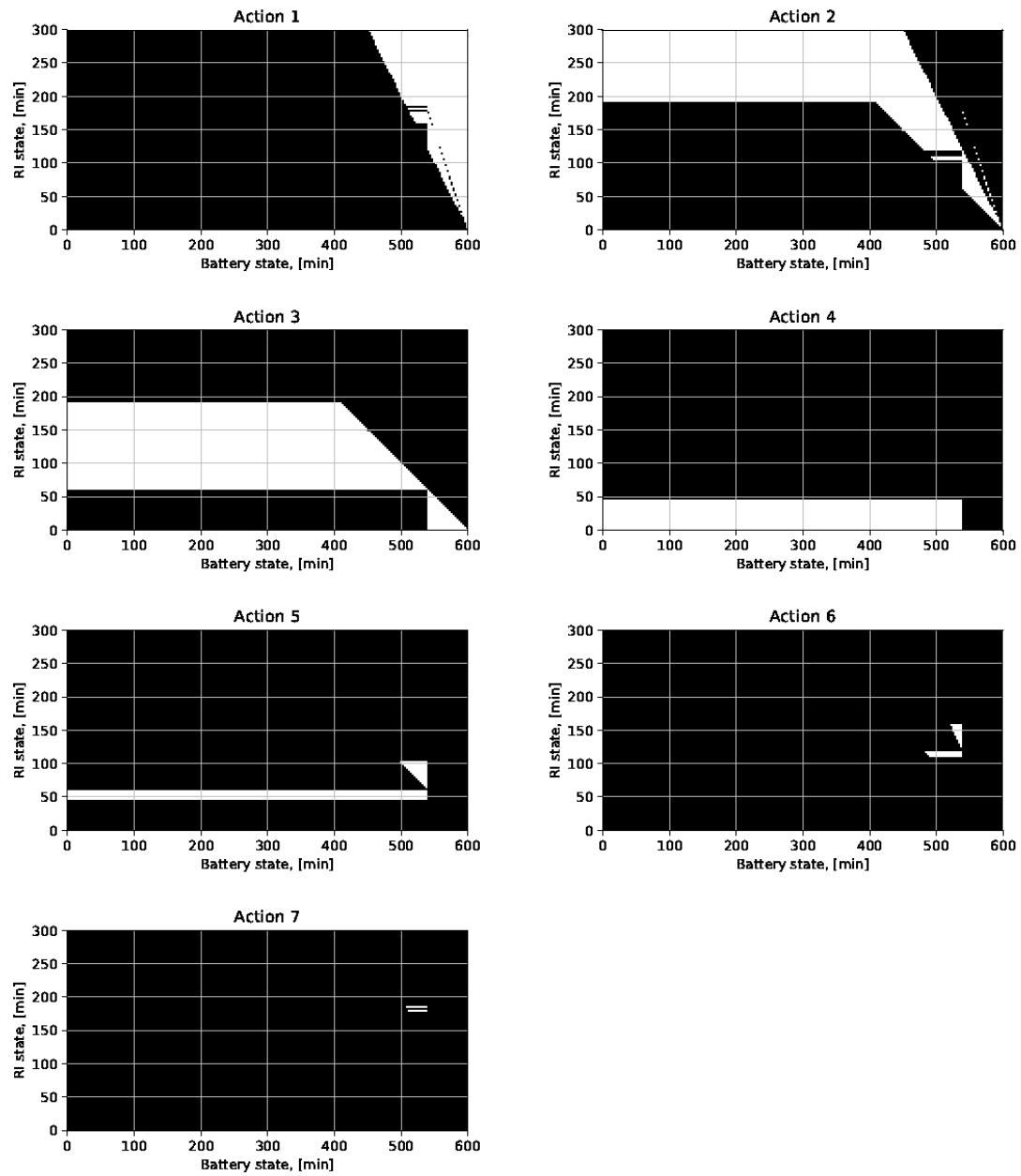


Figure 4.12: Action selection after 1 iteration of the Value Iteration algorithm.

Both Figures 4.11 and 4.12 have the battery state on the  $x$  axis and the workload, or RI state, on the  $y$  axis. Action 1 is selected mainly at the end of the mission, at the end of the battery state axis, as well as when there is large workload. Following Table 4.1, Action 1 sends all vehicles to the RI task, and the Synchronous Rendezvous (SR) interval is adapted to the workload. This action adapts well to high workload and makes sure all targets are visited at the end of the mission. Comparing this to Action 1 in Figure 4.12, the myopic case, the solution haven't picked up yet that if the vehicles are not tasked early on to handle the high workload, the mission will fail at the end as there will be targets left that are not revisited.

Action 1 has a black line in the middle of the white bulk at  $y = 180$ . There are also some black dots going diagonally on the right-hand side. This might look as an inconsistency, however, it can be traced back to the shape of the reward function on Figure 4.7. The black line at  $y = 180$  coincides with all three vehicles being in RI task, which means the overall SR interval is  $60 \cdot t_{SR(min)} = 60$  is the desired interval given as a goal in this mission, so the reward for not deviating from it is higher. Action 7 tasks all 3 vehicles with RI, while the SR interval is kept at 60 so this will be the best situation to chose it. However, given that the rules for Action 1 and 7 are identical for a state with  $RI = 180$ , the reason why Action 7 is given priority is most likely a small rounding error.

Following the RI state space, when the workload is minimal, an action that tasks all vehicles with search should be selected. This is the case for Action 4, while the interval is set to  $t_{SR(min)} = 60$ . When increasing the workload, the optimal action moves to Action 2, which corresponds to two vehicles tasked with RI, one with search and an SR interval adapted to the RI state. The reason Action 3 is not selected, which sends two search vehicles instead, is because the SR interval can be kept near 60 and get higher reward for that with Action 2. Increasing the workload moves the optimal action from 2 to 3.

If we follow the battery state, there is a clear transition from actions 2 and 3 to 1. Action 1 takes all vehicles into search at the end of the mission. Action 2 is optimal when the mission is approaching the end but it is still not risky enough to lead to mission failure by sending a search vehicle.

Actions 5, 6 and 7 show the tradeoff of having a fixed SR interval as opposed to an adaptive one. They are both selected in very limited cases compared to the first four actions. This suggests that their omission in the model might be beneficial as this will decrease computation load without reducing significantly the quality of the mission.

#### 4.4.4 Input Variance Analysis

At the beginning of the simulation, some of the parameters defining the scenario and the capabilities of the vehicles were predefined. This section examines the effects some of these input parameters bring to the overall agent decision making.

The parameters analysed here are the Synchronous Rendezvous (SR) interval duration, the target model, the discount factor, the speed, and the sensor swath width.

Other parameters are not tested as this would require an analytic solution to define the bounds, rather than testing a finite number of variations of the input.

One such important parameter is the discretisation of the state space. The complexity will reduce if the state space is discretised in a more coarse way. However, the resolution depends on the assumption that the value function is true for all of the states that are lumped together. There are more precise methods than discretisation to define this boundary, and be able to analyse the impacts, such as value function approximation. However, the implementation is more complex and the SR-MDP method aims to show a proof of concept, and an uncomplicated solution and representation was selected.

Another parameter that is not analysed is the number of vehicles. Since the action space is directly related to the number of vehicles and how they meet for the SR, changing the vehicle number would require a new action model. At this stage, it is not scalable.

The transition probability is set to explore only ten future states. This requires a truncated Gaussian kernel and reduces the predictive ability of the model. Increasing the number of future states to explore, will make the model more realistic. Since the target model is not based on real data, this accuracy is not considered that important for the SR-MDP. This is also true for the approximation used to calculate the TSP shortest path, as well as the reward model. These parameters all depend on the mission goal and the designer's choice for tradeoff.

Changing the SR interval duration means that the vehicles will have time for more (reducing the SR interval) or less (increasing the SR interval) meeting points. Figure 4.13 gives a comparison of the SR-MDP model run with  $t_{SR} = \{45, 60, 75\}$ . The data for the figure is obtained based on the same method used to produce the results for Figure 4.11 and  $t_{SR} = 60$ , and the analogous figures created for the other values of  $t_{SR} = \{45, 75\}$ . On the  $x$  axis is the action number. The  $y$  axis gives the number of states where the particular action was selected. Going back to Figure 4.11, this is represented by white colour on the plots for selecting each action.

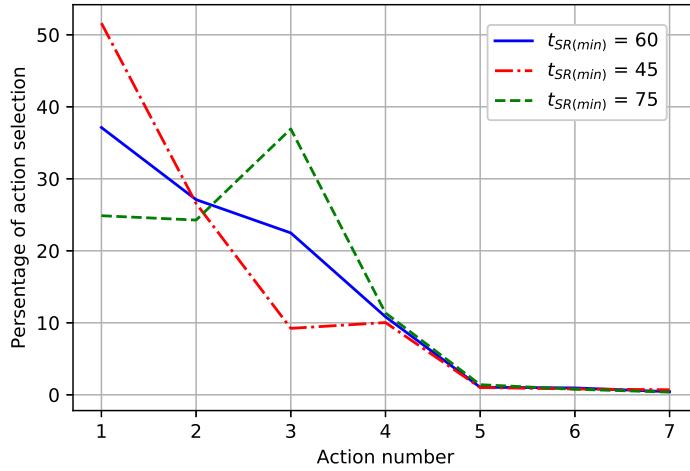


Figure 4.13: Rendezvous Point interval duration analysis.

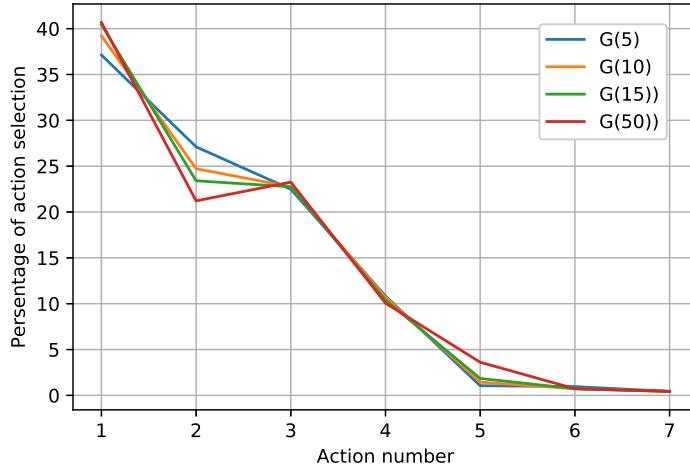


Figure 4.14: Average number of targets per unit area analysis.

Comparing the action choice when varying the SR interval duration from 45 (red line with dash and dot) to 75 (green dashed line), there is a significant difference in the choice of Actions 1 and 3. When  $t_{SR} = 45$ , Action 1 is favoured and for  $t_{SR} = 75$  Action 3 is favoured. Going back to Table 4.1, when the SR interval shortens, Action 1 is selected, as it tasks all vehicles with Reacquire-Identify (RI). This is needed so the vehicles can handle the high initial workload in the short interval between SRs. In contrast, when increasing the SR interval, the initial workload can be managed by less RI vehicles and more can be used for exploration, hence preferring Action 3.

Figure 4.14 shows how the model solution changes by increasing the number of target per unit area. The plots correspond to Gaussian target distribution of  $\mu = 5, 10, 20, 50$  and  $\sigma^2 = 1$ . The variation of the solutions is not as significant as would have been expected after the results in Chapter 3. The

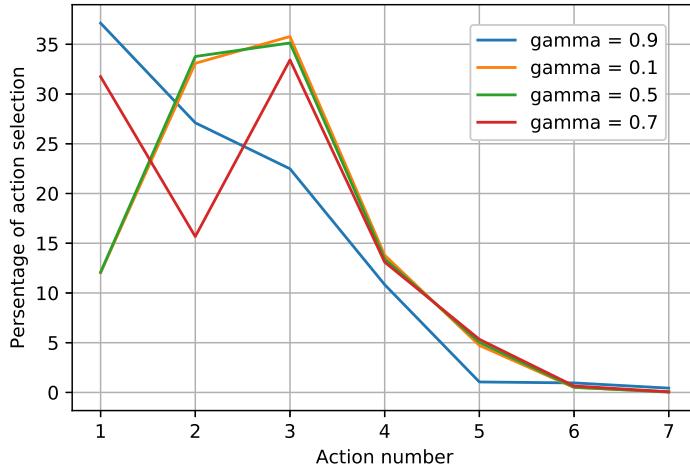


Figure 4.15: Discount factor parameter analysis (see Equation 4.3).

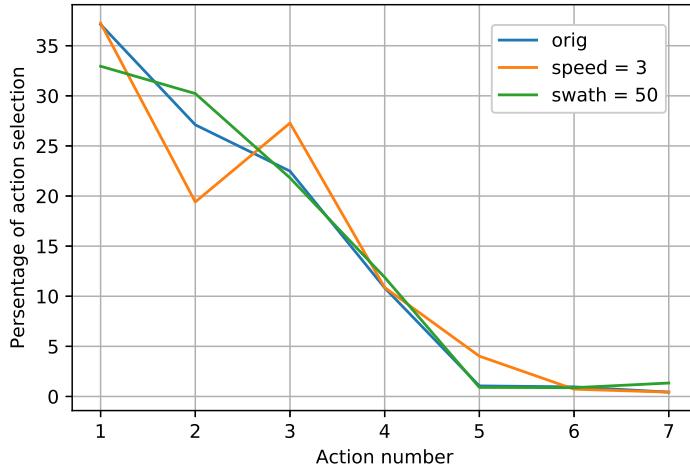


Figure 4.16: Vehicle speed and sensor swath analysis.

reason is partially due to the approximate method used to calculate the shortest path (as opposed to running multiple Monte Carlo simulations as in Chapter 3 and taking the full variance of the uniform distribution into account), as well as the truncation of the transition probability by propagating only ten possible states with a limited variance.

The discount factor  $\gamma$  represents how forward-looking the agents in the model will be. The default value used for the simulations is  $\gamma = 0.9$ , or strongly considering the expected reward when making a decision for the very next action. Reducing  $\gamma$  makes the agent myopic. This is the case of  $\gamma = 0.1$  and  $\gamma = 0.5$ . Increasing its value moves the preferred action from 3 to 1, making the agent to distribute its resources over time better.

Figure 4.16 shows the effect of changing the speed of the vehicles from  $2 \text{ m/s}$  in the default simulation (blue line), to  $3 \text{ m/s}$  (orange line). Changing the swath width of the sensor from  $100 \text{ m}$  in the default simulation is changed to  $50 \text{ m}$  for the green plot. Variations in the speed bring significant change in the preferred action throughout the mission. This is an important result and the speed sensitivity shows that a constant speed model is not a good assumption as it will be hard to achieve in a real and dynamic environment.

#### 4.4.5 Complexity and Convergence

The major problem with any Markov Decision Process (MDP) algorithm is the intractable computation required for any real world problem due to continuous or infinite action and state spaces. As opposed to approximate techniques, the Dynamic Programming method used for the Value Iteration algorithm provides an optimal solution at the cost of doing a full sweep through all the states. The reason this is applicable for the SR-MDP is the representation of the problem in discrete and very small state and action space. The complexity of Value Iteration has a quadratic growth with the number of states and linear with number of actions:

$$O(|A||S|^2) \quad (4.16)$$

The state space for the SR-MDP model is  $\mathbf{S}_1 \times \mathbf{S}_2 = 300 \times 600 = 180000$  and the action space is 7. The computation time for a single iteration on an Intel Core i7 Processor takes 3.5 minutes. A single core was used for the computations so this process can potentially be sped up by multithreading.

The Value Iteration algorithm terminates once the difference in the value function between iteration cycles is smaller than a predefined threshold, see Algorithm 2.

Figure 4.17 follows the variation of the maximum component of the value function,  $\max(V(s_{t-1})) - \max(V(s_t))$ , (x axis) and the overall change in policy in percentage,  $\text{diff}(\text{argmax}(V(s_{t-1})), \text{argmax}(V(s_t)))$  (y axis) between iteration cycles. Each dot on the graph is a difference between iterations. After one iteration, the maximum value difference is near 250 and the change in action selection from a myopic policy is more than 25%. After four iteration, both the value function and the policy change reduce to zero so no change is observed after further iterations. The computation time to reach termination is reached after about 14 minutes. Although this performance is not suitable for a real time mission, optimisation of the code can allow reduction of this time.

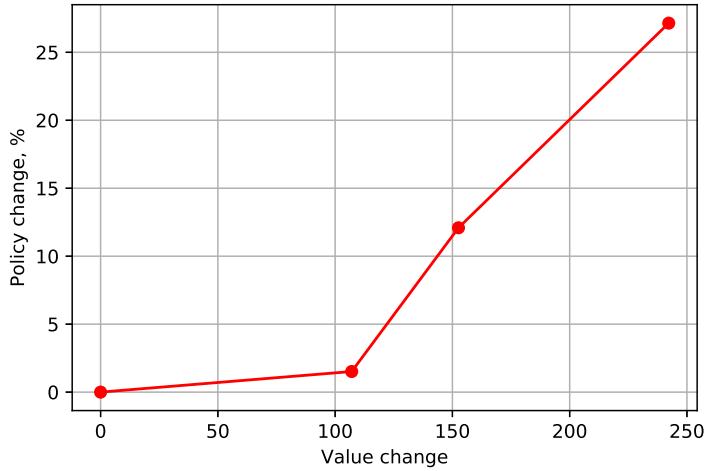


Figure 4.17: Value Iteration termination - convergence condition based on small change in the value function. Axis  $x$  is the value function maximum change between iterations and axis  $y$  is the percentage of action change between iterations.

Figure 4.18 gives a combined plot of how the SR-MDP converges when changing the values of the parameters analysed earlier in this chapter.

#### 4.4.6 Contributions

The main contribution of this chapter is the representation of a multi-AUV Mine Countermeasures (MCM) planning problem with a very small state space MDP. This allows using an optimal solution method in low computation time. As a result, this Markov Decision Process (MDP) model has a potential to be applied to an underwater robotic network and provide a real-time output for the agents. Some key points of the model are:

- State space mapped from physical space to time to reduce dimensionality
- Action space is defined in a way that brings the problem from multi-agent to a single agent by explicit task allocation for all vehicles.
- The interval between the elements of  $t$  is not always the same, as in most MDP formulations, however, the problem is still discrete and the framework applicable.

Some heavy assumptions in this model are made:

- Assuming the environment dynamics are known.

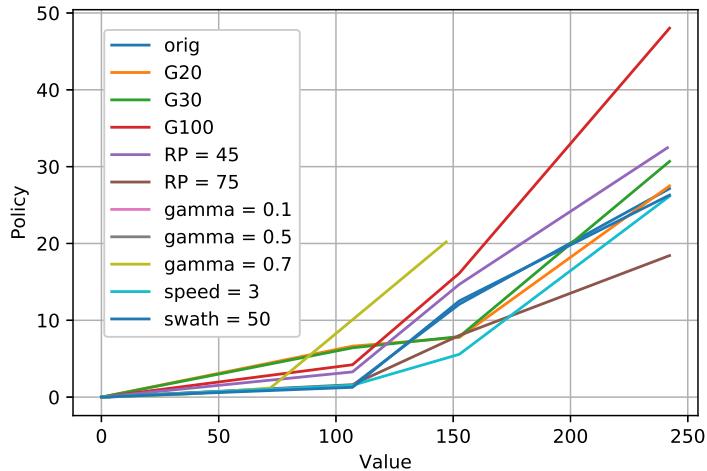


Figure 4.18: Convergence variation

- Using a uniform distribution to represent the number and position of the expected targets,
- Approximate expected shortest path calculation.
- Sensor is treated as blackbox with consistent specifications.
- Automatic Target Recognition (ATR) is available.
- No navigation disruptions are observed during the mission — no currents to change vehicles tracks or to make them use additional battery or change speed.

## 4.5 Summary

This chapter develops a decision-making method that balances the tradeoffs of multi-AUV Mine Countermeasures (MCM) mission and the Synchronous Rendezvous (SR) scheduling. As a result, the agents can select an optimal action to maximise the desired output of the mission while complying with hard and soft constraints. In the mine hunting scenario, the set goal is to maximise a search area until the battery of the vehicles is exhausted. One constraint is that the detections found during the search phase need to be revisited. In order to exchange status and contact data, the vehicles meet multiple times throughout the mission, at a Synchronous Rendezvous (SR). The time and place for this SR also needs to be decided autonomously. This resource management tradeoff of covering more area while revisiting contacts and scheduling meeting points is what the Synchronous Rendezvous (SR)-Markov Decision Process (MDP) is solving. It provides an optimal action policy for the agents, that takes into

account estimates for all possible future states, up until the end of the mission.

The main contribution of this chapter is that the SR-MDP model is capable of computing a solution in manageable time, with potential to reach real time execution. This is achieved by discretising the action and state spaces without losing vital mission information. The other critical element is that utilising the SR method made the decision making intervals very few throughout the mission and this allowed defining the SR-MDP with a finite horizon.

This work can be expanded by using an actual model for mine field distribution or making the agents learn a model by turning the problem from planning to Reinforcement learning (RL). Another aspect for improvement is using a different solution method, rather than Value Iteration. Novel approximate solutions are available and can significantly reduce the computation time to reach real-time execution. Finally, validation of the model, or parts of it, in a real experiment, will bring insights into how suitable the input functions are and what can be updated to make the MDP applicable for a real MCM mission.

# Chapter 5

## Simulation and Experimental Validation

Validation in trials for marine robotics algorithms is not always possible. The reason is the high cost of equipment, such as AUVs, surface vehicles or additional infrastructure. Often, to test autonomy features, the vehicles need to have advanced capabilities in terms of platform design and sensory suite. Buying or building such platforms is expensive. In addition to that, the physical size of the vehicle or the area of interest for data collection might require additional equipment such as hiring and managing a ship and operation crew. This is the reason much of the work in autonomy is validated in simulation or only partially tested in trials.

### 5.1 Simulation Environment

There are software architectures that serve as 'robot frameworks', providing operating system capabilities, such as managing low-level control and software functionality, and middleware capabilities, bridging the operating system with additional applications. They provide suitable simulation testbeds for development before deployment. The two most common robot frameworks used in the underwater community are Robot Operating System (ROS), usually with the UWSim extension [105], and MOOS-IvP (Mission Oriented Operating Suite - Interval Programming) [106]. Both of them are open source. Neptune [107] is another well accepted framework, adopted mainly by defence institutions, since it is proprietary. Some other available open-source frameworks exist, with more limited exposure in the

community, such as Dune [108], Rock [109], AVA[110].

The tool used for simulation validation in this thesis is MOOS-IvP. Its development started with marine autonomy in mind and subsequent range of capabilities were developed. ROS has similar design and functionality, however it has a larger community, both in underwater and other robotics domains.

This section starts with a general overview of the MOOS-IvP tool. Then, the implementation of the Synchronous Rendezvous (SR) method is discussed. Partial validation of features and assumptions of the SR approach is available from trials, and results are presented at the end of the section.

### 5.1.1 MOOS Middleware

The MOOS middleware relies on connecting different applications (processes or libraries) through a central database server unit, MOOSDB. This publish (messages the application produces)-subscribe (messages the application consumes) message-passing between processes allows communication and interaction, while keeping the tool design modular. Each MOOS application inherits from the super-class CoreMOOS, which provides common functionality and allows for new user-defined processes to be added and run within the MOOS middleware.

The star-shaped topology of the MOOS middleware, shown in Figure 5.1, has the advantage of a centralised communication hub. All messages pass through a master node, the MOOSDB unit. This makes it easy to detect any failed node and a failed node does not affect the rest of the system, as no messages are sent directly between applications. The disadvantage is that the central unit is also a single point of failure for the whole system and can be overwhelmed if too many messages need to be distributed in the system. This point is analysed in [111], where different robotic frameworks are reviewed and compared. It evaluates communication overhead and message latency as key performance measures. The review gives very unfavourable evaluation to MOOS. MOOS does not even appear in the graphs of the comparison section since the message latency between publisher and subscriber reaches 100 ms, while it is only between 0,1 and 1 ms for the rest of the frameworks (ROS, Yarp, LCM, Player and Urbi). The reason for this delay is the built in serialisation technique in MOOS, which accepts only string and double data formats, rather than more concise binary streams, and also the built in low fetching frequency. The conclusion of the authors about MOOS is that "it is questionable if such a middleware is suitable for real time robotics applications, that need to respond quickly".

The choice of using string and double data format for the message content, and the need for serialisation, aims at making the tool and logs easier to use and include human readable variable-value pairs,

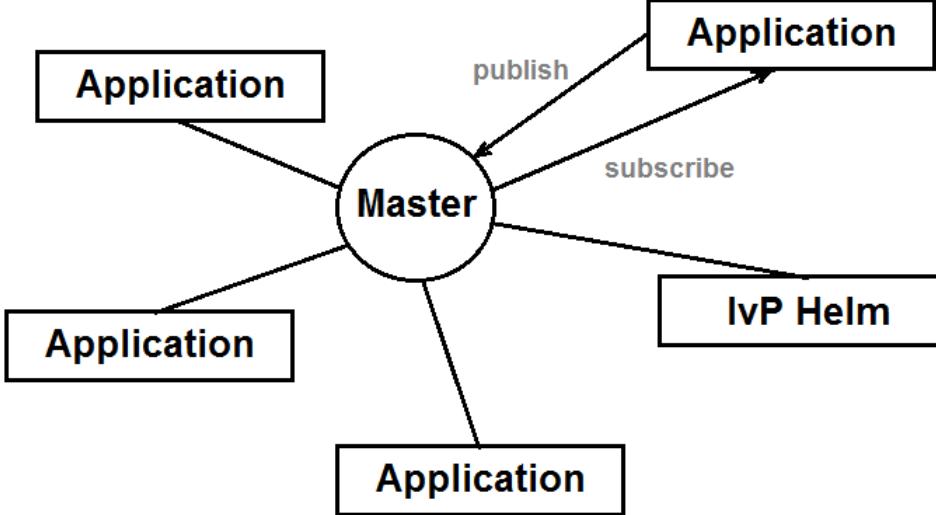


Figure 5.1: A MOOS community representing the publish-subscribe architecture

rather than encoding them. This does increase the overhead, but it is argued in [106] that since the CPU cost for using TCP/IP protocol, used for internal message passing between applications within a single vehicle community, is the same for message sizes of up to 1000 bytes, the overhead from using strings is not frequently experienced. Therefore, we consider the serialisation overhead as not being a constraining factor.

The other criticism in the middleware comparison review [111] is the latency for sending and receiving messages. MOOS has the MOOSDB central server and all applications interact with it to store their output and to subscribe for other application's updates, as opposed to using peer-to-peer connection, such as in ROS. The reason for the limited fetching speed is to relax the workload of the MOOSDB, since it is a single point of failure. Although 100 times higher inter-process communication latency is a disturbing result, the platforms in the underwater environment does not require such quick reaction times as, for instance, an air drone. The reported latency between publisher and subscriber was in the order of 100 ms, which is an acceptable number for slow moving and reacting platforms, such as AUVs. However, this is one reason to drive users in the community to chose ROS over MOOS.

### 5.1.2 IvP Helm Application

The IvP helm provides the core autonomy functionality for the MOOS middleware. It runs as a MOOS application and is connected to MOOSDB, as shown in Figure 5.1. The IvP helm follows a behaviour-based control design, similar to the subsumption architecture, described in Chapter 2.

Each behaviour is a software module that defines a core capability, for instance "follow waypoints", waypoints structured in a typical pattern (for example lawnmower), "avoid obstacles", "go to depth", etc. The behaviours have reconfigurable parameters, such as defining search area, change of speed, change of waypoints. The IvP helm takes into account all the behaviours defined on the vehicles and decides when and which of them are active and how to resolve any competing output commands. The combination of multiple active behaviours to achieve a single global one is a central question in reactive control architectures and the IvP helm resolves this problem using hierarchical decomposition of modes to organise behaviour activation, and multiobjective optimisation method to resolve any conflicts between behaviours. The output of the application gives speed, heading and depth command for the lower control of the AUV. Details of the algorithms and the structure of the IvP application can be found in [106].

## 5.2 Synchronous Rendezvous (SR) implementation in MOOS-IvP

There are substantial differences between implementing the Synchronous Rendezvous (SR) method in a numerical simulation, such as in Chapters 3 and 4, and building MOOS applications to guide vehicle behaviour based on SR rules.

In order to compare the SR method in a numerical simulation and a robotics framework simulation, the relevant assumptions for implementation are discussed. In Chapter 3, the SR algorithms is defined and each step of the numerical simulation explained. Chapter 4 builds on this adaptive SR method by making it non-myopic. The SR-MDP functionality has not been implemented in MOOS and this section considers only the numerical simulation from Chapter 3.

Many of the AUV functionalists were considered a blackbox in the numerical MATLAB simulation from Chapter 3, such as sensor performance, navigation, low-level control, communication, networking. In contrast, MOOS is a middleware which consists of multiple separate processes that are linked through a publish-subscribe database. The way to add new functionality to MOOS is by writing new applications, or overloading the functions in the base class called CMOOSApp. Most of the blackboxed functionalists from Chapter 3 are added to the MOOS simulation through existing or newly written applications. Some of the differences and difficulties in implementation are contrasted further, following the implementation of Algorithm 1 in Chapter 3 and in this chapter:

- The MATLAB simulation implements the SR method as multiple discrete calculations that add up until a threshold parameter is reached (the remaining mission time on line 6 in Algorithm 1). In contrast, the MOOS simulation is continuous in time and allows multiple triggers to enable the SR calculation. While the numerical simulation focuses solely on the SR calculation with the aim to estimate the system's resource loss, MOOS allows to represent and analyse the agents' cooperation. In MOOS, the SR method is only a part of the overall simulation - it is used as a set of rules to define the scheduling of each vehicle's behaviours. The result of these key differences is that the numerical simulation assumes a very simplified and unrealistic model of the sensor (perfect probability of detection and classification of targets, no sensor disruption as a function of movement or distance), communication and networking (no package loss or collision), and navigation (speed and heading are the same as command). In contrast, MOOS allows for easy custom choice for the probability of detection,  $P_d$ , and probability of classification,  $P_c$ , which are central parameters for an MCM mission. The tool also provides a realistic representation of navigation and vehicle dynamics, as well as communication modelling.
- The calculation of the search area in the MATLAB simulation (line 8 in Algorithm 1) gives the location of the vehicles at the time they need to meet for the next SR. The equation uses predefined scenario and navigation parameters as input. In MOOS-IvP, there are multiple applications that describe each vehicle control and dynamics parameters, as well as an implementation of detecting targets and recording their location, rather than providing all vehicles with a randomly generated list of of  $(x,y)$  coordinates.
- The search phase in the numerical simulation defines the number and location of targets. It does not take into account how the area is scanned, but instead only generates targets with constant mean and variance (line 9 in Algorithm 1). In MOOS, the search phase, where targets are detected, is simulated by a vehicle scanning a surface, so that the navigation and the sensor integration are taken into account. This is one of the key improvements of having a MOOS simulation over the numerical results. Such implementation provides a more realistic simulation of the MCM scenario and gives an insight of how the search phase will be executed in a trial.
- At SR, a MOOS application is required to collect and process the target information, followed by scheduling the next SR point and defining the path of the vehicles. Two new applications are developed to enable this - uFldMergeHazard (Algorithm 3) and pSynchronousRP (Algorithm 4). First, all vehicles need to collect the location information of MLOs they find during their search phase, then share the contents of their reports between each other (uFldMergeHazard). A merged list of detected hazards (between all vehicles) is the output that is sent to the schedul-

ing procedure (pSynchronousRP). This is not factored in the numerical simulation but results in numerous issues relevant to multi-agent systems, such as task allocation, consensus problem, Medium Access Control (MAC) implementation, reliability of communication, synchronisation, fault tolerance. Some of these issues were addressed and resolved in the simulations presented further in this chapter, however some are more complex issues that are still open research questions for the community of cooperative underwater robotics. This is the reason the MOOS-IvP simulation is a partial implementation of the SR method from Chapter 3 and hence provides limited method validation.

- Calculating Reacquire-Identify (RI) time is done on lines 10 and 11 in Algorithm 1. In MOOS, once the target list from all vehicles is received in the scheduling application, it is processed (correct format, removing duplicates). Then it is sorted using Nearest Neighbour algorithm. Since the points that need to be sorted are a low number, a greedy sorting algorithm is sufficient. It assigns the closest point on the list, based on its  $x$  and  $y$  coordinates. The initial point is the current SR location. The RI time is calculated based on this shortest distance to travel between the target points. The RI calculation is done separately on each vehicle, which allowed for a decentralised system, rather than relying on a master-slave setting with one vehicle distributing the information to the rest. This resulted in desynchronised SR scheduling due to consensus and networking issues. It is further discussed after simulation results are presented.
- The SR scheduling procedure distributes the search and RI resource (based on the available AUVs). The case where more RI resource is required than available (highly cluttered environment) is not considered in the numerical simulation (lines 12 and 13 in Algorithm 1). The case is not considered for the MOOS simulation either. There is an issue passing a long list of targets between vehicles. This is due to the choice of message content (sending a long string) of the middleware design and the application responsible for the communication simulation (Goby [112], an add-on for MOOS-IvP, responsible for the link layer and acoustic telemetry).
- Lines 15 to 30 in Algorithm 1 give the SR time scheduling based on whether  $t_{RI}$  is between the predefined  $t_{SR}$  limits. Based on this, the number of vehicles tasked with search and RI is decided, all contacts are revisited for further identification and ultimately, the time for the next SR is calculated. In contrast, the MOOS simulation assumes predefined tasks: two vehicles in search and one vehicle in RI. In order to implement the flexibility from the numerical simulation, an additional application to solve the task allocation problem had to be designed, however this is not currently done. The MRTA is still an open question in cooperative underwater robotics.  $t_{SR}$  in MOOS is defined based on the  $t_{RI}$  for a single vehicle visiting all known targets.

- Finally, the calculation of the resource loss, used for evaluating the SR algorithm in Chapter 3, is not required for the MOOS-IvP simulation as the travelling time to the meeting point is part of the vehicle's path. The loss can be extracted later when analysing the mission logs — this is the time spent in between different behaviour triggers. The loss is not optimised (eg. the vehicle waypoints can be calculated based on where the next SR will be and select the closest end point for the route). In contrast, the numerical simulation assumes loss proportionate to travelling half the distance between the end point of the designated mission area and the SR.

The main contributions for the MOOS simulation that have been developed are: collecting a merged list of targets from all vehicles (Algorithm 3), calculating RI and SR time (Algorithm 4), linking the application to the vehicle communities, enabling synchronised mode and behaviour triggering, implementing TDMA MAC scheme to ensure all vehicles receive the same information.

---

#### **Algorithm 3** uFldMergeHazard class - creates merged list of targets between vehicles

---

```

1: procedure HANDLEMAILDETECTIONREPORT(uFldHazardMgr)
2:   create a list of unique detections (own targets)
3:   return request classification report
4: end procedure
5: procedure SENDGLOBALREPORT(IvPBehavior)
6:   process targets from own lists
7:   return send own report to other vehicles
8: end procedure
9: procedure SENDGLOBALREPORT(pAcommsHandler)
10:  process targets from other AUVs
11:  return send merged report to other vehicles
12: end procedure
```

---

#### **Algorithm 4** pSynchronousRP class - schedules Synchronised Rendezvous dynamically

---

```

1: procedure HANDLEMERGEDREPORT(uFldMergeHazard,pAcommsHandler)
2:   check if merged report is up to date
3:   string of data to vector of points and calculate SR interval using greedy shortest path
4:   define new vehicle tracks
5:   return send new paths to vehicle communities
6: end procedure
7: procedure SHARESYNCHRONOUS RENDEZVOUS(pHelmIvP)
8:   checks if all vehicles at the SR have received the next meeting point
9:   return send the next SR
10: end procedure
11: procedure RESETTARGETLIST(pHelmIvP)
12:   return removes visited targets from list and resets all triggers in relevant applications
13: end procedure
```

---

### 5.2.1 Simulation results

MOOS-IvP is used as a partial validation environment for the Synchronous Rendezvous (SR) method. It allows building a realistic simulation taking into account vehicle and sensor dynamics. Functionality, such as task allocation, communication and networking, which is assumed deterministic and lossless in both Chapters 3 and 4, is built and implemented for the MOOS simulation. While there is no statistical data collected from the resulting simulations, due to the complexity of fully implementing the SR method, the partial implementation highlights some bottlenecks, that are sometimes decoupled from mission planning problems, as seen in Chapter 2. This section evaluates most of the assumptions made in previous chapters, however it leaves some advanced features, or components of the mission scenario, that require further research, as future work suggestions. Some examples include:

- Dynamic task allocation, such as auction-based mechanism. Only a limited rule-based method is implemented in this simulation.
- Reacquire and identify procedure aims at collecting additional target information. In order to define the vehicle's manoeuvre around the target, the sensor characteristics, the environment state and the vehicle capabilities need to be taken into account. In the simulation, the vehicle is only passing by the target.
- ATR and probability of detection and classification models.

The implementation of the SR scenario described in Chapter 3 is implemented in MOOS and stills from the graphical interface are provided to illustrate the simulation. Figure 5.2 gives the initial search path of the vehicles after deployment — two AUVs follow a lawnmower path and one is aiming for the meeting point, defined by a polygon path. The main panel shows the vehicle dynamics and the path defined by the active behaviour. This still is from the postprocessing tool in MOOS-IvP and the panels on the left show the heading and speed decision space of the behaviours. The top one shows the objective function for the loiter behaviour (predefined in the MOOS-IvP library) and is used to represent the SR point. The bottom two plots show the waypoint behaviour (predefined in the MOOS-IvP library), following a lawnmower pattern. The function maximum (coloured red) defines the heading and speed of the vehicle.

The vehicles' paths, and the first SR after deployment, are predefined in Figure 5.2. There are three vehicles (jake, kasper and lemon) simulated with their tracks following a lawnmower. The still gives the moment when the vehicles have finished with the search and move towards the meeting point. In MOOS, the SR points are defined by a polygon for the vehicles to circle around (loiter behaviour

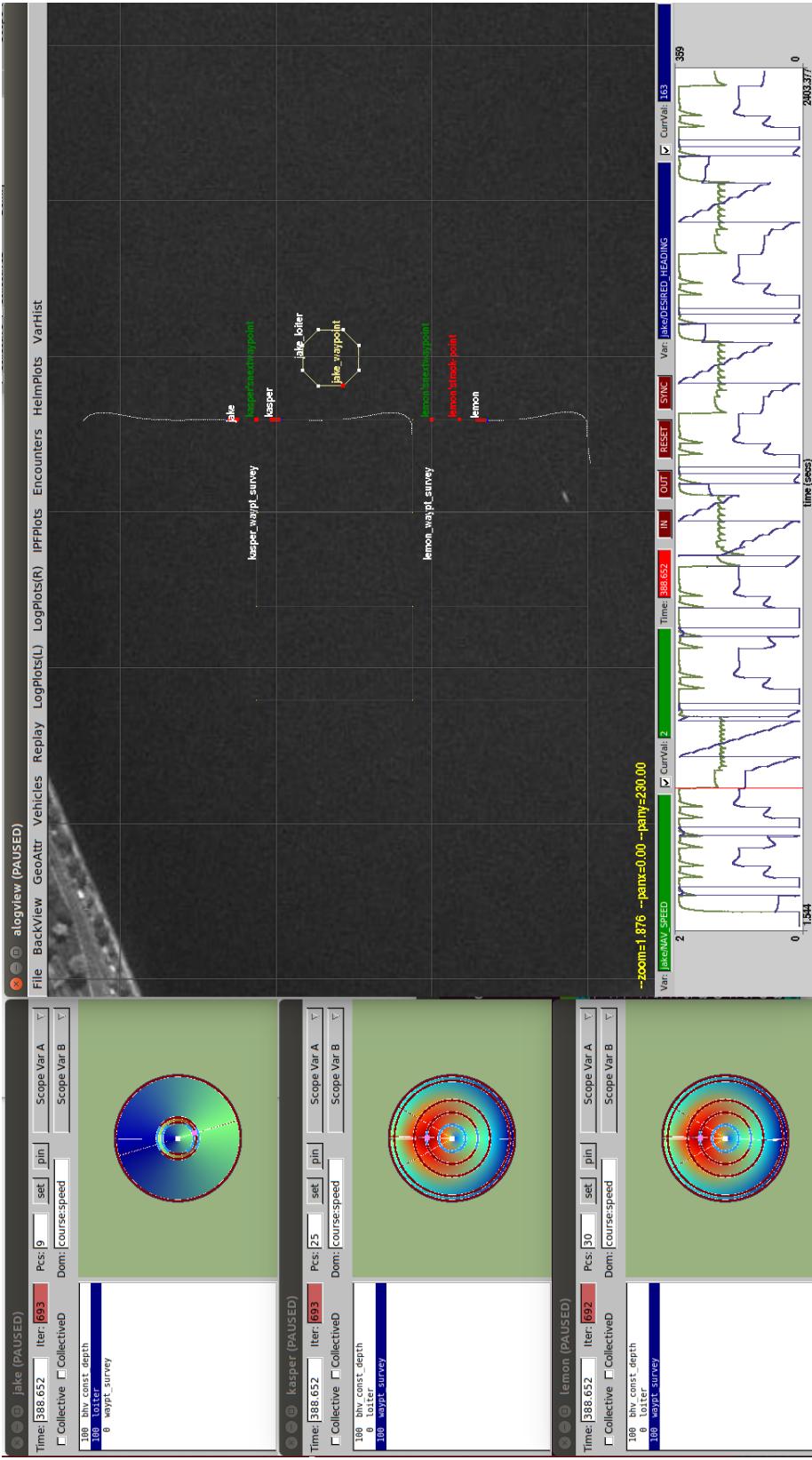


Figure 5.2: Starting path — predefined lawnmower search for all vehicles. On the left is the behaviour plot showing the heading and speed state space of the vehicles during search mode (waypoint behaviour). On the right is a still from the simulation (post-processing).

in MOOS-IvP) while sharing information and deciding their next actions. This is not factored in the numerical simulations from the previous chapters. However, most vehicles cannot hover at a stationary position and constant movement is required. One addition of the current simulation would be to add an obstacle avoidance behaviour. There is a built in one in MOOS and the multiobjective optimisation from the IvP Helm is designed to reconcile conflicts, such as the one that would arise between following waypoints during the loiter behaviour and avoiding close proximity with the incoming vehicles.

Figure 5.3 shows all three vehicles at the SR point. This is where the applications following Algorithms 3 and 4 define the task allocation, vehicle paths and the next meeting point's time and location. The duration of the loiter behaviour was empirically defined. Some of the consideration came from the duration required from the communication and networking application (`pAcommsHandler` — sets up the TDMA protocol). Sometimes repeated transmissions were required for the messages and reports to be shared between the vehicles. This was due to vehicles arriving at the meeting point at different times and the target reports each vehicle had were mismatched. The other consideration for extending the wait at the SR was due to discrepancy of the resulting path. Some vehicles arrived earlier and the buffer time interval was introduced to account for mild unsynchronised arrival. The duration used in the simulation is 200 s. Following the evaluation procedure defined in Chapter 3, this should be factored in the loss calculation. However, more advanced MAC protocol design can remove this problem, by defining better commutation slots and dynamically defining the interval required for data sharing.

Figure 5.4 gives the progression of the vehicles after the first SR. There are two vehicles following lawnmower tracks (`jake` and `lemon`), and one vehicle following the shortest path defined by the shared list of found targets. The targets are not shown on this figure, as the still is from the postprocessing tool. Figure 5.5 shows another still from a similar mission, but during runtime. The targets are visible (some of them are mines (triangles) and some are false alarms (squares)). The analysis of successful detection and classification is not factored in this work, so it is not important what targets define the path of the RI vehicle (`kasper`) in the simulation. During runtime, the sensor is simulated as a rectangle shape on both sides of the vehicle. This is simulating a Side-Scan Sonar (SSS) illumination. When the vehicles are turning, or move with high speed, the area of the sonar swatch width becomes red - the data will be discarded and any present targets are not detected. Otherwise, the colour is grey for the normal operation of the sonar. This is similar to a trial SSS operational considerations.

After looking at a successful SR scheduling in the figures above, there were some commonly experienced problems with the defined settings and implementation that give insight into the issues that would arise if the SR algorithm is to be applied in a sea trial.

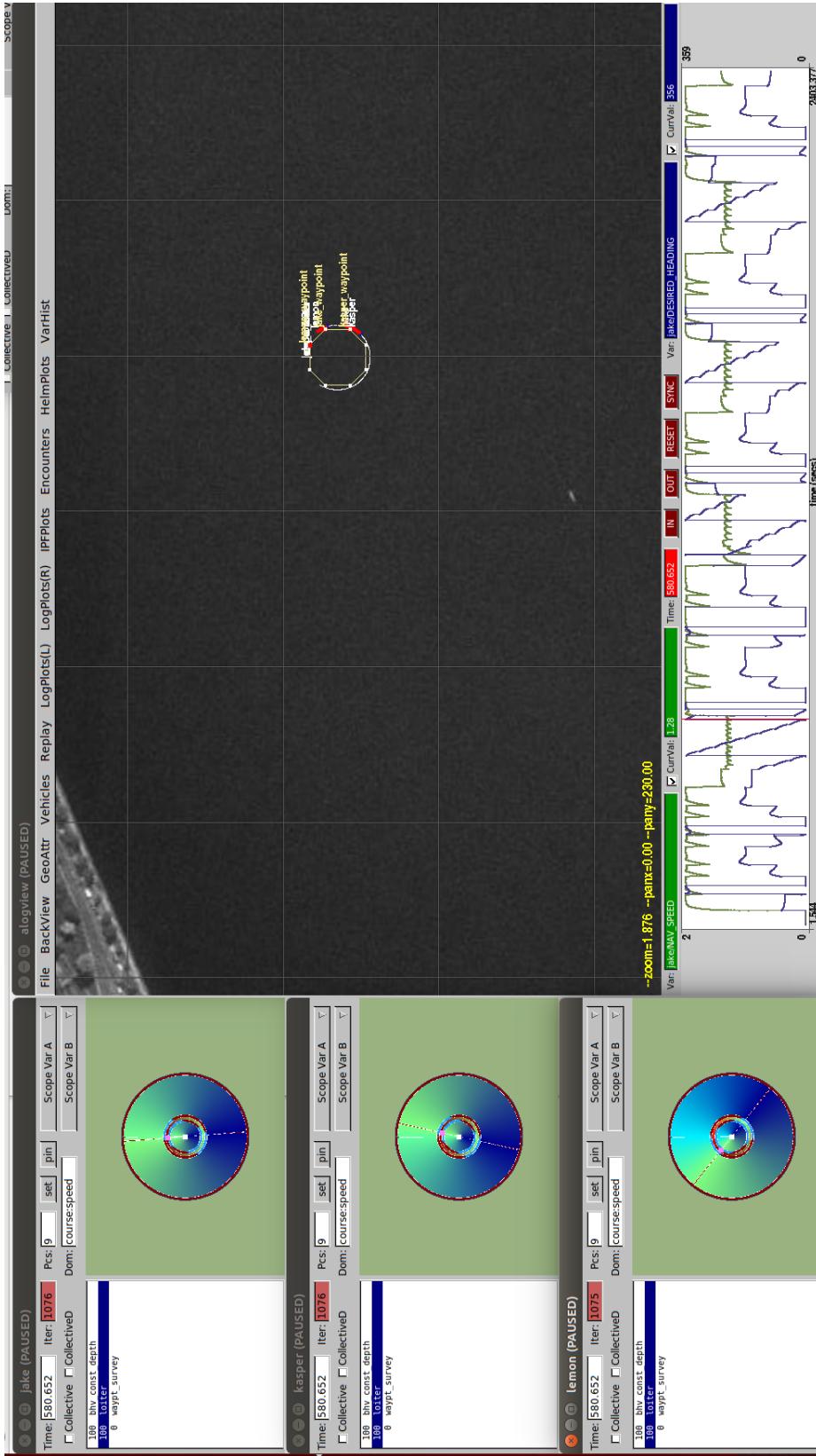


Figure 5.3: First synchronous rendezvous point - predefined location and time. On the left is the behaviour plot showing the heading and speed state space of the vehicles during SR mode (loiter behaviour). On the right is a still from the simulation (post-processing).

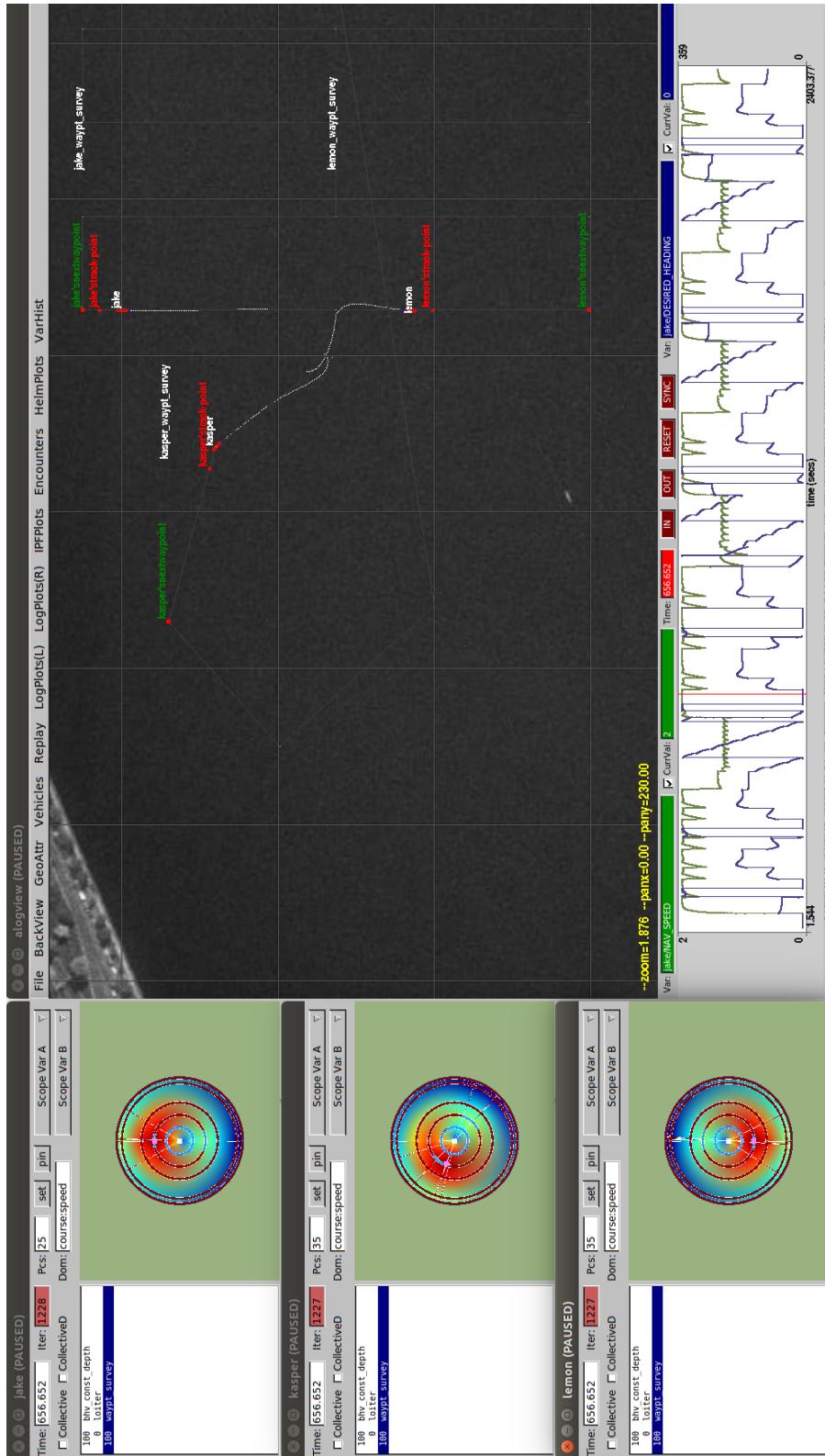


Figure 5.4: After the first SR, the vehicles adapt their paths based on the targets that were found during the search phase. One vehicles is doing RI, following all detections, and 2 vehicles are following a lawnmower path. All vehicles are following the waypoint behaviour, which was configured dynamically based on parameters calculated in uFldMergeHazard and pSynchronousRP classes. (post-processing)

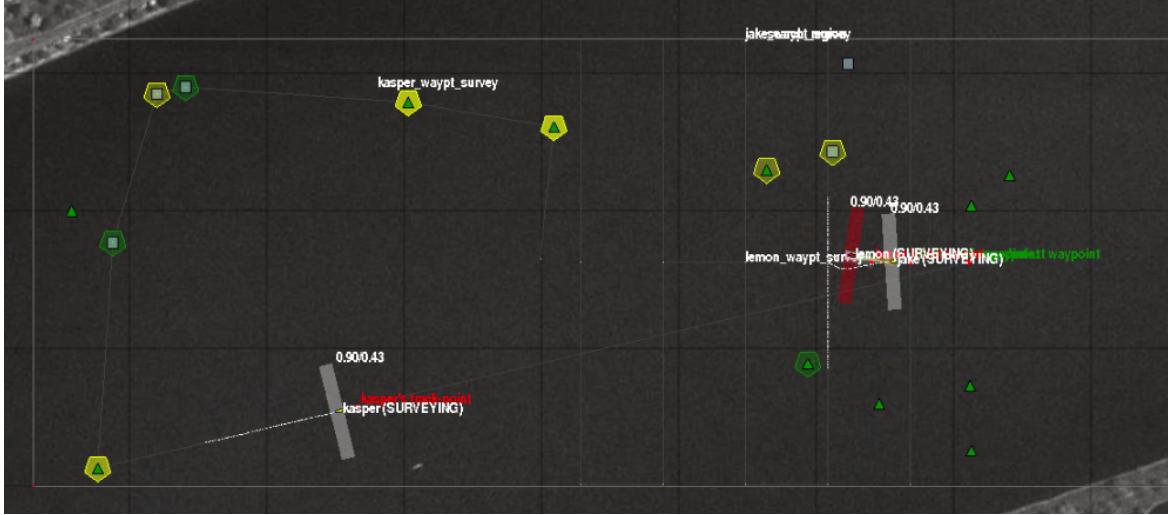


Figure 5.5: After the first SR, the vehicles adapt their paths based on the targets that were found during the search phase. One vehicle is doing RI, following all detections, and 2 vehicles are following a lawnmower path. All vehicles are following the waypoint behaviour, which was configured dynamically based on parameters calculated in `uFldMergeHazard` and `pSynchronousRP` classes. (runtime)

Figure 5.6 shows the path of the two search vehicles being significantly different. The vehicle on the bottom continues for two more lawnmower legs compared to the one above. As a result, the top vehicle will finish its search task and aim for a SR, while the other search vehicle will still be employed in search and reach a different SR position at the end of its path. The aim for scheduling meeting points for the vehicles is to synchronise the execution time for their tasks. However, mismatch like this results in a severe disruption of the mission. The reason for this desynchronisation is that the vehicles received different target reports during the previous SR. This can easily be avoided using a master-slave approach, where a single vehicle collects the data, defines the tasks and paths for all vehicles, and distributes the decision. However, this brings a single point of failure and does not reduce the communication load significantly, as the vehicles need to send their data, receive the merged target report, and send an acknowledgement message. There is also the risk that the master vehicle does not receive the correct data. Instead, the approach used for data exchange in the MOOS simulation is that all vehicles receive each others reports and compute the shortest path between the detections on the list. Then, they exchange the result and compare for discrepancies. In case there is a mismatch, the longest report is selected, assuming that it is the full one. The vehicles acknowledge the final report choice is the same for the whole network. This follows the MOOS application described in Algorithm 3. There were multiple occasions during the simulations, when the communication was unreliable, and the vehicles produced different merged reports. If the procedure to unify the reports is successful, this problem is fixed. However, confirming that the vehicles have agreed upon the final report, or achieving reliable communication, brings the consensus problem, which is common for multi-agent



Figure 5.6: Mismatch in the synchronous rendezvous time and vehicle paths. The problem is due to the merged reports which did not get synchronised. Some vehicles calculated their path and SR based on a partial report. This happens when a message does not get through or when a vehicle is late to arrive at the SR.

systems. During the tests, there were simulations that resulted in failed consensus and mismatched paths, such as the one in Figure 5.6.

Another case, where mismatched paths were present appeared when the list of targets was too large. This limitation was due to the string that had to be passed through internal and external messages. Figure 5.7 shows an experiment, where the RI vehicle followed the path from its previous SR window, rather than update to the new one. This was the result of a string message containing the list of targets that was too long to be processed. The vehicle repeated it's previous path, however a contingency can be added for the vehicle to surface or to signalise the problem. This has not been pursued further.

Due to implementation and operational issues, such as task-allocation, communication and networking, consensus and lack of fault tolerance, the SR method was not implemented as the MCM scenario described in Chapter 3. However, elements of the algorithm were tested in MOOS. The observations from the simulation tests suggest that bringing the platforms together gives an opportunity for robust communication and networking windows. It was confirmed that negotiating task allocation requires a reliable channel to pass messages between the nodes. The rendezvous point can enable and secure

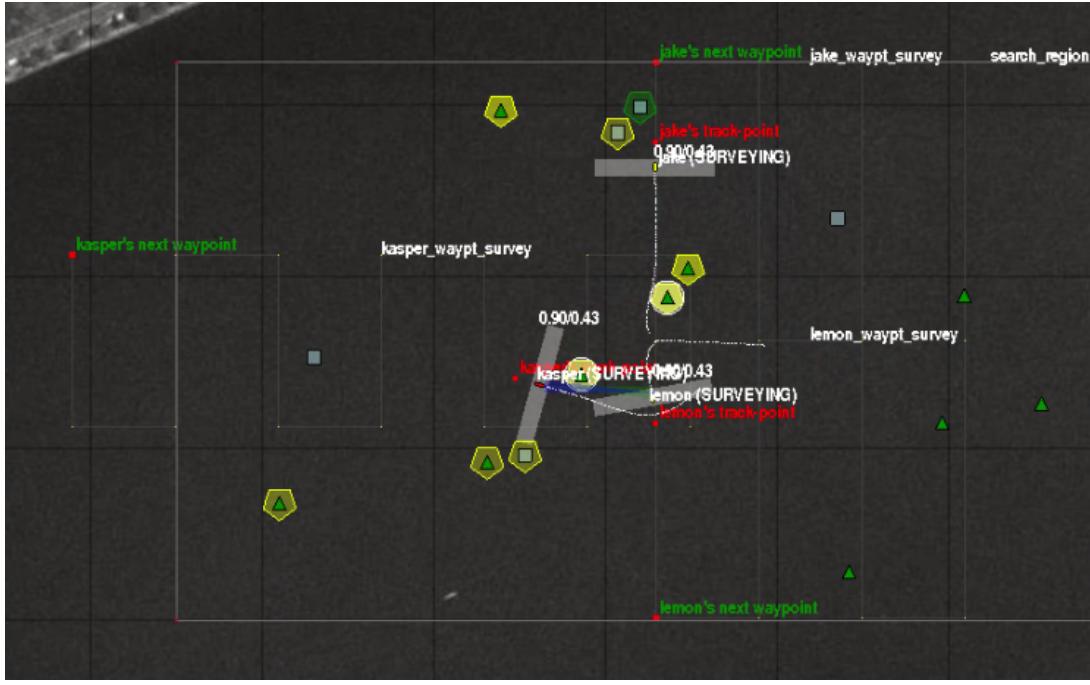


Figure 5.7: Vehicle does not get an updated track, follows its previous track instead. The problem is caused by the same issue as in Figure 5.7 - the merged target report was not updated. In this case it was caused due to salutation of targets. The merge list did not get through at all.

sufficient condition for this exchange. The MAC protocol and the distance between vehicles can be adapted depending on the capabilities of the vehicles and their equipment.

Building the MOOS-IvP simulation helped identify bottlenecks and additional loss terms that were not factored in the numerical simulations from the previous chapters. An example is the additional time at SR for information sharing. Another significant issue is that a mechanism is needed to manage any desynchronisation issues. This work assumes stationary environment and paths matching the planned ones. However, when currents are added, the vehicles need to adapt their speed and heading in order to cancel these deviations, which would affect sensors and the battery level.

### 5.3 Experimental data

In addition to the simulation results from MOOS-IvP, showing validation in a controlled environment, experimental trials were conducted to validate the vehicle’s dynamics during the search and RI tasks, and observe typical sensor capability.



Figure 5.8: Kapitas (Seahorse) vehicle

### 5.3.1 Kapitas Trials

The aim of the Kapitas trial was to observe the behaviour of an AUV in a real sea trial, where the vehicle dynamics and sensor quality are affected by the environment. Two experiments were designed:

1. During the search task, the AUV follows a lawnmower path — long straight lines with constant speed. Data was collected for a vehicle moving in two parallel lines collecting SSS data.
2. During the RI task, the AUV manoeuvres around a target. The second experiment was designed to gain insight of the procedure of collecting data for target identification. This is relevant to the  $t_H$  parameter defined in Chapter 3, giving the additional time needed to reallocate and collect high-resolution data for each detection. In chapter 3,  $t_H = 150$  s, and this value was assumed without previous data or literature available.

The trial was run at Rungsted Havn in Denmark, at the Øresund (the Sound), the channel that connects the Baltic and the North Sea. One characteristic of the area is that it has very poor visibility in the water so a camera sensor is not suitable. The vehicle used for the trial is Kapitas (or SeaHorse, a non-commercial vehicle), shown in Figure 5.8. It is larger than the average torpedo-shaped AUVs and its manoeuvrability and speed are low. The trial was done on the 16 and 17 November 2016.

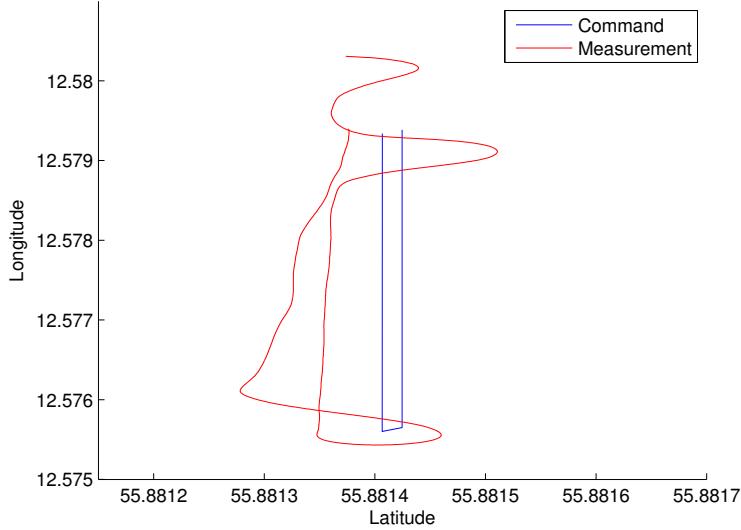


Figure 5.9: Experiment 1, Mission 1: Position of vehicle ( $x$  and  $y$  coordinates)

### 5.3.2 Lawnmower Path Experiment

The navigation data from two separate missions represent the lawnmower experiment and is given in Figure 5.9 and Figure 5.10. The blue line is the predefined route. The red line shows the actual location measurement. There is a large deviation between planned and actual path of the vehicle, due to strong currents on the day of the trial. The experiments were performed in shallow water: the first mission was done at 6 metres depth and the second at 1 metre depth. The depth of the area was 11 metres. The speed of the vehicle was set to 1.4 m/s, and the currents reached up to 1 m/s at times. There was one instance when the vehicle aborted the mission and could not dive due to the strong currents.

#### Side-scan sonar data collection

Side-Scan Sonar (SSS) data was collected while following a lawnmower path. The target that was observed is given in Figure 5.11. The size is about 1.5 m length and 0.5 m diameter. The target's size is representative of a bottom mine. It was submerged to the sea floor. The waypoints of the mission were planned based on the target's coordinates.

The side scan sensor used in the trial is Atlas's Marine Sonic SSPC AUV system<sup>1</sup>. Figure 5.12 gives a snapshot of the target. The higher frequency band of the sonar is used - 900 kHz and the operational range, or swatch width, was 50 m. The target is marked with a teal circle. The cylindrical shape is

---

<sup>1</sup><http://www.marinesonic.com/>

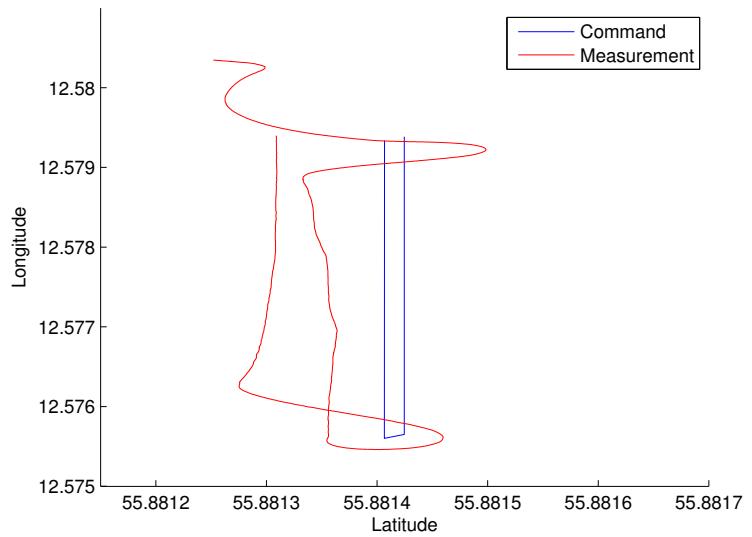


Figure 5.10: Experiment 1, Mission 2: Position of vehicle ( $x$  and  $y$  coordinates)

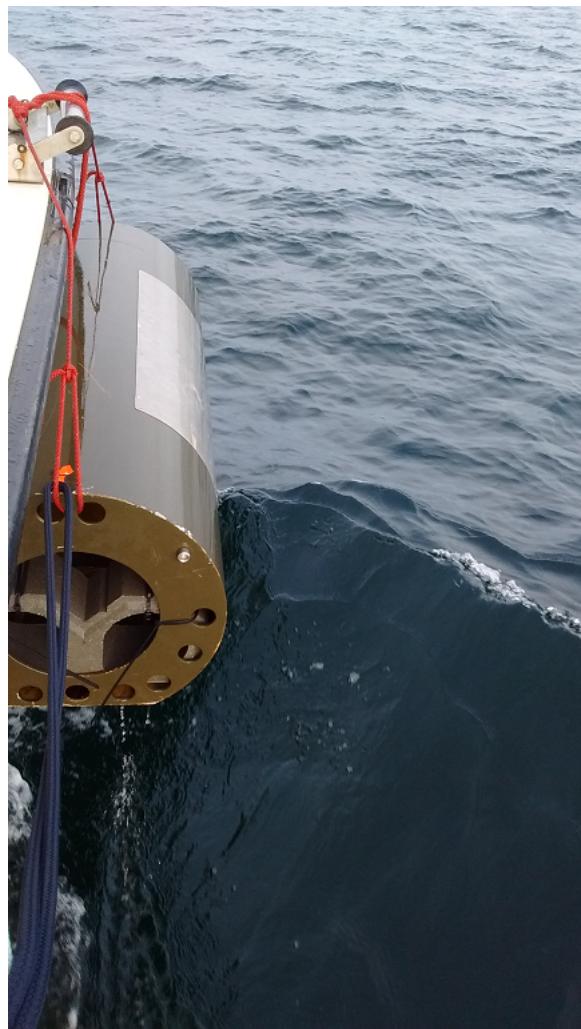


Figure 5.11: Target

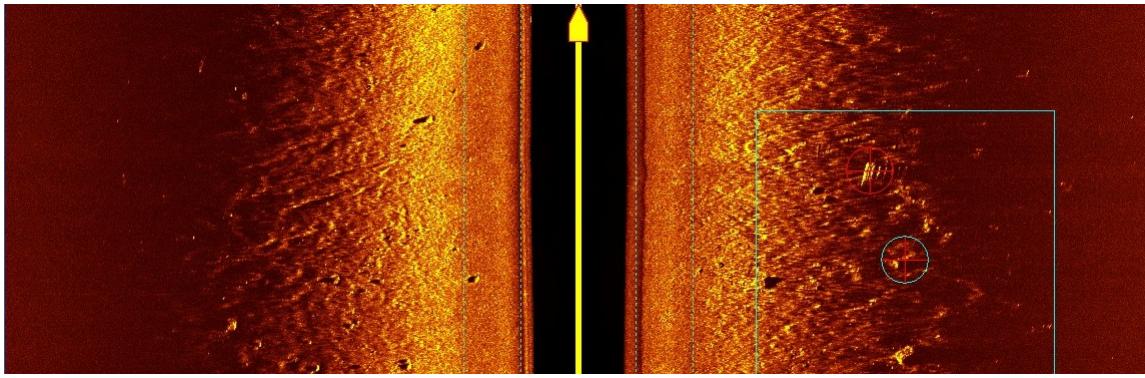


Figure 5.12: Side scan sonar imagery of the target

noticeable, even though the image is not very clear. The artefact in the red circle is the reflection of the floating buoy that was tied to mark the location of the submerged target.

### 5.3.3 Target Identification Manoeuvre Experiment

The second experiment was set up to perform a manoeuvre around a waypoint, however no imagery data was collected or processed. The aim was to simulate collecting different aspect information of the target. The distance to the target and the shape of the manoeuvre path (whether it is a circle or a square around the target, or a lawnmower over the target), depends on the condition of the water, the vehicle capabilities and the sensor. It is not straightforward to determine the best trajectory of the manoeuvre, however data from two missions is available to confirm the uncertainty that the underwater environment brings. Another aspect of the MCM mission shown in this experiment is the disruption between phases - how much time a vehicle needs to return back to search phase if it is disrupted (resurfacing for position fix, SR or RI a target).

The target identification manoeuvre trial involved predefined waypoints and results from two missions are available. Figures 5.13 and 5.14 give the latitude and longitude of the command position ( $x$  and  $y$  coordinates) of the vehicle in blue and the actual movement of the vehicle in red.

One problem that was experienced was that the vehicle merged multiple waypoints and took the average distance between them instead of visiting them all explicitly. The reason for this behaviour was not clear. Another issue, that caused three other mission attempts to abort, was the vehicle's premature resurfacing. The reason for this was also not clear. Figure 5.14 shows partial execution of a mission. The mission data from Figure 5.13 was the most successful experiment and the remaining of the section provides additional data from that mission.

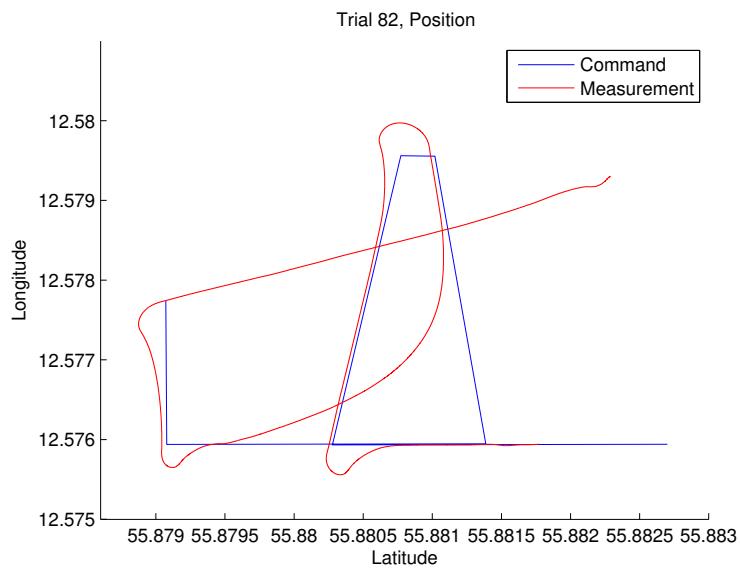


Figure 5.13: Experiment 2, Mission 1: Position of vehicle ( $x$  and  $y$  coordinates)

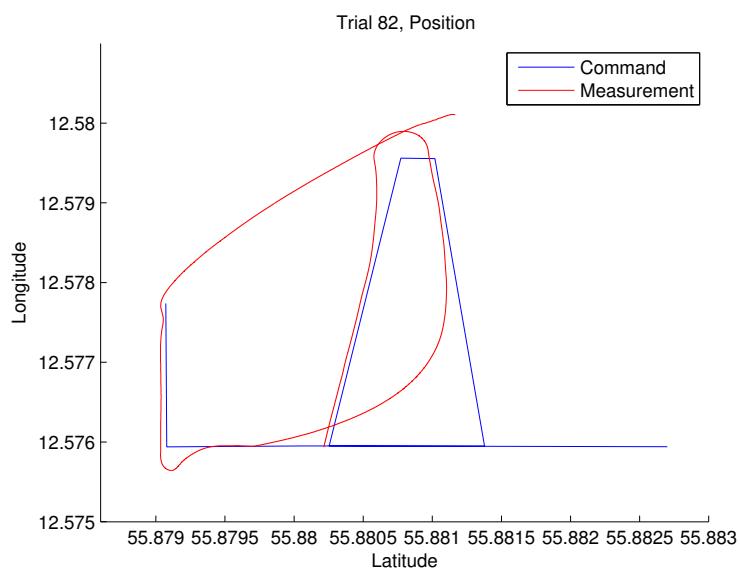


Figure 5.14: Experiment 2, Mission 2: Position of vehicle ( $x$  and  $y$  coordinates)

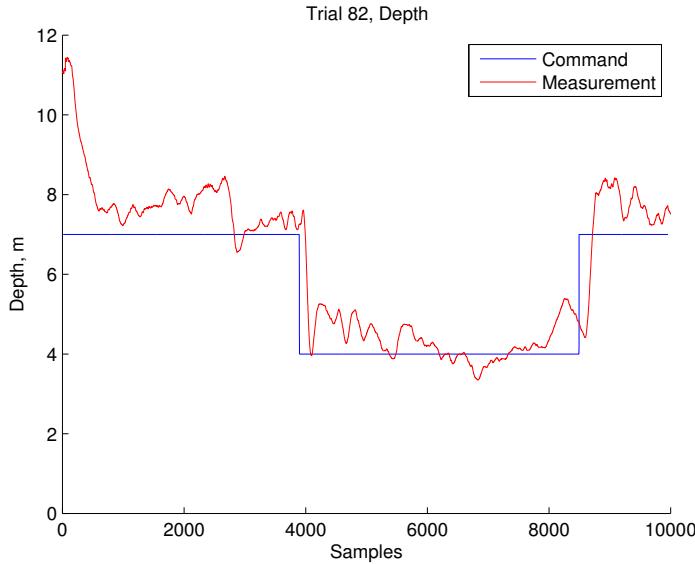


Figure 5.15: Depth of vehicle ( $z$  coordinate)

The depth information is in Figure 5.15. The  $y$  axis is depth in meters, where 0 is the sea bottom, and the  $x$  axis is time (samples). The command signal is in blue and the actual reading is in red. The whole mission took about 20 minutes. The vehicle dives to 7 meters depth. Then it diverts from its course and makes a loop around a target on the sea bottom. This manoeuvre is intended to collect multiple views around the target. The depth is reduced to 4 meters once the vehicle circles around the target. The manoeuvre is not a circle because the sensor is distorted when collecting data at an angle and because the vehicle's navigation is more suited for driving in straight lines.

The vehicle speed was set to a constant of 1.5 m/s throughout the mission. However, there was a current of 0.5 m/s, southward direction. Figure 5.16 gives the command speed in blue and the ground speed measured by the Doppler Velocity Log (DVL) sensor in red. The DVL is an instrument that measures the AUV velocity based on the Doppler shift in the signal's return from the sea bottom.

### **Additional data from target identification manoeuvre experiment**

Some additional sensory information was collected during the target identification manoeuvre experiment. Although it is not directly relevant to the navigation and SSS data, it gives a better idea of the environmental surrounding during the mission. Figure 5.17 gives the Conductivity Temperature Depth (CTD) sensor output. The correlation with the changing depth can be seen on all three parameters. Then Figure 5.18 shows additional measurements of turbidity, pH and redox (organic matter). Finally, Figure 5.19 gives the sound speed measurement. This data would be interesting to analyse if we had

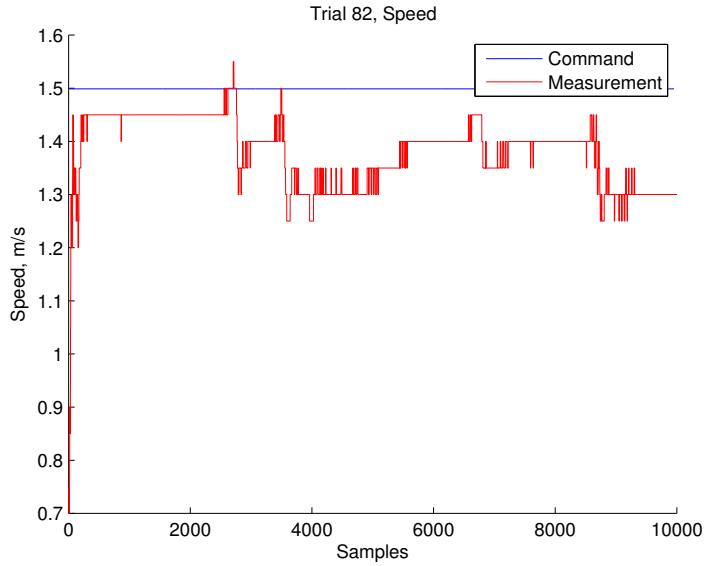


Figure 5.16: Ground speed

some acoustic communication experiments but unfortunately no modem was available during the trial. The sensors from this subsection are commonly used for oceanography applications, such as adaptive sampling.

### 5.3.4 SeaCat Trials

The Kapitas trials give a good example of an AUV behaviour in harsh weather conditions and strong currents. However, the vehicle is not representative, as its size and manoeuvrability are unconventional for a vehicle that would be used for a MCM mission. The SeaCat AUV from Atlas Eketronik GmbH, shown in Figure 1.2, is a better candidate for a MCM application. It is large and powerful enough to be mounted with a SSS, while its manoeuvrability and control are more advanced than the SeaHorse vehicle, resulting in a better navigation precision. The SeaCat vehicle capabilities and parameters were used as a basis for the simulation results in this work.

The SeaCat Simultaneous Localisation And Mapping (SLAM) trials were performed in the Baltic sea, near Eckernförde, Germany. The local currents in the area are influenced partially by dredging and the shallow seabed. The data was collected during two different missions. It includes control parameters of the vehicle, collected every 0.2 seconds.

Figure 5.20 gives an overview of the available data, with the latitude and longitude of two SeaCat missions plotted. The top graph shows a typical lawnmower pattern that is executed twice - once in horizontal and once in vertical direction. Such pattern is useful for applications such as MCM, as it

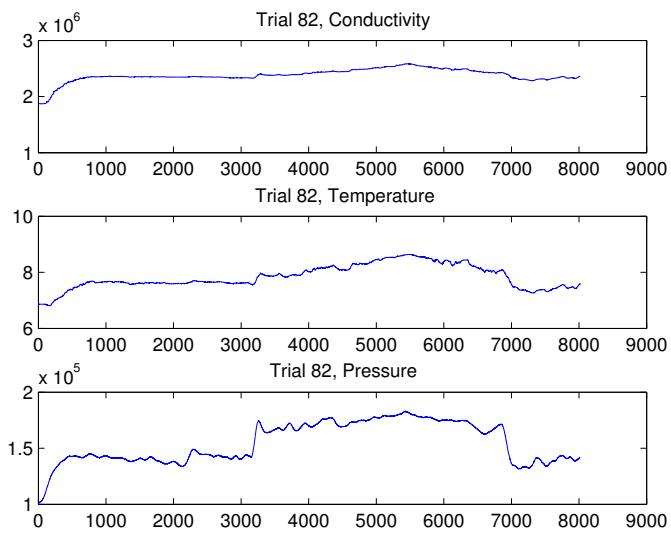


Figure 5.17: Conductivity, Temperature, Depth

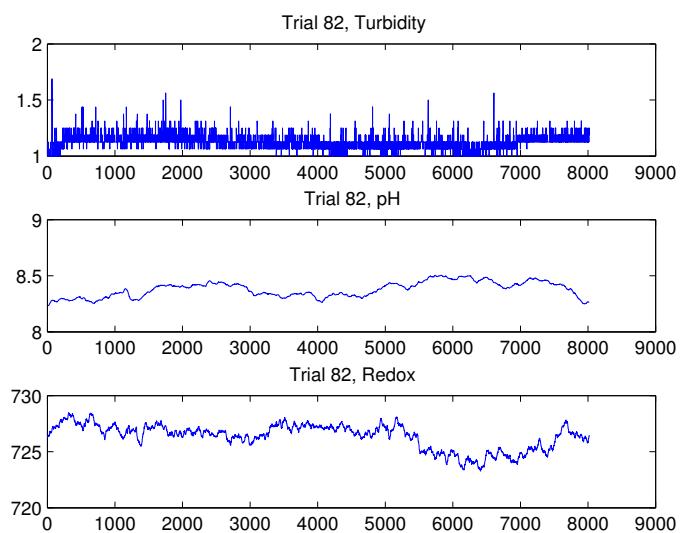


Figure 5.18: Turbidity, pH, Redox

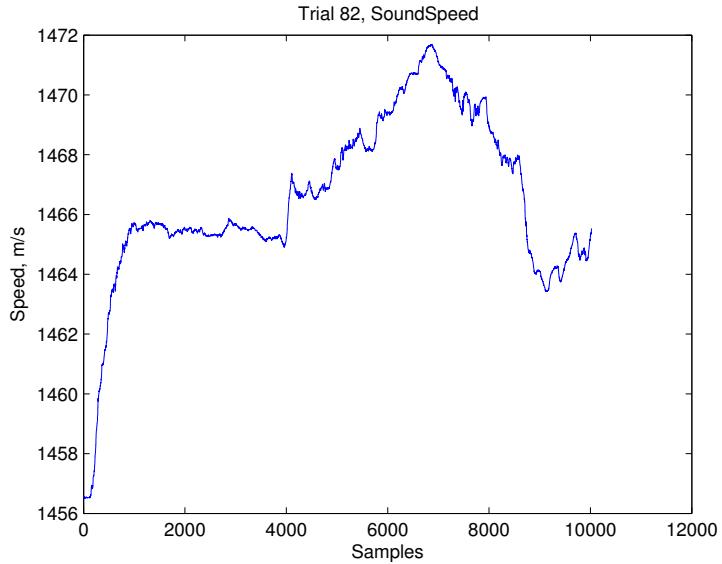


Figure 5.19: Sound speed

provides scans from different angles. The vehicle covers the same tracks twice in this mission. No environmental data is available for these trials, however it can be seen how stable the path of the vehicle is. The bottom graph shows lawnmower pattern again, and there is also a star-shaped track.

Figure 5.21 shows the difference between real (blue) vs command speed (red) that corresponds to the top graph of Figure 5.20. Compared to the Kapitas experiments, there is an improvement in how the measured speed matches the command speed. Some interesting things to note is the amount of time required for the vehicle to calibrate, before it starts following the assigned waypoints. This can be seen on both graphs in Figure 5.20. This is another source for unrealistic assumption and should be factored in that not all the battery capacity is available for mission time. Another detail is the repeated tracks in both missions did not align perfectly — sometimes, especially at the curves, the vehicle was moved substantially from its previous/assigned trajectory. This might be due to environmental changes, however this data is not currently available to match with.

## 5.4 Summary

This chapter provides validation for the Synchronous Rendezvous (SR) method. The simulation experiments in MOOS-IvP test the assumption of reliable communication and networking at close proximity. The results show how unstable and prone to errors the data exchange can be, and highlights the importance of reliable Medium Access Control (MAC) and Multi-Robot Task Allocation (MRTA) mechanism for the successful implementation of the SR method. Despite the instances of mismatched task allo-

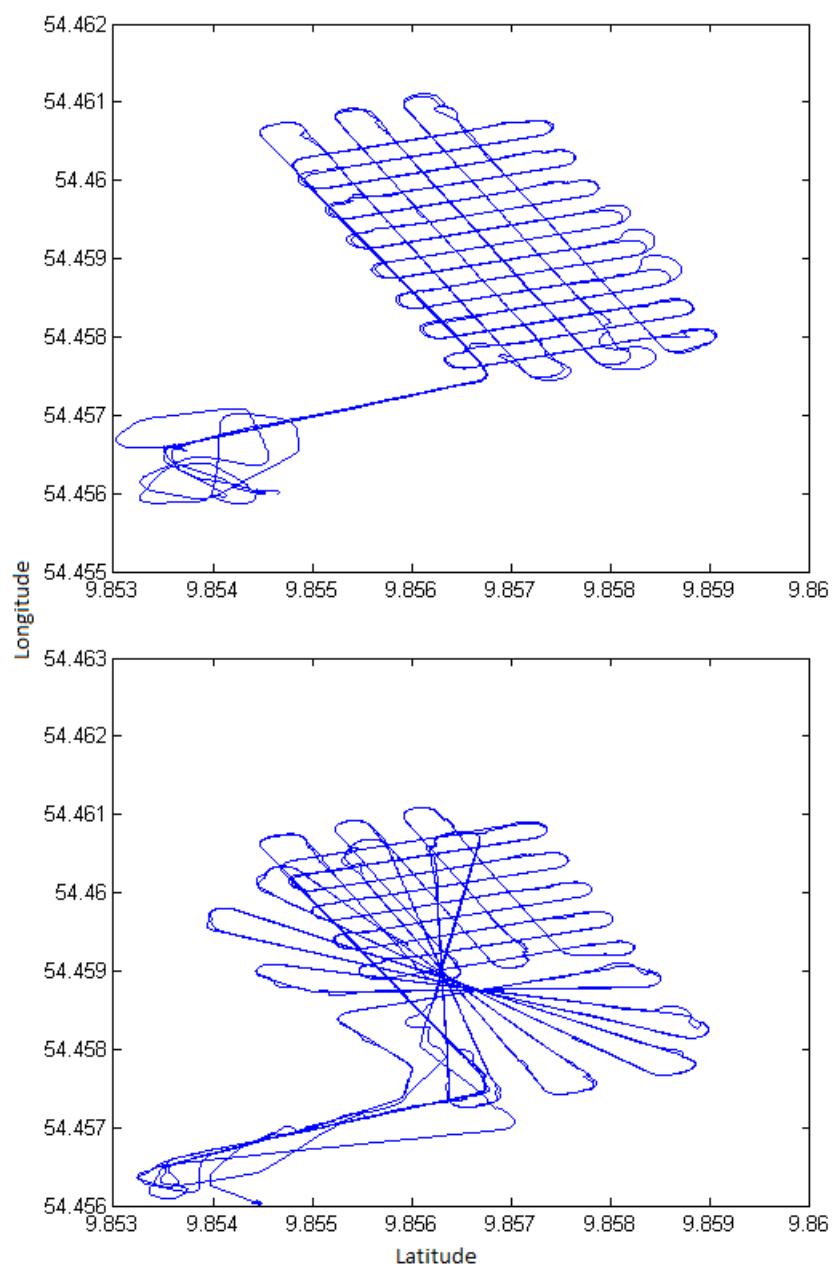


Figure 5.20: Longitude and Latitude data points from SeaCat trials

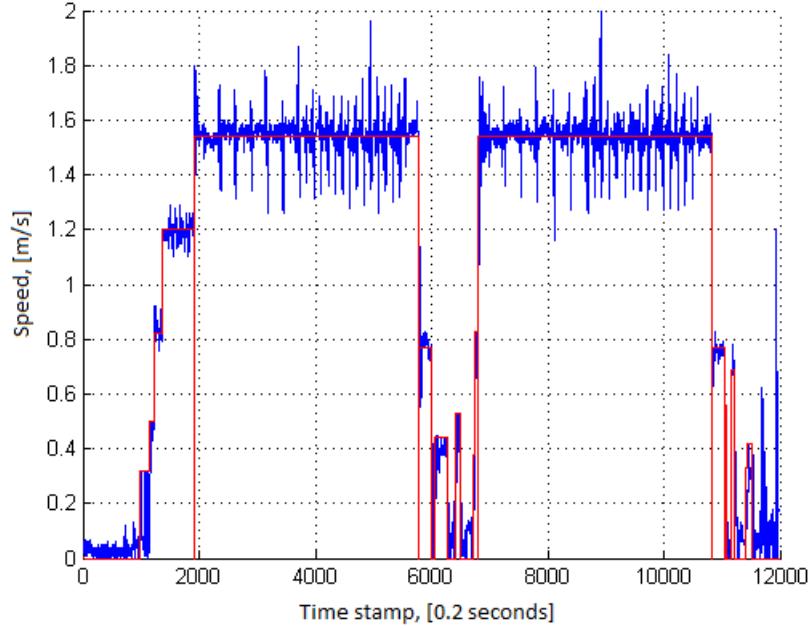


Figure 5.21: Real (blue) vs command speed (red) from SeaCat trials

cation, the experiments showed successful mission executions, scheduling adaptively multiple meeting points and allocating correctly the vehicles paths. The MOOS-IvP simulation can be transferred to a real AUV by changing the environment input application with real sensory data stream coming from the vehicle. The remaining applications from the middleware can be used in the same was as during simulation. Transferring the algorithms developed for the simulation to a real vehicle and sea trial is straightforward, since the middleware is modular.

The environmental effects on the performance of an AUV were observed from data collected from two trials. The Kapitas trial was performed at sea with strong currents present. Two elements of the SR method were tested — the lawnmower search pattern and manoeuvring around a target for the RI task. While the limited trial data and the unconventional vehicle characteristics does not allow to define  $t_H$  as intended, the environmental effects and the unpredictable behaviour of the platform were observed. This confirms how difficult it is to operate even a single underwater vehicle, showing the need of a robust procedure, such as the SR method, to allow cooperative behaviour when working with a multi-AUV network.

The platform dynamics of the SeaCat, the vehicle used as a basis for the simulations from the previous chapters, were examined. Trial data is available showing a stable and precise vehicle movement, similar to the path that needs to be followed during the search task in a MCM mission. The tests were performed in a lake, where no harsh weather conditions were experienced. The trial data validates

the assumption of relying on a stable vehicle operation and extending this to synchronised tasks and paths for a robotic network using the SR method.

# **Chapter 6**

## **Conclusion**

Multi-vehicle robotic networks find a broad application in modern scientific, commercial and defence settings. The ability to communicate robustly is a critical component of any cooperative system. The research conducted in this thesis on developing the Synchronous Rendezvous (SR) method provides an adaptive and intelligent mechanism for reliable underwater sensor network operation.

### **6.1 Summary**

Autonomous Underwater Vehicles (AUVs) have improved significantly over the years and reached a maturity level sufficient to implement them in commercial and defence applications. Recent efforts have been focused on cooperative operations using multiple platforms to share the workload. The main issue for multi-vehicle underwater networks is communications. The physics of the environment does not allow stable signal propagation like in terrestrial settings. Acoustic signals are currently the only reliable shallow-water communications for distances larger than few metres.

Mine Countermeasures (MCM) applications can benefit from using AUVs as the platforms have the potential to be cost and time efficient compared to traditional mine sweeping and hunting vessels. In addition, AUVs bring the imaging sensor closer to the target and allow for more accurate and multi-aspect data collection. The tasks that AUVs can perform in a MCM mission are area search and target reacquisition.

Artificial Intelligence (AI) has a long history of developing methods and solutions for the problem of multiple agents executing a mission, where they need to adapt and reason about their choices based

on limited resources and environmental conditions. Current marine robotics applications apply AI and mission planning methods to multi-AUV problems. However, often unrealistic assumptions of the communication and networking limitations of the system are present. It is non-trivial to compare different solutions, as simulations and trial results follow distinct applications, problem settings and platform limitations.

This thesis develops the Synchronous Rendezvous (SR) method - scheduling multiple meeting points over the mission duration where vehicles can robustly exchange data and decide their actions cooperatively. The SR points are scheduled dynamically and are adapted based on updated knowledge, or an estimation of future gain.

The SR method is evaluated in a MCM scenario where multiple AUVs perform search and Reacquire-Identify (RI) tasks in order to collect data in a minefield. The goal of the vehicles is to maximise the search area while revisiting all detected Mine-Like Objects (MLOs). The vehicles adapt their tasks based on the emerging RI workload by exchanging data at regularly scheduled SR points. The results of a numerical simulation define the working parameters of the approach and evaluate its gains and limitations. The advantage of flexible task allocation that the SR method brings, makes it efficient when both small and large number of targets are present in the mine field.

Using the Markov Decision Process (MDP) framework to represent the multi-vehicle MCM problem and SR scheduler allows one to look into non-myopic solutions, in contrast to the adaptive heuristic-based approach. The advantage is that the vehicles can make a choice not only based on past information, but also factor in their future estimations and expectations of how the goal will change, given their current state. The SR-MDP method is solved using Dynamic Programming (DP) - an optimal recursive algorithm. The common issue of a DP approach is the unmanageable large state and action spaces, resulting from the MDP representation. In this work, a discretisation is applied to reduce the computational complexity and make the approach executable in near real-time. The SR-MDP method is a proof of concept for using a non-myopic algorithm for multi-AUV application.

The numerical simulation evaluation has strong assumptions of the platform dynamics and the environmental conditions. These limitations are studied using a robotics framework simulator, MOOS-IvP, where realistic experiments are simulated. It considers how the AUVs are moving and what is the effect on the sensor in terms of coverage and probability of detection. The communication and networking aspect of the system is also represented. Possible complications, that were not considered in the numerical simulations, but will be crucial for actual implementation of the SR method, are highlighted, such as scheduling mismatch, reliable communication and inability to synchronise data

between platforms.

The environmental effects were studied during sea and lake trials. The platform dynamics was observed under the condition of still water and strong currents. Other parameters, such as speed, size of the platform and reliable control, show how different every mission can be from each other, making the evaluation and assumptions in simulation non-trivial.

Overall, this work presents, develops and evaluates the SR method - a mission planning approach for multi-AUV MCM application.

## 6.2 Key Contributions

This thesis has explored aspects of using Autonomous Underwater Vehicles (AUVs) for MCM application, and developed adaptive and intelligent solution methods. The key contributions of the work are:

- Highlighted common assumptions in current underwater multi-vehicle research. Some of them were addressed when building and testing the SR method.
- Defined a scenario and a benchmark based on MCM requirements and AUV operational restrictions. Using multiple AUVs for MCM is still a novel topic and no benchmarks are available. State of the art capabilities and requirements based on published literature were defined.
- Developed the adaptive SR method. It considers both the environmental and operational constraints of a MCM application — limited communication and regular system monitoring. The method's limitations and gains were evaluated. The simulation results show that the flexible SR method, despite the loss due to meeting, outperforms the benchmark case for both low and high number of targets.
- Developed the SR-MDP method. The method provides a proof of concept for using a non-myopic approach for the multi-AUV MCM problem. The novelty is the representation of the MCM scenario, resulting in low computational load. Real-time execution of the SR-MDP will allow optimal resource management for the robotic network.
- Validated the SR method in a simulated mission experiment and using trial data. The results highlighted the disturbances that can hinder the application of the SR method. However, it was

also shown that providing means for reliable data exchange is a necessary condition for enabling cooperative robotic network underwater. The SR method is one such solution.

## 6.3 Future Work

Given the different approaches used to evaluate and improve the SR method, there were necessary approximations and blackbox models to make the system manageable. These are some of the ways this work can be improved by extending it or making the model more realistic:

In Chapter 3, the main effort was to develop a suitable MCM test bed and evaluate what are the strengths and flaw of the SR method. The simulation was abstracted to include only the coordination aspect between agents, without taking into account physical limitations of the platforms or environment. Some of the possible extensions of the work are:

- Testing other benchmark cases. Interesting benchmarks would be all vehicles performing both tasks, and vehicles surfacing, instead of meeting underwater.
- Evaluating other target distribution models, such as Gaussian, and patterns, based on mine layer movement.
- Considering task allocation when more than one vehicle are performing RI. The current simulation divides the overall RI time between the platforms, assuming equal workload is possible.
- Using a better shortest path algorithm, as a greedy one is used in this thesis.
- Increasing the number of vehicles. All simulations in this work are done with three vehicles. The effect of the SR scheduling and vehicles positioning would be interesting to see if more vehicles are present. Some examples for further execution could be to form multiple small groups or see the behaviour applying swarm techniques.
- Developing a better model for the multi-aspect high-resolution target acquisition.

In Chapter 4, the goal was to develop a network of intelligent agents with a focus on the ability to reason about future actions based on current estimates, while keeping the computation complexity at minimum. Showing that using general methods from RL for a specific and complex problem such as multi-AUV MCM problem is possible, is the first step, and a proof of concept, that they are applicable for real time execution. However, novel methods and updates can significantly improve this work:

- Updating the transition probability and reward function after every SR, or introducing a predictive function. Currently, the algorithm provides a static model, that cannot learn from new information, changing resources or platform parameters.
- Using continuous rather than discrete state space. This will help incorporate changing resource and environment. For example, the battery level is dependent on the currents and how the vehicle adapts its speed to them. In this case, the battery resource cannot be assumed to change linearly and be represented in discrete state space.
- Automatic Target Recognition (ATR), segmentation of the sea bottom and sonar signal processing research can help build a better false alarm model, rather than requiring relocation of all targets. This can be factored in the reward function of the vehicles to select a path planning mechanism for the RI vehicle.
- Implementing the scenario online. Currently, only an offline planning is provided with the available algorithm.
- Experimenting with value function approximation techniques in order to make the computation real time and manageable by the on-board processor.

In Chapter 5, the platform and sensor dynamics are considered, as well as some environmental effects on the vehicle's behaviour:

- Implementing SR-MDP in MOOS-IvP. Only a heuristic approach was implemented in this work.
- Using a better Medium Access Control (MAC) scheme. The current simulation uses a Time Division Multiple Access (TDMA) protocol implemented in the goby extension tool, however it did not provide good control over the experiment. Better evaluation will help understand some of the message collisions and mismatch of data between platforms.
- Implementing adaptive Multi-Robot Task Allocation (MRTA).
- Implementing fault detection and recovery mechanisms as this can hinder synchronisation in a real world experiment.
- Simulating a surface node to give a navigation fix and collect data from AUVs. The ability to incorporate a surface node to the underwater network is one of the advantages of using the SR approach.
- Taking into account probability of detection and classification. This knowledge can be the basis

of introducing another action, such as repeating the search from a different angle, as sometimes this can reduce false alarms significantly.

- Using a different search area pattern rather than a lawnmower. An example is the Zamboni path, used to clean ice hockey fields.

# References

- [1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.
- [2] M. Dumbabin and L. Marques, “Robots for environmental monitoring: Significant advancements and applications,” *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 24–39, 2012.
- [3] B. P. Gerkey and M. J. Matarić, “A formal analysis and taxonomy of task allocation in multi-robot systems,” *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.
- [4] J. Kalwa, A. Pascoal, P. Ridao, A. Birk, T. Glotzbach, L. Brignone, M. Bibuli, J. Alves, and M. Silva, “EU project MORPH: Current status after 3 years of cooperation under and above water,” *IFAC-PapersOnLine*, vol. 48, no. 2, pp. 119–124, 2015.
- [5] M. Benjamin and H. Schmidt, “Unmanned Marine Vehicle Autonomy, Sensing, and Communications lecture notes,” Massachusetts Institute of Technology, 2015.
- [6] A. B. Phillips, N. Gold, N. Linton, C. A. Harris, E. Richards, R. Templeton, S. Thuné, J. Sitbon, M. Muller, I. Vincent *et al.*, “Agile design of low-cost autonomous underwater vehicles,” in *OCEANS Aberdeen*. MTS/IEEE, 2017.
- [7] S. Cowen, S. Briest, and J. Dombrowski, “Underwater docking of autonomous undersea vehicles using optical terminal guidance,” in *OCEANS*, vol. 2. MTS/IEEE, 1997, pp. 1143–1147.
- [8] J. C. Piskura, M. Purcell, R. Stokey, T. Austin, D. Tebo, R. Christensen, and F. Jaffre, “Development of a robust line capture, line recovery (LCLR) technology for autonomous docking of AUVs,” in *OCEANS Monterey*. MTS/IEEE, 2016.

- [9] D. Ribas, N. Palomeras, P. Ridao, M. Carreras, and A. Mallios, “Girona 500 AUV: From survey to intervention,” *IEEE/ASME Transactions on Mechatronics*, vol. 17, no. 1, pp. 46–53, 2012.
- [10] F. Baralli, M. Couillard, J. Ortiz, and D. G. Caldwell, “GPU-based real-time synthetic aperture sonar processing on-board autonomous underwater vehicles,” in *OCEANS Bergen*. MTS/IEEE, 2013.
- [11] G. J. Cornish, “US naval mine warfare strategy: Analysis of the way ahead,” DTIC Document, Tech. Rep., 2003.
- [12] S. C. Truver, “Taking mines seriously,” *Naval War College Review*, vol. 65, no. 2, 2012.
- [13] M. Chitre, S. Shahabudeen, and M. Stojanovic, “Underwater acoustic communications and networking: Recent advances and future challenges,” *Marine technology society journal*, vol. 42, no. 1, pp. 103–116, 2008.
- [14] I. F. Akyildiz, D. Pompili, and T. Melodia, “Challenges for efficient communication in underwater acoustic sensor networks,” *ACM Sigbed Review*, vol. 1, no. 2, pp. 3–8, 2004.
- [15] V. Yordanova, “Data Communication for Underwater Sensor Networks,” Master’s thesis, University College London (UCL), 2013.
- [16] A. Zoksimovski, D. Sexton, M. Stojanovic, and C. Rappaport, “Underwater electromagnetic communications using conduction - channel characterization,” *Ad Hoc Networks*, vol. 34, pp. 42–51, 2015.
- [17] L. J. Johnson, F. Jasman, R. J. Green, and M. S. Leeson, “Recent advances in underwater optical wireless communications,” *Underwater Technology*, vol. 32, no. 3, pp. 167–175, 2014.
- [18] M. Stojanovic, “On the relationship between capacity and distance in an underwater acoustic communication channel,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 4, pp. 34–43, 2007.
- [19] L. E. Kinsler, A. R. Frey, A. B. Coppens, and J. V. Sanders, *Fundamentals of acoustics*, 4th ed. ISBN 0-471-84789-5. Wiley-VCH, 1999.
- [20] M. A. Ainslie, *Principles of sonar performance modelling*. Springer, 2010.
- [21] I. F. Akyildiz, D. Pompili, and T. Melodia, “Underwater acoustic sensor networks: Research challenges,” *Ad hoc networks*, vol. 3, no. 3, pp. 257–279, 2005.

- [22] M. Stojanovic, J. A. Catipovic, and J. G. Proakis, “Phase-coherent digital communications for underwater acoustic channels,” *IEEE Journal of Oceanic Engineering*, vol. 19, no. 1, pp. 100–111, 1994.
- [23] B. Li, J. Huang, S. Zhou, K. Ball, M. Stojanovic, L. Freitag, and P. Willett, “MIMO-OFDM for high-rate underwater acoustic communications,” *IEEE Journal of Oceanic Engineering*, vol. 34, no. 4, pp. 634–644, 2009.
- [24] G. Ferri, A. Munafò, A. Tesei, P. Braca, F. Meyer, K. Pelekanakis, R. Petroccia, J. Alves, C. Strode, and K. LePage, “Cooperative robotic networks for underwater surveillance: An overview,” *IET Radar, Sonar & Navigation*, 2017.
- [25] L. Paull, S. Saeedi, M. Seto, and H. Li, “AUV navigation and localization: A review,” *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 131–149, 2014.
- [26] J. J. Leonard and A. Bahr, “Autonomous underwater vehicle navigation,” in *Springer Handbook of Ocean Engineering*, 2016, pp. 341–358.
- [27] B. Jalving, K. Gade, O. K. Hagen, and K. Vestgard, “A toolbox of aiding techniques for the HUGIN AUV integrated inertial navigation system,” in *OCEANS San Diego*, vol. 2. MTS/IEEE, 2003, pp. 1146–1153.
- [28] S. Carreno, P. Wilson, P. Ridao, and Y. Petillot, “A survey on terrain based navigation for AUVs,” in *OCEANS Seattle*. MTS/IEEE, 2010.
- [29] G. Salavasidis, C. Harris, S. McPhail, A. B. Phillips, and E. Rogers, “Terrain aided navigation for long range AUV operations at arctic latitudes,” in *IEEE/OES Autonomous Underwater Vehicles (AUV)*, 2016, pp. 115–123.
- [30] M. P. Hayes and P. T. Gough, “Synthetic aperture sonar: A review of current status,” *IEEE Journal of Oceanic Engineering*, vol. 34, no. 3, pp. 207–224, 2009.
- [31] M. Brissette and J. E. Clarke, “Side scan versus multibeam echo sounder object detection: Comparative analysis,” *International Hydrographic Review*, vol. 76, no. 2, pp. 21–34, 1999.
- [32] T. Brown, J. Damiano, S. Jhala, R. Moore, B. Morgan, V. Nguyen, T. Opheim, T. Ringwald, W. Roman, and J. Turk, “Next generation mine countermeasures for the very shallow water zone in support of amphibious operations,” DTIC Document, Tech. Rep., 2012.

- [33] N. Kreger, “Putting sea mammals to work: Dolphins help coalition forces in Iraq,” *Journal of Conventional Weapons Destruction*, vol. 7, no. 2, p. 37, 2003.
- [34] M. P. Brito, N. Bose, R. Lewis, P. Alexander, G. Griffiths, and J. Ferguson, “The role of adaptive mission planning and control in persistent autonomous underwater vehicles presence,” in *IEEE/OES Autonomous Underwater Vehicles (AUV)*, 2012.
- [35] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009.
- [36] T. Lozano-Perez, “Techniques in Artificial Intelligence lecture notes,” Massachusetts Institute of Technology, 2002.
- [37] M. Cashmore, M. Fox, T. Larkworthy, D. Long, and D. Magazzeni, “AUV mission control via temporal planning,” in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6535–6541.
- [38] F. Maurelli, M. Carreras, J. Salvi, D. Lane, K. Kyriakopoulos, G. Karras, M. Fox, D. Long, P. Kormushev, and D. Caldwell, “The PANDORA project: A success story in AUV autonomy,” in *OCEANS Shanghai*. MTS/IEEE, 2016.
- [39] M. Molineaux, M. Klenk, and D. W. Aha, “Goal-driven autonomy in a navy strategy simulation.” in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2010.
- [40] M. A. Wilson, J. McMahon, and D. W. Aha, “Bounded expectations for discrepancy detection in goal-driven autonomy,” in *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.
- [41] P. Ridao, J. Batlle, J. Amat, and G. Roberts, “Recent trends in control architectures for autonomous underwater vehicles,” *International Journal of Systems Science*, vol. 30, no. 9, pp. 1033–1056, 1999.
- [42] C. Harris and R. Dearden, “Contingency planning for long-duration AUV missions,” in *Autonomous Underwater Vehicles (AUV)*. IEEE/OES, 2012.
- [43] C. R. German, D. R. Yoerger, M. Jakuba, T. M. Shank, C. H. Langmuir, and K. Nakamura, “Hydrothermal exploration with the autonomous benthic explorer,” *Deep Sea Research Part I: Oceanographic Research Papers*, vol. 55, no. 2, pp. 203–219, 2008.

- [44] N. E. Leonard, D. A. Paley, R. E. Davis, D. M. Fratantoni, F. Lekien, and F. Zhang, “Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in Monterey Bay,” *Journal of Field Robotics*, vol. 27, no. 6, pp. 718–740, 2010.
- [45] A. Alvarez, A. Caffaz, A. Caiti, G. Casalino, L. Gualdesi, A. Turetta, and R. Viviani, “Folaga: A low-cost autonomous underwater vehicle combining glider and AUV capabilities,” *Ocean Engineering*, vol. 36, no. 1, pp. 24–38, 2009.
- [46] A. Munafò, E. Simetti, A. Turetta, A. Caiti, and G. Casalino, “Autonomous underwater vehicle teams for adaptive ocean sampling: A data-driven approach,” *Ocean Dynamics*, vol. 61, no. 11, pp. 1981–1994, 2011.
- [47] T. O. Fossum, “Intelligent autonomous underwater vehicles,” Norwegian University of Science and Technology, Tech. Rep., 2016.
- [48] P. W. van de Ven, C. Flanagan, and D. Toal, “Neural network control of underwater vehicles,” *Engineering Applications of Artificial Intelligence*, vol. 18, no. 5, pp. 533–547, 2005.
- [49] M. Valdenegro-Toro, “Objectness scoring and detection proposals in forward-looking sonar images with convolutional neural networks,” in *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*. Springer, 2016, pp. 209–219.
- [50] B. Wehbe, M. Hildebrandt, and F. Kirchner, “Experimental evaluation of various machine learning regression methods for model identification of autonomous underwater vehicles,” in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4885–4890.
- [51] J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.
- [52] M. Carreras, J. Yuh, J. Batlle, and P. Ridao, “A behavior-based scheme using reinforcement learning for autonomous underwater vehicles,” *IEEE Journal of Oceanic Engineering*, vol. 30, no. 2, pp. 416–427, 2005.
- [53] N. Palomeras, A. El-Fakdi, M. Carreras, and P. Ridao, “COLA2: A control architecture for AUVs,” *IEEE Journal of Oceanic Engineering*, vol. 37, no. 4, pp. 695–716, 2012.
- [54] L. Liu and G. S. Sukhatme, “Making decisions with spatially and temporally uncertain data,” *arXiv preprint arXiv:1605.01018*, 2016.

- [55] K.-C. Ma, L. Liu, and G. S. Sukhatme, “An information-driven and disturbance-aware planning method for long-term ocean monitoring,” in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, 2016, pp. 2102–2108.
- [56] J. Heidemann, M. Stojanovic, and M. Zorzi, “Underwater sensor networks: Applications, advances and challenges,” *Philosophical Transactions of the Royal Society A*, vol. 370, no. 1958, pp. 158–175, 2012.
- [57] A. J. Sørensen and M. Ludvigsen, “Towards integrated autonomous underwater operations,” *IFAC-PapersOnLine*, vol. 48, no. 2, pp. 107–118, 2015.
- [58] L. Brignone, J. Alves, and J. Opderbecke, “GREX sea trials: First experiences in multiple underwater vehicle coordination based on acoustic communication,” in *OCEANS Bremen*. MTS/IEEE, 2009.
- [59] R. Cui, S. S. Ge, B. V. E. How, and Y. S. Choo, “Leader-follower formation control of underactuated autonomous underwater vehicles,” *Ocean Engineering*, vol. 37, no. 17, pp. 1491–1502, 2010.
- [60] S. Kemna, J. G. Rogers, C. Nieto-Granda, S. Young, and G. S. Sukhatme, “Multi-robot coordination through dynamic Voronoi partitioning for informative adaptive sampling in communication-constrained environments,” in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2124–2130.
- [61] S. Climent, A. Sanchez, J. Capella, and J. Serrano, “Study of MAC protocols for a real underwater sensor network application,” in *International Conference on Wireless Networks, ICWN*, 2012.
- [62] F. Favaro, P. Casari, F. Guerra, and M. Zorzi, “Data upload from a static underwater network to an AUV: Polling or random access?” in *OCEANS Yeosu*. MTS/IEEE, 2012.
- [63] J. Partan, J. Kurose, and B. N. Levine, “A survey of practical issues in underwater networks,” *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 4, pp. 23–33, 2007.
- [64] M. Chitre, I. Topor, R. Bhatnagar, and V. Pallayil, “Variability in link performance of an underwater acoustic network,” in *OCEANS Bergen*. MTS/IEEE, 2013.

- [65] J. Potter, J. Alves, D. Green, G. Zappa, I. Nissen, and K. McCoy, “The JANUS underwater communications standard,” in *Underwater Communications and Networking (UComms)*. IEEE, 2014.
- [66] K. Foo, P. Atkins, T. Collins, S. Pointer, and C. Tiltman, “Sea trials of an underwater, ad hoc, acoustic network with stationary assets,” *IET Radar, Sonar & Navigation*, vol. 4, no. 1, pp. 2–16, 2010.
- [67] N. Mivsković, A. Pascoal, M. Bibuli, M. Caccia, J. A. Neasham, A. Birk, M. Egi, K. Grammer, A. Marroni, A. Vasilijević *et al.*, “CADDY project, year 1: Overview of technological developments and cooperative behaviours,” *IFAC-PapersOnLine*, vol. 48, no. 2, pp. 125–130, 2015.
- [68] A. J. Shafer, M. R. Benjamin, J. J. Leonard, and J. Curcio, “Autonomous cooperation of heterogeneous platforms for sea-based search tasks,” in *OCEANS Quebec*. MTS/IEEE, 2008.
- [69] M. Faria, J. Pinto, F. Py, J. Fortuna, H. Dias, R. Martins, F. Leira, T. A. Johansen, J. Sousa, and K. Rajan, “Coordinating UAVs and AUVs for oceanographic field experiments: Challenges and lessons learned,” in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6606–6611.
- [70] G. Ferri, F. Ferreira, and V. Djapic, “Boosting the talent of new generations of marine engineers through robotics competitions in realistic environments: The SAUC-E and euRathlon experience,” in *OCEANS Genova*. MTS/IEEE, 2015.
- [71] A. O. Hero and D. Cochran, “Sensor management: Past, present, and future,” *IEEE Sensors Journal*, vol. 11, no. 12, pp. 3064–3075, 2011.
- [72] T. H. de Groot, O. A. Krasnov, and A. G. Yarovoy, “Mission-driven resource management for reconfigurable sensing systems,” *IEEE Systems Journal*, 2016.
- [73] A. Charlish, “Autonomous agents for multi-function radar resource management,” Ph.D. dissertation, UCL (University College London), 2011.
- [74] L. E. Parker, “ALLIANCE: An architecture for fault tolerant multirobot cooperation,” *IEEE Transactions on Robotics and Automation*, vol. 14, no. 2, pp. 220–240, 1998.
- [75] B. P. Gerkey and M. J. Mataric, “Sold!: Auction methods for multirobot coordination,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, 2002.

- [76] S. Sariel, T. Balch, and N. Erdogan, “Naval mine countermeasure missions,” *IEEE Robotics & Automation Magazine*, vol. 15, no. 1, pp. 45–52, 2008.
- [77] S. Sariel, “An integrated planning, scheduling and execution framework for multi-robot cooperation and coordination,” Ph.D. dissertation, Istanbul Technical University, Turkey, 2007.
- [78] A. E. Deng, Beaujean, “Task allocation and path planning for collaborative autonomous underwater vehicles operating through an underwater acoustic network,” *Journal of Robotics*, 2013.
- [79] K. DeMarco, M. E. West, and T. R. Collins, “An implementation of ROS on the Yellowfin autonomous underwater vehicle (AUV),” in *OCEANS Santander*. MTS/IEEE, 2011.
- [80] S. Kemna, M. J. Hamilton, D. T. Hughes, and K. D. LePage, “Adaptive autonomous underwater vehicles for littoral surveillance,” *Intelligent Service Robotics*, vol. 4, no. 4, pp. 245–258, 2011.
- [81] R. T. Kessel and R. D. Hollett, “Underwater intruder detection sonar for harbour protection: State of the art review and implications,” NATO Undersea Research Centre La Spezia, Tech. Rep., 2006.
- [82] L. Freitag, M. Grund, C. von Alt, R. Stokey, and T. Austin, “A shallow water acoustic network for mine countermeasures operations with autonomous underwater vehicles,” *Underwater Defense Technology (UDT)*, 2005.
- [83] G. Ferri, A. Munafò, R. Goldhahn, and K. LePage, “Results from COLLAB13 sea trial on tracking underwater targets with AUVs in bistatic sonar scenarios,” in *Oceans St. John’s*. MTS/IEEE, 2014.
- [84] S. Dugelay, D. Williams, G. Okopal, W. Connors, O. Midtgård, T. Sabo, M. Geilhufe, O. Lorentzen, M. Ditzel, S. Giordini, and G. Beckers, “MANEX’14: Experimental description and preliminary results,” NATO Science and Technology Organization., Tech. Rep., 2015.
- [85] P. Patron, Y. Pailhas, J. Cartwright, F. Maurelli, Y. Petillot, J. Sawas, and N. Valeyrie, “Fully integrated multi-vehicles mine countermeasure missions,” in *Proceedings of Underwater Acoustics Conference (UAM)*, 2011.
- [86] C. C. Sotzing and D. M. Lane, “Improving the coordination efficiency of limited-communication multi-autonomous underwater vehicle operations using a multiagent architecture,” *Journal of Field Robotics*, vol. 27, no. 4, pp. 412–429, 2010.

- [87] N. Roy and G. Dudek, “Collaborative robot exploration and rendezvous: Algorithms, performance bounds and observations,” *Autonomous Robots*, vol. 11, no. 2, pp. 117–136, 2001.
- [88] J. Lin, A. Morse, and B. Anderson, “The multi-agent rendezvous problem. part 1: The synchronous case,” *SIAM Journal on Control and Optimization*, vol. 46, no. 6, pp. 2096–2119, 2007.
- [89] Z. Zeng, L. Lian, K. Sammut, F. He, Y. Tang, and A. Lammas, “A survey on path planning for persistent autonomy of autonomous underwater vehicles,” *Ocean Engineering*, vol. 110, pp. 303–313, 2015.
- [90] A. Tsourdos, B. White, and M. Shanmugavel, *Cooperative path planning of unmanned aerial vehicles*. John Wiley & Sons, 2010, vol. 32.
- [91] T. W. McLain and R. W. Beard, “Trajectory planning for coordinated rendezvous of unmanned air vehicles,” in *AIAA Guidance, Navigation, and Control Conference*, vol. 4369. AIAA Reston, VA, 2000.
- [92] D. Köhntopp, B. Lehmann, D. Kraus, and A. Birk, “Segmentation and classification using active contours based superellipse fitting on side scan sonar images for marine demining,” in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3380–3387.
- [93] O. Daniell, Y. Petillot, S. Reed, J. Vazquez, and A. Frau, “Reducing false alarms in automated target recognition using local sea-floor characteristics,” in *Sensor Signal Processing for Defence (SSPD), 2014*. IEEE, 2014, pp. 1–5.
- [94] V. Yordanova and H. Griffiths, “Rendezvous point technique for multivehicle mine countermeasure operations in communication-constrained environments,” *Marine Technology Society Journal*, vol. 50, no. 2, pp. 5–16, 2016.
- [95] ——, “Synchronous rendezvous technique for multi-vehicle mine countermeasure operations,” in *OCEANS Washington*. MTS/IEEE, 2015.
- [96] V. Yordanova, H. Griffiths, and S. Hailes, “Rendezvous planning for multiple autonomous underwater vehicles using a markov decision process,” *IET Radar, Sonar & Navigation*, vol. 11, no. 12, pp. 1762–1769, 2017.
- [97] J. Araujo, P. Sujit, and J. Sousa, “Multiple uav area decomposition and coverage,” in *Computational Intelligence for Security and Defense Applications (CISDA), 2013 IEEE Symposium on*. IEEE, 2013, pp. 30–37.

- [98] Z. Feng, R. Dearden, N. Meuleau, and R. Washington, “Dynamic programming for structured continuous Markov decision problems,” in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. AUAI Press, 2004, pp. 154–161.
- [99] C. Guestrin, D. Koller, R. Parr, and S. Venkataraman, “Efficient solution algorithms for factored MDPs,” *Journal of Artificial Intelligence Research*, vol. 19, pp. 399–468, 2003.
- [100] R. Bellman, “Dynamic programming and lagrange multipliers,” *Proceedings of the National Academy of Sciences*, vol. 42, no. 10, pp. 767–769, 1956.
- [101] A. Ng, “CS229: Machine learning lecture notes,” Stanford University, 2016.
- [102] D. Silver, “COMPM050/COMP GI13: Reinforcement learning lecture notes,” University College London, 2015.
- [103] P. Abbeel, “CS287: Advanced robotics lecture notes,” University of California at Berkeley, 2011.
- [104] J. Beardwood, J. H. Halton, and J. M. Hammersley, “The shortest path through many points,” in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 55, no. 04. Cambridge Univ Press, 1959, pp. 299–327.
- [105] M. Prats, J. Pérez, J. J. Fernández, and P. J. Sanz, “An open source tool for simulation and supervision of underwater intervention missions,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 2577–2582.
- [106] M. R. Benjamin, H. Schmidt, P. M. Newman, and J. J. Leonard, “Nested autonomy for unmanned marine vehicles with moos-ivp,” *Journal of Field Robotics*, vol. 27, no. 6, pp. 834–875, 2010.
- [107] J. Weaver and D. Dunlap, “Hell bay trials: A multinational collaborative marine autonomy field experimentation program,” in *OCEANS Monterey*. MTS/IEEE, 2016.
- [108] T. Teubler, L. Shuang, and H. Hellbrück, “Integrating expert system clips into dune for auv control,” in *OCEANS Genova*. MTS/IEEE, 2015.
- [109] S. Joyeux, J. Schwendner, T. M. Roehr, and R. I. Center, “Modular software for an autonomous space rover,” in *The 12th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2014)*, 2014.
- [110] J. Weaver, J. Perkins, and D. Sternlicht, “Advanced autonomy architecture for maritime applications AVA,” in *OCEANS Monterey*. MTS/IEEE, 2016.

- [111] E. Einhorn, T. Langner, R. Stricker, C. Martin, and H.-M. Gross, “Mira-middleware for robotic applications,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 2591–2598.
- [112] T. Schneider and H. Schmidt, “Goby-acomms version 2: Extensible marshalling, queuing, and link layer interfacing for acoustic telemetry,” in *9th IFAC Conference on Manoeuvring and Control of Marine Craft, Arenzano, Italy*, 2012.