NTNU
Norwegian University of
Science and Technology

Autonomous Robotic Intervention using ROV

Ida Rist-Christensen

Marine Technology
Submission date: December 2016
Supervisor: Martin Ludvigsen, IMT

Norwegian University of Science and Technology
Department of Marine Technology

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

MASTER'S THESIS IN MARINE CYBERNETICS

Spring 2016

FOR

STUD. TECH. Ida Rist-Christensen

Title: Autonomous robotic intervention using ROVs

Work Description

Remotely Operated Vehicles (ROVs) are commonly used for subsea inspection and light intervention tasks as they have high manoeuvrability and payload capacity.

By increasing the level of autonomy, we can:

- Reduce human interaction
- Increase precision, repeatability and quality

To achieve increased autonomy for ROVs, concepts from AUVs can be adapted by implementing a guidance and mission control level for the vehicle. This project will consider developing the mission and guidance and control level for the ROV 30k and to connect these to the existing control system for the ROV. The control system must be able to handle a sequence of objectives constituting a mission. The system should also contain a level of diagnostics to allow the on-board vehicle system to detect fault and abort the operation when unwanted incidents occur.

Scope of work

1. Literature study of mission control systems for autonomous vehicles, but generic and underwater
2. Proposal of a system for autonomous operation of the ROV 30k
3. Verification of the proposal in simulation
4. Preliminary field tests of the proposed system on ROV 30k

The report shall be written in English and edited as a research report including literature survey, description of mathematical models, description of control algorithms, simulation results, model test results, discussion and a conclusion including a proposal for further work. Source code should be provided on a CD, memory stick or similar. It is supposed that the Department of Marine Technology, NTNU, can use the results freely in its research work, unless otherwise agreed upon, by referring to the student's work. The thesis should be submitted in two copies within December 22nd.

Co-supervisor: Stein M. Nornes Supervisor: Martin Ludvigsen

Preface

This thesis is based on research carried out during the fall semester of 2016 at the Department of Marine Technology, Norwegian University of Marine Technology (NTNU). It is based on the research and results from the project thesis carried out in the spring of 2016. This is a Master's thesis regarding autonomous ROVs and is a result of both individual work and help from fellow students and professors at NTNU. The main topic of this thesis is to design a semi-autonomous agent architecture to perform autonomous robotic interventions using ROVs.

It is assumed that the reader of this report retains a basic knowledge within engineering science.

Trondheim, December 22, 2016

Ida Rist-Christensen

Ida Rist-Christensen

Abstract

This thesis proposes a semi-autonomous agent architecture for a Remotely Operated Vehicle (ROV). The purpose of this architecture is to apply control strategies from Autonomously Operated Vehicles (AUVs) onto ROVs, in order to perform certain subsea operations autonomously. As a result, an autonomous ROV (AROV) can be used to perform tasks that are currently performed by human divers or by an operator through remote control. The architecture increases the safety, reduces the costs, and minimizes the need for human intervention.

The agent architecture focuses on a general subsea operation, the approach and localization of a Structure Of Interest (SOI) located on the seabed. For this purpose, a hybrid control architecture is developed. The term *hybrid* describes the combination of deliberative and reactive control layers. **The deliberative layer** accounts for the slowly moving behavior components, taking the ROV from the surface to the SOI. It is comprised of six underlying system states, dividing the mission into sub-problems. The states follow the mission anatomy in the order: *(1) Launch, (2) Descent, (3) Transit, (4) Sonar tracking, (5) Camera tracking* and *(6) Inspection/Intervention.* **The reactive layer** takes care of the contingency handling, i.e. the behavior of the vehicle when it is exposed to unexpected situations. The reactive behavior implemented in this thesis is *Obstacle/Collision Avoidance*, using computer vision techniques for detection of the obstacle. The reactive layer has priority in **the control execution layer**, which composes the third layer of the agent architecture, by deciding which behavior should be carried out.

The capability and limitations of the agent architecture are demonstrated through software simulations and full-scale field experiments on the research vessel R/V Gunnerus using the ROV Sub Fighter 30 k. Some behaviors are only simulated, such as feedback navigation using sonar and camera data, and autonomous maneuvering of the manipulator's arm. This thesis aims to provide the architecture framework, and consequently, the optimization of the sub-behaviors has not been considered a priority of this work. Optimization of each state should be considered for future research on the subject. The result is an architecture steering the vehicle autonomously, according to the implemented deliberative and reactive behaviors. The agent architecture has the possibility of adding and removing an infinite amount of these behaviors, enabling for a variety of subsea operations.

The results were quite satisfactory, both from simulations and field work. Deliberative and reactive behaviors were tested separately and simultaneously, using known SOIs and obstacles in the Trondheim fjord. All results confirmed the expected autonomous behavior of the vehicle, moving from the surface until the localization of the SOI and stationkeeping in front of it, using the sensors and equipment available.

This thesis shows that the agent architecture is a well-designed framework for increasing the autonomy in ROV operations. Results imply that further research should be carried out on the topic, such as optimization of each sub-behavior, implementation of stereo vision position estimates and autonomous interventions/inspections. This thesis is considered a pioneering contribution to autonomous ROV interventions, adding new results to previous research.

Sammendrag

I denne avhandlingen presenteres en semi-autonom kontroll-arkitektur for ROV operasjoner. Hensikten med denne er å bruke kontrollstrategier fra AUVer på ROVer, for å utføre visse undervannsoperasjoner autonomt. Resultatet blir en autonom ROV (AROV) som brukes til å utføre oppgaver som i dag utføres av dykkere, eller av en operatør via fjernstyring. På denne måten økes sikkerheten, reduseres kostnader, og reduseres behovet for menneskelige inngrep i sliike operasjoner.

Arkitekturen fokuserer på en generell undervannsoperasjon: tilnærming og lokalisering av en Structure of Interest (SOI), en gjenstand eller et objekt på havbunnen som kan være av interesse. Med dette som et mål, blir en hybrid kontroll-arkitektur utviklet gjennom denne avhandlingen. Begrepet *hybrid* beskriver kombinasjonen av deliberative ("bevisste") og reaktive kontroll-lag. **Det bevisste laget** representerer de langsomt bevegende atferdskomponentene som tar ROVen fra overflaten til en SOI. Laget består av seks underliggende systemtilstander, med andre ord sub-problemer, som tilsammen utgjør hele oppdraget. Tilstandene følger oppdragets anatomi i rekkefølgen: *(1) Launch, (2) Descent, (3) Transit, (4) Sonar Tracking, (5) Camera Tracking* og *(6) Inspeksjon/intervensjon*. **Det reaktive laget** tar seg av beredskapshåndtering, dvs. oppførselen til ROVen når den blir utsatt for uventede situasjoner. Den reaktive oppførselen implementert i denne avhandlingen er *Collision Avoidance (kollisjonshåndtering)*, og bruker Computer Vision (teknikker for tolking av digitale bilder ved hjelp av data) for påvisning av hindringen. Det reaktive laget har prioritet i **kontroll-laget**, som komponerer det tredje laget av kontrollarkitekturen, ved å bestemme hvilken adferd som skal utføres.

Evner og begrensninger i arkitekturen vises gjennom programvare-simuleringer og fullskala feltforsøk på forskningsskipet R/V Gunnerus, ved hjelp av ROV Sub Fighter 30 k. Enkelte atferder er bare simulert, for eksempel tilbakekoblingsnavigasjon ved hjelp av sonar- og kameradata (Sonar og Camera Tracking), og autonom manøvrering av manipulatorarmen. Denne avhandlingen fokuserer på å lage et rammeverk av en kontroll-arkitektur, og derfor har optimalisering av hver enkelt systemtilstand/atferd ikke vært prioritert. Optimalisering av hver tilstand bør vurderes for fremtidig forskning på emnet. Resultatet av denne avhandlingen er en kontroll-arkitektur som gjør ROVen selvstyrt, i henhold til de implementerte bevisste og reaktive oppførselene. Arkitekturen har muligheten til å legge til og fjerne uendelig mange av de underliggende atferdene, slik at den kan brukes i en rekke forskjellige undervannsop-

erasjoner.

Resultatene var nokså tilfredsstillende, både fra simuleringer og feltarbeid. Bevisst og reaktiv atferd ble testet separat og samtidig, ved hjelp av kjente SOIer og hindringer i Trondheimsfjorden. Alle resultater bekreftet den forventede autonome oppførselen til ROVen; den beveger seg fra overflaten til lokalisering av SOI og står deretter i ro foran den, ved hjelp av tilgjengelige sensorer og utstyr.

Denne avhandlingen viser at den utarbeidede kontroll-arkitekturen er et godt designet rammeverk for å øke autonomi i ROV-operasjoner. Resultatene antyder at videre forskning bør utføres på emnet, for eksempel på hvordan optimalisere hver sub-tilstand, implementere stereovisjon for posisjons-estimering, og autonome intervensjoner / inspeksjoner. Denne oppgaven er ansett som et banebrytende bidrag til autonome ROV-operasjoner, da den behersker å anvende tidligere forskning for å oppnå nye resultater.

Acknowledgement

During the work with this thesis I have received help and assistance from many people.

First, I would like to thank my inspiring supervisor, Martin Ludvigsen. During writing this thesis, he provided me with help and feedback on my progress, as well as ideas for improvement and additions. He created motivation through technical challenges and teamwork and made the entire work possible by taking the initiative.
Ph.D. candidates Trygve O. Fossum, Mauro Candeloro, and co-supervisor Stein M. Nornes also deserves special thanks. They have always been available to answer questions and provided me with creative thinking and confidence. Not to mention the incredible effort in proofreading these last days. This work would not have been possible without their help.
I would like to thank fellow students for their help and contributions throughout the semester, a special thanks to Msc student Michele Gazzea and Ph.D. candidate Marco Leonardi for good cooperation and motivation during the teamwork.
The whole crew at R/V Gunnerus and AUR-lab have my sincere gratitude for all their help provided during the experimental testing. They were always positive even when problems occurred, and without them, the testing would not have been possible. Professor Ageir Sørensen also deserves a thank for always being enthusiastic, and for sharing motivation and knowledge to his students.

Finally, I would like to thank my family and friends for support and help during my time here at NTNU.

I.R.C

Nomenclature

Vehicles and Payload

ROV Remotely Operated Vehicle

AUV Autonomous Underwater Vehicle

UUV Unmanned Underwater Vehicle

AROV Autonomous Remotely Operated Vehicle

DVL Doppler Velocity Log

BL Bottom-Lock

WL Water-Lock

SONAR Sound Navigation and Ranging

HiPAP High Precision Acoustic Positioning

IMU Inertial Measurement Unit

LBL Long Baseline

USBL Ultra Short Baseline

Software and Interfaces

UDP User Datagram Protocol

TCP Transmission Control Protocol

cRIO Compact Reconfigurable Inputs and Outputs

HIL Hardware-In-the-Loop

GUI Graphical User Interface

HW Hardware

SW Software

FPGA Field-Programmable Gate Array

NI National Instruments

Algorithms and Mathematics

OA Obstacle Avoidance

CT Camera Tracking

SV Stereo Vision

ST Sonar Tracking

(I)LOS (Integral) Line Of Sight

SOI Structure Of Interest

NED North-East-Down

DOF Degree Of Freedom

UTM Universal Transverse Mercator

3D Three-Dimensional

NE North-East

Titles and Organizations

NTNU Norwegian University of Science and Technology

AUR-lab Applied Underwater Robotics Laboratory

TBS Trondheim Biological Station

MsC Master of Science

PhD Doctor of Philosophy

Others

DP Dynamic Positioning

IMR Inspection, Maintenance and Repair

HAA Hybrid Agent Architecture

Greek Letters

η Position vector in NED frame

ψ Heading, i.e. orientation between NED and BODY frame

Contents

List of Figures

Page

List of Tables

Chapter 1

Introduction

Remotely Operated Vehicles (ROVs) are used for several subsea tasks, such as detailed mapping of the seabed, inspection, intervention, maintenance and repair of subsea installations, research and rescue operations. Remote control of such vehicles requires a certain amount of human interaction, which may lead to poor results under the influence of stress, and during time consuming or difficult operations. Worst case, these results can cause low quality of the operation and damaging of the equipment or staff involved. An autonomous approach on the ROVs could reduce the need for human intervention, and thus increase the accuracy, repeatability, quality, and safety around such operations. As well stated by [39]:

"*A general argument for using a robot is if the task is 'dull, dirty, or dangerous*"

which are the commonly used three D's when referring to tasks that are better suited for robotic vehicles than human operators.

In order to increase the level of autonomy of the ROVs, concepts from Autonomous Underwater Vehicles (AUVs) could be considered and implemented through a guidance and mission management level. The aim of this thesis is to develop such a level for general ROV operations and connect this with the existing control system of ROV Sub-fighter 30k for testing. The purpose is to make the ROV reach the third level of autonomy, i.e. management by exception, according to the classification in [8]. This is carried out using the already existing auto functions, by structuring the mission into sub-problems and combining deliberate and reactive control with computer vision techniques. A mission and an autonomous mission control architecture are presented and tested through simulations and field work. This chapter presents the most important requirements needed to design a mission control system for ROVs (Section 1.1), as well as the limitations (Section 1.4), approach (Section 1.3) and objectives (Section 1.2) of this thesis. Finally, Section 1.5 describes the structure of the thesis, as well as its main contributions.

1.1 Background

This section presents the motivation and background behind the work of this thesis. In the following, basic information about Unmanned Underwater Vehicles (UUVs) is given. Focus is on the sub-group ROVs and its current applications. Section 1.1.6 gives a review on autonomy and its previous approaches on ROVs.

1.1.1 UUVs: Characteristics and Applications

UUVs are defined as vehicles which are able to operate underwater without carrying a human operator [21]. In general, UUVs are divided into two categories; AUVs and ROVs [5]. UUVs have a variety of applications, from [39]: oceanic warfare, underwater surveys, inspection of submerged structures, oceanographic feature tracking, subsea mapping and monitoring, underwater cable laying, underwater searching and detection of sea mines, and other oceanographic and military tasks.

Figure 1.1 presents the spatial and temporal resolution and coverage of different manned and unmanned vehicles, and one of the motivations for this thesis is to expand the coverage of a general ROV.

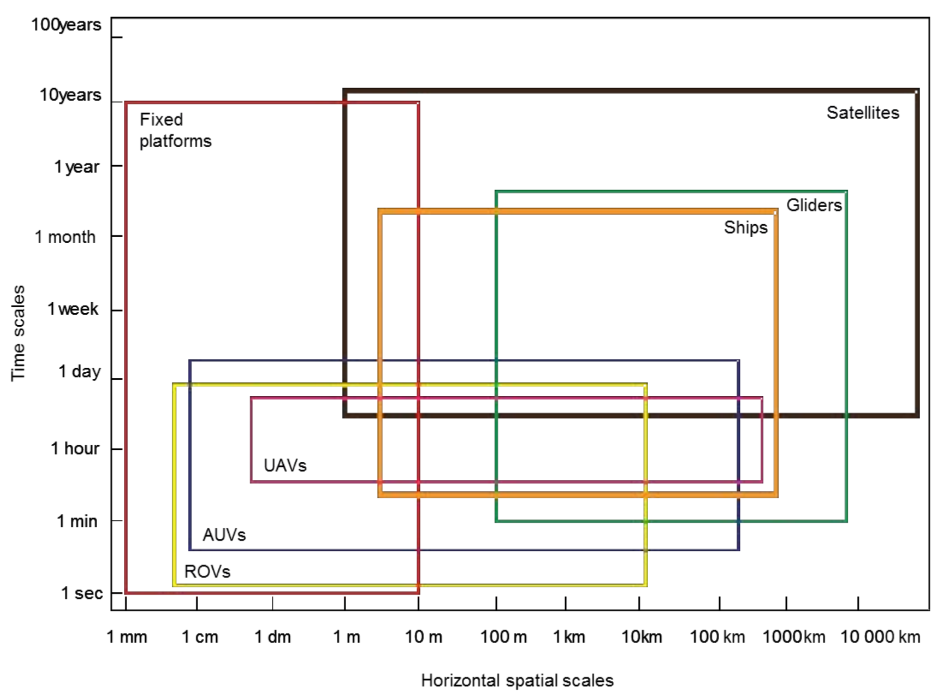


Figure 1.1: *Spatial and temporal resolution and coverage of different offshore and over- and underwater platforms [48]*

1.1.2 AUVs

Autonomous Underwater Vehicles (AUVs) operate without physical connection to the onboard station, with a pre-planned mission. AUVs can perform offshore assessments and surveys, environmental monitoring, hydrography, search and recovery [22]. Due to no physical restriction, AUVs are flexible and can operate over great distances and in complex environment. However, long lasting AUV operations require a high capacity power source onboard the vehicle, and high complexity of the control system in order to ensure a reliable autonomous operation. AUVs are generally designed for minimal resistance in order to optimize the energy consumption. Moreover, they are often are underactuated, i.e. only individually controllable in certain degrees of freedom (DOFs) [21]. Due to the listed limitations, AUVs are unsuited for manipulation and intervention purposes [5].

1.1.3 ROVs

Unlike AUVs, ROVs are physically connected to the surface operator by an umbilical carrying and transferring electrical power, video and data signal [42]. ROVs are robotic divers that allow the vehicle's operator to remain onboard the surface vessel, while the ROV performs operations in any condition and environment below the surface. The disadvantages of an ROV are given by the tether, limiting the spatial range and increasing the drag and resistance, and the need for an operator which remotely controls its operation. However, being remotely operated and having an unlimited power supply, an ROV is in general equipped with several tools, cameras, lights, and sensors, allowing the vehicle to perform a variety of subsea tasks. ROVs are mainly used for surveillance, and can be fully actuated, allowing for DP operations, which are not generally possible with AUVs [17]. Due to the complementary advantages and disadvantages of both AUVs and ROVs, it is easy to understand the desire for a combination of the two vehicles. This would allow for a high precision robotic vehicle performing complex tasks that human operators could not conduct with the same accuracy. The design techniques of AUVs can, therefore, be applied on ROVs in order to create Autonomous ROVs (AROVs). ROVs are classified in terms of size and application [5], Notice that this thesis will only consider the NTNU owned SUB-fighter 30k (Section 4.2).

1.1.4 Applied Underwater Robotics Laboratory

The Applied Underwater Robotics Laboratory (AUR-Lab) was established by NTNU in 2009 with the purpose of developing tools for applications of underwater instrumentation [31]. The lab is used for both education and research at NTNU, and combines simulations, model-scale experiments, full-scale experiments and field work. The AUR-Lab consists of experts from cybernetics, control engineering, marine biology, marine archaeology, electrical engineering and communications. To disposal for the AUR-Lab are several underwater vehicles, as well as NTNU's own research vessel R/V Gunnerus (Section 4.3). As part of the AUR-Lab, the Trondheim Biological Station (TBS) has been launched in order to test subsea technology for oil, gas and marine operations. Technological surveys are often combined with biological surveys in order to optimize the use of AUR-Lab. Among the vehicles provided by NTNU via AUR-Lab, is a SUB-fighter 30k (30k) (Section 4.2), and the ROV Minerva. The former will be used in this thesis.

1.1.5 Autonomy Group

With the purpose of increasing the autonomy of 30k, a dedicated research group was established in 2016. It consisted of the author, MSc student Lars Brusletto, Ph.D. candidates Trygve O. Fossum and Stein M. Nornes and Professor Martin Ludvigsen. Within the end of 2016, the group was extended to include several members, most important for this thesis are MSc student Michele Gazzea, Ph.D. candidate Marco Leonardi and Professor Annette Stahl at the Department of Engineering Cybernetics, NTNU. During the *spring* semester of 2016, the group goal was to develop an autonomy framework using an experimental approach and to present the results through a paper contribution to the Ocean's 16 conference (see [12]). Stud. Tech. Brusletto contributed with computer vision techniques in order to receive and process information from stereo cameras for detection of obstacles. Ph.D. candidates Nornes and Fossum contributed with knowledge, supervision, and experience that made the operation possible. Professor Ludvigsen was the team leader and the supervisor of the project.
During the *fall* semester, the previous work was expanded and improved through more work and additional group members. The work resulted in a contribution to the Ocean's 2017 Conference, see the preliminary abstract to the contribution in Appendix B. Ph.D. candidate Marco Leonardi contributed with improvement of the stereo vision system in cooperation with Professor Stahl. MSc student Michele Gazzea studied the possibility of using stereo vision for more accurate position estimates. This thesis is the result of the author's contributions to the group, combined with help and knowledge from the rest of the group members.

1.1.6 Autonomy

The motivation behind this thesis is to increase the autonomy in ROV operations. Autonomy is a wide subject and can be defined through different point of views, dependent on the context. This thesis focuses on autonomous unmanned vehicles, which are defined by [39] as:

"*Capable of operating without operator input for extended periods of time. Implicit in this description is the requirement that the UUV's sortie accomplishes its assigned goal and makes the appropriate rendezvous for a successful recovery.*"

By introducing this capability in current UUVs, one can perform subsea Inspection, Maintenance and Repair (IMR) operations with high safety, reduced costs and increased efficiency. To obtain a vehicle which is able to manage unexpected events in unknown environment, integration of mathematical models with real-time sensor measurements is necessary [48]. It is common to divide autonomy into levels from manual/remote control to fully autonomous vehicles. According to USA National Research Council [8], autonomy can be divided into four levels:

Level 1 - Manual operation A human operator controls all mission functions.

Level 2 - Management by concent The system automatically recommends actions for selected functions and prompts the operator for specific information or decision making. State of the art autonomous vehicles operate at this level.

Level 3 - Management by exception The system automatically executes mission-related functions, but the operator may override or alter parameters and redirect or abort actions if necessary.

Level 4 - Fully autonomous The system automatically executes mission-related functions, and the operator is only alerted to function progress.

It is common to distiguish the terms *automatic* and *autonomous* when developing an autonomous system. An *automatic* system can perform well-defined functions without human intervention following a pre-defined list of tasks [48, 5]. Such a system follows the principle *sense* → *plan* → *act*. An *autonomous* system can perform complex tasks that are not pre-defined, meaning that the system must be able to learn, adapt and improve when exposed to an unknown environment. Such a system follows the principle *sense* → *act*. Based on the above distinguishment, it can be argued that autonomoy is the ability to follow a series of preprogrammed instructions (automation), but also to be able to replan and reconfigurate by itself. The question is, can a system become fully autonomous? The answer is ambiguous, and an interesting point of view is stated by Hiroshi Ishiguro [14]. In an interview with the Italian newspaper "La Repubblica" [35], Ishiguro claims that not even human are "*autonomous*", but we are following a pre-defined program in our DNA:

"*We are "programmed" to feel emotions and desires from our DNA. The robots are by software. I have some difficulty in understanding even the separation that is made between "real" and "unreal" based on the fact that the feelings of a robot are originated from a code.*"

In the context of this thesis, however, *autonomy* is considered the property of making a system more *intelligent*. Agreeing with the definition of *automatic* and *autonomy* according to [48, 5], a system can be defined as autonomous if it retains both automatic and autonomous properties.
The aim of this thesis is to study the possibility of increasing the level of autonomy in current ROV operations from level 2 to 3. [48] and [38] present research projects already carried out on the subject, as well as proposal and solicitation for further research. [48] also proposes a control architecture framework using a "bottom-up" approach towards autonomy in UUVs. This architecture is used as a motivation for the literature study in this thesis (Chapter 2).

1.2 Objectives

The objective of this thesis can be formulated based on the challenges and research fields described in Section 1.1. In order to reach the desired level of autonomy in ROV operations, the following tasks should be performed.

1. Perform a literature study on mission control systems for autonomous robots.
2. Propose an autonomous agent architecture for conducting a predefined mission and implement it in the control and guidance system of 30k. The agent should be an autonomy framework for general subsea ROV operations.
   (a) The architecture should include both deliberative and reactive control components in order to assure the desired level of autonomy.
   (b) Obstacle avoidance should be studied as part of the reactive layer.
   (c) Mission planning should be established as part of the deliberative layer.

3. Perform a verification of the agent through software simulations.

4. Perform preliminary field tests of the agent using 30k.

The objectives should be conducted using equipment and software available. The work should follow an experimental approach, with the priority of achieving test results and progress in short time. In that manner, this thesis should introduce an innovative contribution to state of the art autonomous ROVs, and open for further thorough research on the subject.

1.3 Approach

To develop autonomous robots, one must widen the engineering approaches to the problem, and allow for other possibly cross-disciplinary methods. The pragmatic V-model is a good candidate when developing a product in a systematic and thoughtful manner, to assure quality of every module. The V-model is illustrated in figure 1.2, from [51]. This thesis is based on the aforementioned V-model, following the top-down design approach in the left branch, and the bottom-up testing and verification steps in the right branch. The left branch represents all theoretic preparations for the mission of this thesis, while the right branch represents the testing and verification after implementing the system, i.e. the experimental approach.

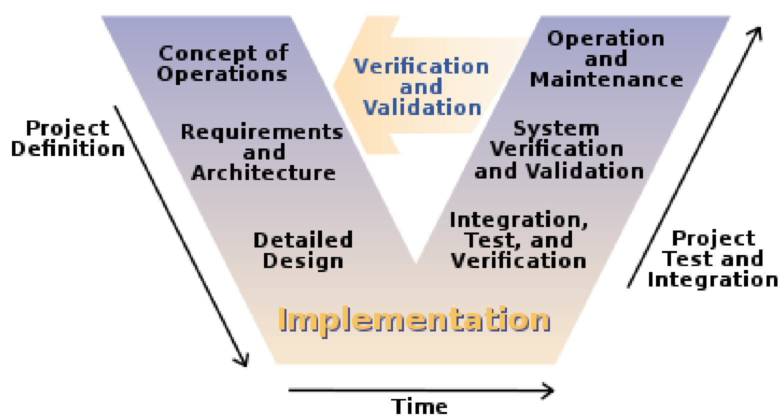


Figure 1.2: *V-model for development of a product [51]*

To create a good foundation for the mission control system, a profound literature study was performed. Further, a general autonomous operation was divided into sub-problems and planned in detail. This study formed the basis of the control architecture chosen for conductance of the mission. A pseudo code and a flow chart of the mission planning were

made for further implementation in the ROV control system, in accordance with the designed architecture and mission. Implementation of the code was performed in the existing ROV control system, and the architecture was slightly modified for interfacing with the control system.
Software simulations using the existing ROV simulator were performed to verify the performance of the control architecture. Deliberative and reactive control algorithms were tested separately and together, both in simulations and in field tests on Gunnerus with 30k. In addition, on-line communication and feedback from an external computer with the task of recognizing the objects from the stereo cameras was performed.

1.4 Limitations

There have been some limitations during the execution of the work described this thesis. First of all, the thesis focuses on the design of an architecture framework for autonomous missions, and not on the development of each specific behavior of the vehicle. Behavior refers, for instance, to the module which positions the ROV using stereo cameras. This is currently being studied by a fellow MSc student, and could thus not yet be implemented in the architecture proposed in this thesis. As a result, these behaviors are simplified and simulated when performing tests on the agent architecture. All behaviors are described in Chapter 3, and the simplifications introduced by this thesis in order to simulate their effects. Obviously, time has also been an important limitation that constrained the work presented in this thesis. Autonomous ROVs form a wide field of study, and a Master's thesis is not sufficient for covering the entire subject. Nevertheless, experiments and full-scale tests have been used to improve the design and optimize the system.
Despite the above limitations, this thesis contains successful results which were obtained through good equipment, staff, operational conditions and diligent work.

1.5 Structure of Thesis

This thesis is organized following the main objectives listed in Section 1.2.

Chapter 2 presents a litterature study of mission control systems and agent architectures. It presents three different and typical architectures and their main objectives. Although this chapter is brief, the literature study has been time consuming due to lack of previous knowledge on the area.

Chapter 3 proposes a hybrid agent architecture for autonomous mission control of the 30k ROV. A general mission is used as a base for the architecture; the approach and localization of a structure of interest located on the sea bottom. The architecture is based on the literature study from Chapter 2. The mission is divided into sub-problems (states) and organized sequentially to follow the mission anatomy. Section 3.2-3.8 describes each state in detail, whereas Section 3.9 describes the planning of a mission according to the defined states.

Chapter 4 presents the equipment and experimental set up for testing the hybrid agent. The ROV control system is described in detail in Section 4.1, and specifications of 30k are described in Section 4.2. The rest of the chapter presents additional equipment and tools used for simulations and full-scale experiments. Chapter 2, 3 and 4 consider the methods behind the work of this thesis.

Chapter 5 presents a verification of the implemented agent architecture through software simulations. The used simulator is designed at AUR-lab and it is called *Verdandi*. The results are presented on a cartesian diagram representing the north and east position of the vehicle in NED frame, as well as, time varying position, heading and depth, and finally 3D illustrations of the simulations.

Chapter 6 presents the results from experimental testing in the Trondheim fjord using 30k on R/V Gunnerus. The tests were performed during three working days on Gunnerus, and the results are presented in the same manner as in Chapter 5. Summerizing, Chapter 5 and 6 contain the main results of this thesis.

A discussion of the results is presented in chapter 7. This chapter analyses and compares the plots presented in chapter 5 and 6, in order to draw a final conclusion.

Finally, a conclusion is presented in chapter 8.1. Both individual and overall conclusions are drawn, based upon the discussion given in chapter 7. This chapter also presents a section called "Further Work", that suggests further development that could improve and extend the work proposed in this thesis.

1.5.1 Appendices

The attached appendices in this thesis are mainly presenting additional work associated with the content of the thesis. In **Appendix A**, the digital attachments are listed and explained briefly. **Appendix C** presents an alternative way of organizing the mission in terms of sub-behaviors, using Huffman table theory [49]. This contribution is considered relevant, but not necessary for the conductance of this thesis, thus is it summarized in an appendix.

Appendix D and **E** contains additional results from simulations and full-scale tests, respectively. The results are considered additional because they do not prove more than what is already proven in the thesis.

1.5.2 Contributions

Focus in this thesis has been to achieve experimental results, thus is there more results presented from the full-scale experiments than the simulations. Although several simulations were tested, the post-processing of the raw data in order to visualize the results was only performed for specific cases. This, in order to reduce the amount of repeating results in this thesis. The simulations were performed to verify and prepare the mission control system for experimental results, but the latter is the main result of this thesis.

Based on the work done in the project thesis, this Master's thesis proposes improvement in the hybrid agent architecture for autonomous ROVs. The project thesis contributed with a mission control architecture which was simplistic in structure, and results were not completely tested due to certain limitations. The architecture made a solid fundament for the work done in this thesis. The overall structure of the architecture remains the same, but focus has been on improving each state to become as close as possible to realistic behaviors, and to expand the architecture to be able to add and remove as many states as desired. Lastly, more full scale experiments has been carried out and strengthened the design through positive results. Chapter 3 presents the main contributions of this thesis.

Chapter 2

Mission Control Architectures for AUVs

This chapter presents a literature study on control architectures for UUVs. The study focuses on the work done by Pere Ridao in [36]. His work proposes a new hybrid architecture for the underwater vehicle GARBI. The hybrid architecture is used as a base for the architecture developed and presented in this thesis. The work is also inspired by the research from the Heriott Watt University presented in [19]. Section 2.1.1, 2.1.2 and 2.1.3 present three different, commonly used control architectures for UUVs, the latter being the hybrid architecture from [36]. The architectures are compared and described in detail.

2.1 Autonomous Architectures

An autonomous vehicle is a vehicle with a sensorial-actuator system managed by an agent control architecture, to perform a specified mission. Hence, a control architecture is the framework containing control laws, error detections, recovering, path planning, task planning and monitoring of the event along the process of the specified mission [12]. The control architecture is one of the most important elements when developing the autonomy of a vehicle [7]. There exist three main approaches for designing a control architecture:

1. Deliberative architecture: Strongly based on planning and highly depended on knowledge about the environmental dynamics. Follows the structure, $sense \rightarrow plan \rightarrow act$

2. Behavior-based/reactive architecture: Strongly based on quick response and suitable for highly dynamic environment difficult to model. Follows the structure, $sense \rightarrow act$

3. Hybrid architecture: Uses the advantages of the two previous types, to assure both

conductance and quick planning of the mission. Usually follows a structure of three layers, *Deliberative layer→ Control execution layer →Functional reactive layer*

The different architectures have similarities and differences, further discussed in the following.

2.1.1 Deliberative Architectures

Deliberative architectures presents predictable behaviors, allowing for planning and modeling of the mission before execution. The architecture includes a world model on which planning decisions can be taken [6]. Planning can be done prior to the mission or adaptively in-situ. An accurate world model is crucial for the mission planning to be executable. The development of such a model is complex and demanding since faults and inaccuracies would cause non-optimal behavior. Results and description from successful research projects on deliberative architectures can be found in [52] and [34]. Deliberative architectures may be classified into two groups according to the organization of the planning of the mission:

- **Hierarchical architectures**: Mission is decomposed into subtasks and planned in detail. The plan is executed as a sequence of actions, and each subtask modifies the control or signal flows, manipulating the behavior of the vehicle. The mission is managed in tasks, leading the vehicle to the goal and avoiding pre-known obstacles. The execution sequence is fixed, and no unpredictable behaviors are considered. Examples of previous hierarchical architectures can be found in [4] and [15].

- **Centralized architectures**: The subtasks are organized as a set of agents communicating through a central control module. Each module has a specified function, and the modules are typically divided into *Monitors* and *Exceptions*. Monitors are conditions for different modules tested under certain circumstances. Exceptions are handling cases where there is no procedure, i.e. when the behavior of the vehicle does not fit into any of the pre-planned modules. Previous research and implemented centralized architectures can be found in [40] and [13].

2.1.2 Reactive Architectures

Reactive/behavioral architectures are designed to accomplish the mission without predictive representation of the surrounding environment, but rather using real-time sensory information [39]. Such architectures become more popular due to the low requirements of environmental knowledge. It integrates with deliberative architectures by altering the mission on the fly. As a result, the autonomous vehicle should be able to modify the existing plan based on the real-time sensory information. Reactive control is the coupling of action and perception, to assure autonomous and accurate response of the vehicle, independent of the world surrounding dynamics.

Unlike the deliberative architectures, the behavioral architecture does not directly require a world model, and can thus be considered more simple to build [12]. This sense-act principle makes reactive architectures robust and assures the independence of dynamical environments. Two research questions regarding reactive behaviors occur when designing control architectures: how are the specific behaviors defined, and how should one decide which behavior should be carried out when several behaviors are active. These questions forms the base for the choice of architecture. The first question may be answered with a subsumption-based architecture, i.e. architectures with defined rules for behaviors being active. An example on subsumption-based architectures can be found in [11]. To solve the problem of conflicting active behaviors, an arbitration mechanism can be implemented [36]. Such a mechanism is implemented and described using different approaches in [36].

2.1.3 Hybrid Architectures

Deliberative architecture has the disadvantage of not being able to react within a desired amount of time if the vehicle is exposed to other environmental dynamics than the pre-planned mission. The vehicle is only able to react if it is not exposed to any external threat or unexpected action. The reactive layer, on the other hand, enables the vehicle to avoid threats, and modify the initial plan to execute the mission independent of the environmental dynamics. The disadvantages of reactive architectures are the difficulties of taking into account and designing reactive behaviors, and the corresponding actions to fulfill them. As an attempt to benefit from the advantages of each architecture, and reducing their disadvantages, a hybrid control architecture has been developed. Hybrid architectures combine the properties from deliberative and reactive layers, to increase the utility of the enhanced system. Hybrid systems incorporate a coordinating mechanism responsible for regulating the behavior execution, similar to the arbitration mechanism of reactive architectures. These architectures are composed of three layers:

1. Deliberative layer: Used for obtaining a vehicle with a predictable function. It is also responsible for replanning on demand from the other two layers or by the robot, forcing the vehicle to show a predictable behavior [36]. The mission planning will take place in

this layer.

2. Functional reactive layer: Used for obtaining a quick response action to situations that the system is not able to predict, in other words, the handling of contingency. Contingency is an unaffected event or situation that affects the execution of the mission. Examples of contingency handling are avoiding obstacles that are not predicted along the path.

3. Control execution layer: This layer is the logic that manages the order of the task execution, i.e. the coordinating mechanism. It is responsible for enabling, disabling and parametrizing the low-level behaviors [12]. The intelligence of this layer is the main difference between the several existing hybrid architectures, and typical approaches are fuzzy-implemented behaviors and hierarchical classifications of the behaviors [10].

Hybrid systems have been successfully applied to AUV applications, examples and comparisons are studied in [37]. Figure 2.1 is a decomposition of the hybrid control architecture based on the above statements.

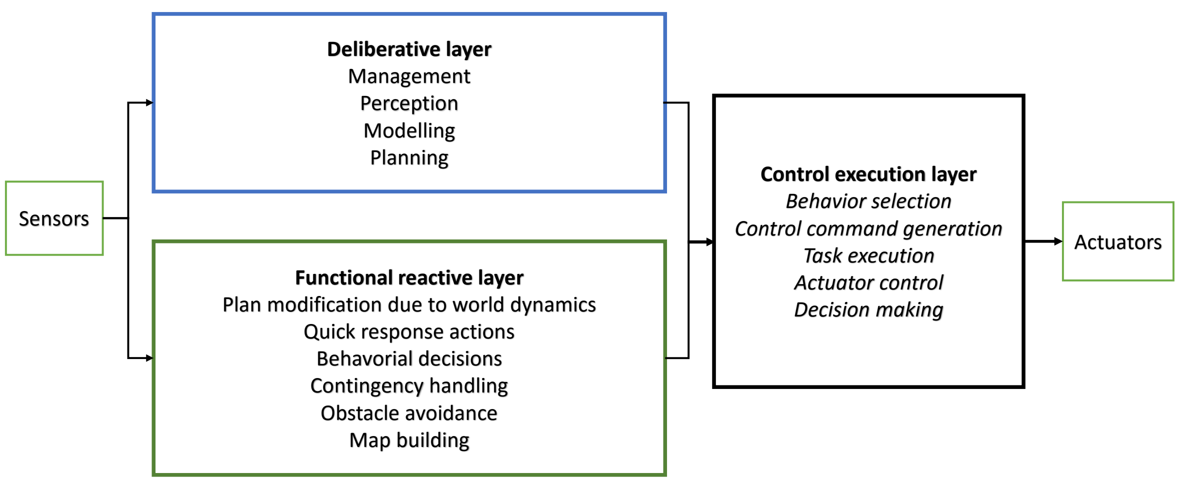


Figure 2.1: *Decomposition of hybrid control architecture into layers 1-3. The deliberative layer models the surrounding dynamics and takes care of the slowly-moving behaviors of the vehicle. The functional reactive layer changes the plan according to real-time sensory information that triggers a reactive behavior. The control execution layer decides which behavior should be carried out according to the coordinating mechanism of the architecture. In addition, behaviors are translated to actuator commands and task execution.*

The hybrid control architecture is a balance between carefully planned, optimal actions and quickly-responding, satisfying actions. The first assuring achievement of the desired mission, and the second to account for a dynamic, unknown environment. The deliberative control provides the achievement of high-level goals, while reactive control ensures adaptation according to the dynamic world. The absence of either control could result in the vehicle being unable to satisfy both the goal and the adaptive mission objectives [58].

Chapter 3

Hybrid Agent Architecture

Based on the hybrid autonomy architecture explained in Section 2.1.3, this chapter proposes a Hybrid Agent Architecture (HAA) for autonomous ROV interventions. A hybrid approach is considered adequate for ROV control purpose, where deliberate control takes parent decisions, and the reactive layer takes control of the vehicle when necessary. The purpose of the agent is to introduce a higher level of autonomy in ROV applications, by applying theory from previous work [36], and improving it through further research, simulations and field trials. In that way, this thesis presents a pioneering agent architecture for autonomous control of ROVs. Section 3.1 describes the structure of the agent and the different layers in detail.

The architecture is based on behavior-based principles; the desired vehicle behavior is divided into separate sub-behaviors, or states, where each state represents a particular aspect of the vehicle autonomy. This allows the user to add or remove states dependent on the ROV mission, and thus the architecture appears straightforward and open to modification. The architecture is similar to a state machine, deciding which sub-behavior the vehicle is in, based on sensory information and outputs from the deliberative and functional reactive layer.

The main work behind such an architecture lies in the definitions and optimizations of each state. This chapter proposes possible deliberative and reactive states for simple ROV interventions. Due to the simplistic architecture, infinitely many states can be added to the framework, opening for a variety of autonomous interventions. This thesis considers six slowly moving, deliberative states (Section 3.2-3.7) and one reactive state (Section 3.8), assuring fast response of the vehicle when exposed to the unknown environment.

3.1 Structure of HAA

The agent architecture is built to control the ROV when performing a general subsea operation, to approach and localize a Structure of Interest (SOI) located on the seafloor. Figure 3.1 is a diagram of the agent architecture based on the diagram presented in Section 2.1.3, but customized for this type of ROV mission. The diagram organizes the content of this section.

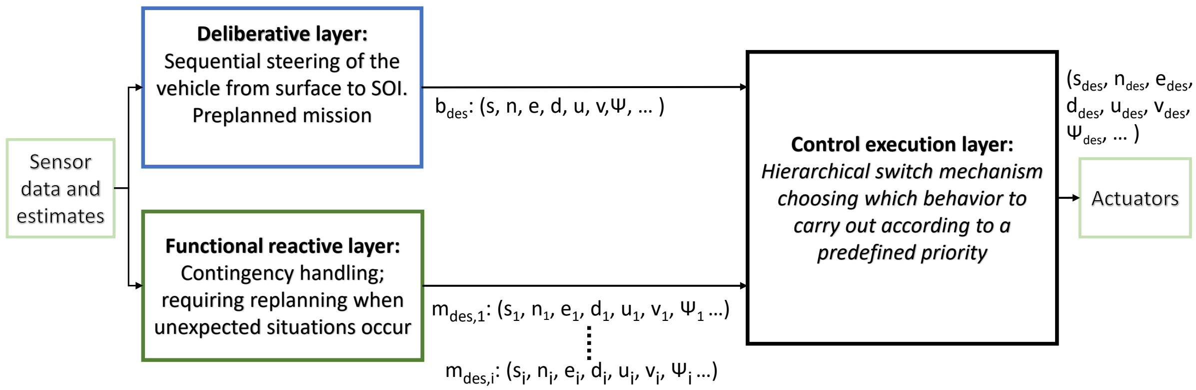


Figure 3.1: *Decomposition of the hybrid agent into three layers, based on Figure 3.1. This diagram is specialized for approaching and locating an SOI on the sea bottom. Each layer outputs a behavior in terms of parameters described in Table 3.1. $m_{des,i}$ are the active reactive behaviors, while b_{des} is the active deliberative behavior.*

The output behaviors from the two first layers are given by parameters described in Table 3.1. The reference frame refers to the coordinate intertial reference frame in which the variables are given. The different reference frames are defined in [46].

Parameter	Description	Reference frame
s_i	Desired state	-
n_i	Desired North position	UTM
e_i	Desired East position	UTM
d_i	Desired depth	UTM
u_i	Desired surge velocity	Body
v_i	Desired sway velocity	Body
ψ_i	Desired heading	UTM

Table 3.1: *Parameters defining the active behaviors in the agent architecture. The denotation* i *presents which exact behavior is active, since the reactive layer may output conflicting behaviors at the same time.*

In addition to these parameters, the behaviors can require changes in other variables, such as the ROV altitude. The inputs and outputs of the layers are further described in the description of each specific behavior (Section 3.2-3.8).

3.1.1 Deliberative Layer

The deliberative layer accounts for the slower varying components, that is taking the ROV from the surface to the SOI. This layer consists of six possible behaviors, following a sequential structure according to the mission anatomy. Only one deliberative state can be active at a time, due to this sequential steering. Each state has a set of conditions that must be satisfied to be active, as well as conditions to be fulfilled to progress to the next state. The deliberative layer is organized as a case structure, where each sub-behavior is implemented as a block. The block represents the planning in the deliberative structure ($sense \rightarrow plan$) and hence translates sensory information into high level desired actions. The deliberative planning consists of simple logic expressions with temporal and spatial conditions being either True or False. An example of such an expression is given as pseudocode in Equation 3.1, where X represents a time-dependent condition.

$$\begin{aligned} &if(depth == 80m \quad \&\& \quad timer >= 10s) \\ &X = true; \end{aligned} \tag{3.1}$$

Such simple commands from the deliberative layer are practical for an early phase design of autonomous ROV control. It opens for simplified debugging, fault detection and improvement of the states. Notice that only the active state receives information from the sensors in the deliberative layer.

3.1.2 Functional Reactive Layer

The reactive layer will constantly listen to the sensor input, independent of the active state. This, because as opposed to the deliberative layer, reactive states are triggered by specific sensor inputs, while the deliberative states are triggered by state transitions. The reactive layer consists of the fast-moving behaviors, in this case, Obstacle Avoidance (OA), further described in Section 3.8. A reactive behavior requires little knowledge on the environment and will be active as soon as it is triggered by its respective sensor output, regardless of the deliberative state that is active. When a reactive behavior is triggered (sense $\rightarrow$), the agent architecture makes it possible to modify the existing plan, to account for unexpected situations.

3.1.3 Control Execution Layer

An important aspect of this agent is that any reactive behavior will have priority in the switching mechanism: the control execution layer. The reactive layer will thus have control as soon as an obstacle is detected. The desired actions from the deliberative and functional reactive layer are sent to the actuators through this layer ($\rightarrow$ act). Due to the sequential steering from the deliberative layer, it will never require several deliberative behaviors at the same time. As a result, the control execution needs only to choose between the reactive and deliberative behaviors. This layer is thus organized as a hierarchical switch, selecting the desired reactive behavior as soon as it is desired from the functional reactive layer. When several reactive behaviors are added to the agent, the control execution layer would need to prioritize the possible reactive behaviors, or possibly require two behaviors at the same time from the control system. The agent architecture is combined with a computer vision system for detection of obstacles. Chapter 4 will describe the details of the equipment and setup of the architecture. The behavior definitions, conditions and simplifications are further explained in Section 3.2-3.7.

3.2 Launch

The launch is the state where the entire mission is initialized. The vehicle must be deployed in the surface before the autonomous mission can take over control, and as soon as it does, Launch is activated. The purpose of this state is to stabilize the vehicle at a certain depth while verifying that all systems are ready for mission control. A Launch state will be necessary for all types of missions, and the state is created in a general structure, such that it might be applied to different types of applications. The conditions for this state to be active are that the vehicle must be deployed, all sensors are available and working, the vehicle has communication with the onboard station, and the autonomy mission is started by the operator. Then, the vehicle is sent to a certain predefined depth, and when it has stabilized here for a certain amount of time, the state machine switches to the next state.

3.3 Descent

Descent brings the vehicle from a certain depth below the surface until it has a constant specific altitude above the seafloor. The vehicle moves exclusively in the downward direction throughout the state, with the purpose of minimizing the control parameters and thrust, as well as the directions in which collision may occur. The vehicle moves with a steady heave velocity and is kept constant in the other DOFs. As the Descent state is started, the vehicle is set to reach a specific depth, and depth control is activated, i.e. it is controlled using the estimated and desired depth. The reason for using depth control as opposed to altitude control, is that the Doppler Velocity Log (DVL) [1] sensor does not receive signals from the sea bottom before it has an altitude of 30-40 meters. As long as the vehicle is above this altitude, there exists no measurement nor estimates of the altitude. Therefore, Descent takes the vehicle to the sea bottom using depth control until the DVL sensor receives signals from the sea bottom, and then switches to altitude control. The estimated and desired altitude are then used as the control input.

When a desired depth or altitude is sent to the control system, the guidance system translates these values to several sequent setpoints from the current altitude until the vehicle reaches the desired value. This assures a smooth input and more stable control since the control error will always be kept small. A time delay is implemented in the agent on purpose when the DVL sensors start receiving signals from the sea bottom, keeping the vehicle in depth control. In that way, a stable estimate of the altitude is assured before switching from depth to altitude control.

When the vehicle reaches the predefined altitude, it must stabilize at that position before the autonomy agent switches to the next state. If the DVL sensor does not receive signals from the seafloor, and the vehicle is approaching the seafloor with less than 30 meters altitude, the autonomy layer sends a message to the operator that the DVL status is idle.

A drawback in this state is that no sensors nor cameras are pointing in the downward direction. Hence, obstacle detection is impossible as the ROV 30k is today. However, this will be a problem of any ROV, and to be fully autonomous, sensors would be needed in all directions. In addition to the above description, this state has the opportunity of slowing down if, somehow, more cable is not available. This may be caused by the human error due to manual cable control, large currents, or the cable being stuck in some obstacle. This state uses the desired thrust to detect cable tension (large thrust implies a tight cable), and if

[1] Sensor for measuring the ROV velocity relative to the seabed or water layer, further explained in Section 4.2.2

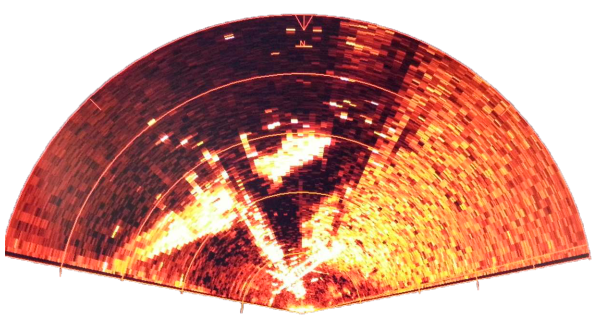
the thrust exceeds a certain limit, the vehicle slows down. This feature is only implemented in this state, but it could also be implemented as a distinct reactive behavior, since cable tension may occur throughout the autonomous mission. In addition, one could consider the use of autonomous winch handling for releasing cable according to the movement of the ROV.

3.4 Transit

This state corresponds to a trajectory tracking or path following state. The purpose is to bring the vehicle sequentially through all the desired, user-specified waypoints. That way, the vehicle is brought autonomously closer to a SOI, knowing the almost position of the SOI. The possibility of having several waypoints opens for operations where the vehicle must move around known structures on the seafloor, or simply to replan the path after having launched the autonomous mission. The waypoints are defined by North and East coordinates, as well as a desired altitude above the seafloor. There is no limit in how many waypoints this state can require, and the waypoints are defined in the desired order as elements in an array. The control system provides a heading based on the direction of the desired path through an Integral Line of Sight (ILOS) algorithm [5]. This means that the heading is always set to assure the vehicle is looking in the forward moving direction. As soon as the ROV reaches the desired altitude, it is kept constant until another altitude is desired.
A distance of acceptance is created around the desired waypoints, and as soon as the vehicle is within this distance, the desired waypoint changes to the next one. This is done until the vehicle has been through all the waypoints in the array, and the autonomy layer then switches to the next state as the vehicle reaches the last desired waypoint. The conditions for the vehicle to be in this state is that the last state was Descent, and the vehicle has reached an adequate altitude relative to the sea bottom. The simplifications of this state are that the vehicle must be within a certain distance of the desired waypoint, instead of a radius of acceptance, which could make the trajectory and waypoint transitions more smooth.
Transit enables a large spatial coverage of the ROV when operations take place where an assisting vessel could not go, or when the vehicle is operating in a sea mining or oil reservatory, and known structures on the sea bottom must be avoided. The only spatial constraint in the ROV will be the cable. Transit will assure the vehicle moves towards the SOI in a safe manner after having reached the sea bottom in the previous state.

3.5 Sonar Tracking

Sonar Tracking (ST) is a state where the vehicle is looking for the SOI located on the sea bottom, using information from the sonar[2] sensor. This thesis does not consider the processing of sonar data, but it assumes and simulates the input from the sonar, and uses it to navigate the vehicle. Sonar scans the sea bottom and returns a distance to and relative angle to the sea bottom, and all objects in the sonar range. It is possible to locate an object when it appears in the sonar, since the intensity of the acoustic reflections from the seafloor is recorded along the direction of motion [48], and by doing so, the sonar creates an image of the sea bottom. Figure 3.2a shows a sonar scan of an old bomber wreck in the Trondheim fjord. Its contours are clearly visible, and since the distance to every point in the picture can be extracted from the sonar data, it is possible to understand the use of such a sensor for localization of SOIs.



(a) *Sonar scan of the old English Halifax bomber from World War II, taken by NTNU scientists in November 2014 [5]*

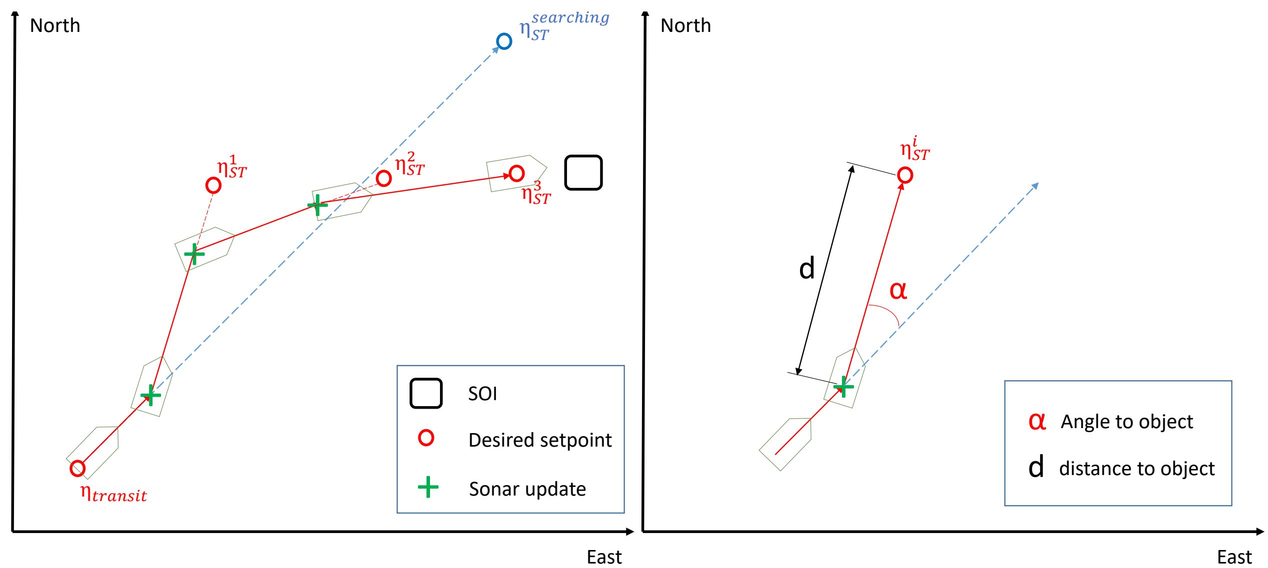
(b) *A similar Halifax bomber from 1944, taken from [54]*

Figure 3.2: *Sonar scan of an old Halifax wreck in the Trondheim fjord, and picture of a similar Halifax bomber. Notice the clear reconstruction of the object in the sonar scan.*

In other words, the sonar can locate any object on the sea bottom with a distance and angle relative to the sonar position. This way, since the sonar is mounted on the ROV, it is possible to use the sonar for feedback navigation towards the SOI as soon as it is detected. The detection of the SOI is assumed possible through computer vision processing of the sonar data, recognizing the size and shape of the object. This state defines a desired setpoint based on the last transit coordinate, where the sonar should be looking for the SOI. As soon as the SOI is detected in the sonar, a new setpoint is calculated based on the distance and angle information from the sonar. As soon as the vehicle moves and the distance and angle to

[2]Sensor for acoustic scanning and detection, further explained in Section 4.2.1

the object changes, this is updated as a new setpoint, and the change in setpoint is made immediately. When the vehicle is within a certain limit from the object according to the sonar, the autonomy layer switches to the next state. One simplification of this state is that since no sonar processing is performed, the setpoints are set manually. However, since the state is designed for an input according to raw sonar data, in terms of distance and relative angle, the vehicle will be able to navigate towards the SOI as soon as such processing is available. See Figure 3.3 for an illustration of the state.



(a) *Illustration of the ROV from the last transit waypoint, $\eta_{transit}$, until the state ST is complete.*

(b) *Definitions of parameters used as simulated input to the autonomy layer from the sonar data, defining the new desired setpoint.*

Figure 3.3: *Illustration of the state ST*

$\eta_{ST}^{searching}$ in Figure 3.3a is set as the initial waypoint, and as the vehicle moves towards this point, it simultaneously waits for updates from the sonar. As soon as an object is detected in the sonar, a new setpoint is created in the autonomy layer, η_{ST}^{1}. The vehicle starts moving towards this new setpoint instantly, and repeats this action for every new sonar update, until finally, it reaches the SOI with an adequate ahead distance, in this case in η_{ST}^{3}. Figure 3.3b illustrates the input from the sonar to the autonomy layer; the angle of the object relative to the sonar position (on the ROV), α, and the distance to the object from the current ROV position, d. These parameters creates the new setpoint, η_{ST}^{i}. As long as the distance to the SOI is larger than a threshold value, the ST state will be active. When the vehicle is within this threshold value, it should be sufficiently close to the SOI for it to appear in the stereo cameras, and the autonomy layer may switch to the next state.

The conditions for this state to be active are that the Transit state must be completed, and thus the SOI is within a range of 5 to 100 meters. The approximate location and shape of the SOI must be familiar to the operator, such that it is recognizable in the sonar. In addition, the vehicle must be at an altitude of 40 meters at the most from the seafloor to get any reflections in the sonar. However, considering the Transit state is completed, this conditions must already be satisfied. This state also has the opportunity of changing the desired altitude together with the updated sonar setpoints.

3.6 Camera Tracking

When the vehicle has localized the SOI through ST, it is located in an adequate distance to the object for applying camera vision techniques for navigation, and Camera Tracking (CT) is activated. First of all, it is desirable to have more accurate position estimates when being this close to an object on the seafloor. The vehicle should approach the object further, and to assure precise positioning and avoid a collision, one should assure the position estimates are extremely accurate. This can be done by introducing the stereo cameras in the low-level control system. Information from the cameras may be used to determine the movement of the vehicle relative to an object. This camera vision technique should be implemented as an observer that is switched on when an object is available in the stereo cameras. The implementation and background of this observer are studied in detail by Stud. Tech. Michele Gazzea (see Section 1.1.5). Further, the distance to the object should be measured with high accuracy, to use it as feedback to the controller. This is also done by estimating the distance to the SOI using images from the stereo cameras and computer vision methods, primarily investigated by Ph.D. candidate Marco Leonardi (see Section 1.1.5 and an abstract of the work in Appendix B).

When the camera based observer is enabled, and the stereo images can propose a distance to the object, the vehicle is ready to approach the SOI further. This should be done in a steady, slow manner, to assure safety and avoid damage to equipment. The vehicle should approach the SOI until it is sufficiently close to perform intervention/inspection, whereupon the autonomy switches to the next state.

The conditions for CT to be the active state is that ST is complete, and the vehicle is located in a distance to the SOI such that it appears in the stereo cameras. The implementation of this state in the autonomy layer is currently simplistic in structure, since the main work lies in the low-level control system, and the above work is not yet implemented in the control system. The agent architecture should use the distance from the stereo cameras to approach the SOI, assuming that the precise camera-based observer is enabled. Since the distance to the SOI is currently rather insecure, this state does simply keep the vehicle in front of the SOI while manually verifying the distance to the SOI. Due to the above limitations, this state has not been properly tested yet, but as soon as the observer is implemented in the control system, this state will be possible to test and apply on a real system. The purpose of CT is to approach the SOI in a safe manner, by using accurate position estimates and precise sensors (stereo cameras) for navigation. The vehicle should be brought sufficiently close to the object to perform intervention or inspection.

3.7 Intervention/Inspection

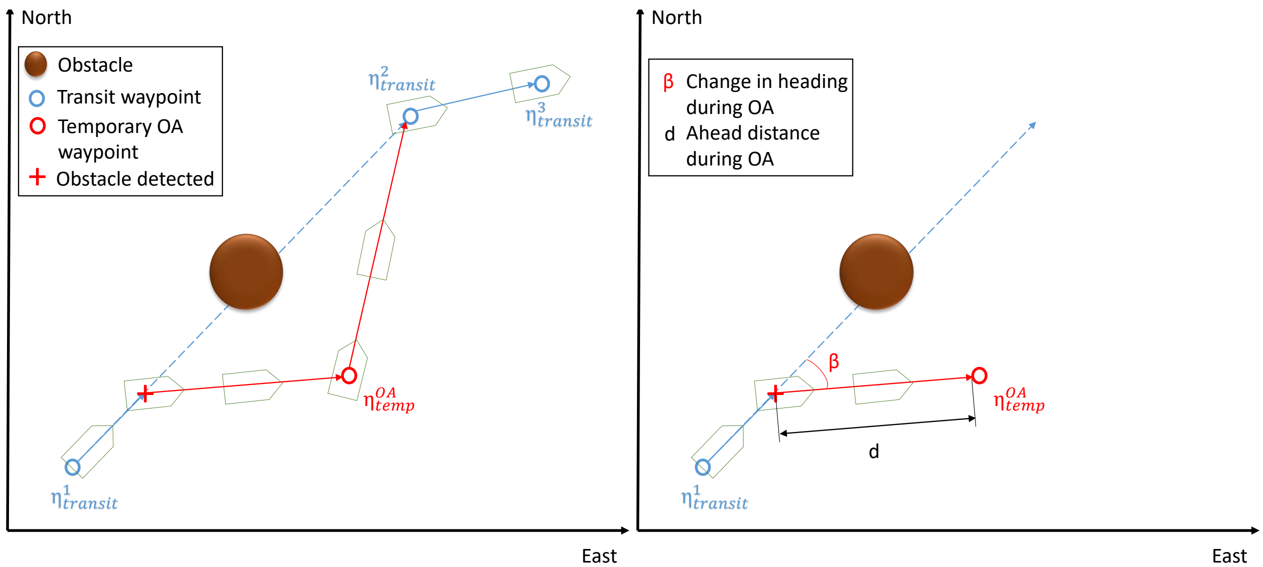
As opposed to the previous states, this state is highly dependent on the type of operation. It should cover a high variety of tasks, and would, in a progressive point of view, be divided into several behaviors, each optimized for the current operation. In that way, this state is the only one that distinct the mission anatomy according to the specific application of the architecture. This means that the agent architecture proposes a general approach to performing autonomous operations using ROVs. This state, however, needs modification according to the specific operation. This reduces the time and complexity when designing an autonomous mission for a specific task, hence only one state needs modification. Such specific tasks may be everything from inspection of a subsea pipeline, where the vehicle should follow the pipeline and perform inspection with the cameras, to performing an action using the manipulator arm, such as opening a valve. Using the latter one as an example, it is easy to see that this opens an entirely new field of study, where the autonomous handling of the manipulator arm must be considered. In addition, recognition of the valve must be studied, to know the exact valve position before opening. There has been researching already carried out in this area, one of them is presented in [41]. However, more specific research is needed, and thus no autonomous intervention is performed in this thesis. To recap, this state covers a large area and a variety of tasks which are not considered in this thesis. The agent architecture proposes a framework for autonomous intervention, which must be specifically modified for each applied operation. As a result, this state is currently a simple stationkeeping mode in front of the SOI.

3.8 Collision/Obstacle avoidance

The above states are slowly moving actions performed sequentially and according to a known environment, i.e. deliberative behaviors. To test the functional reactive layer in the agent architecture, one significant reactive state has been implemented: Obstacle/Collision Avoidance (OA). Collision avoidance is a wide field of study, where several solutions have already been studied and tested. The groundbreaking of this thesis, however, is the integration of such an algorithm in an autonomy framework, and the detection of obstacles using stereo vision. Therefore, this state is rather simplified concerning how the obstacle is avoided, but the final result of having an embedded stereo camera detection and following avoidance is complex and pioneering.
The work of Ph.D. candidate Marco Leonardi is summarized in Appendix B, and proposes an obstacle detection based on a computation of disparity maps from the stereo image pairs. A 3D point cloud is extracted from each disparity map, which, with proper noise filtering of false correspondence matches, represents the obstacles. The point cloud makes it possible to estimate the distance to and the orientation of the obstacle. The ROV is equipped with stereo cameras pointing straight forward to simplify detection of obstacles. The image stream is sent to a desktop computer onboard the assisting vessel, and the 3D point clouds are computed on the same computer. The computer vision output; the distance to and orientation of the object, was sent to the computer running the control system and autonomy layer using UDP communication[3]. Receiving information about an obstacle, the autonomy layer computes a new heading for the vehicle, based on the orientation of the object. That is, if the obstacle is left oriented, the vehicle should go right, and vice versa. If the vehicle reaches this new heading and still has an obstacle appearing in the cameras, the autonomy layer computes a new change in heading. When the obstacle does no longer appear in the cameras, the autonomy layer computes a temporary waypoint ahead and sends this waypoint to the current deliberate behavior of the vehicle.

Currently, OA is only tested during the deliberative state Transit, since this is when the vehicle moves significantly in the xy-plane and is most likely to experience a collision. The temporary avoiding waypoint is, therefore, sent into the Transit state as the next desired waypoint, and the vehicle starts moving towards this waypoint as soon as the obstacle does no longer appear in the cameras. As the vehicle reaches the temporary waypoint, the next Transit waypoint is set, and the vehicle continues from where it was interrupted due to OA. The state is illustrated in figure 3.4.

[3]Protocol for information transfer, further explained in Section 4.1.4



(a) *Complete OA during Transit state. The OA state creates a new, temporary waypoint, η^{OA}_{temp}, for the vehicle to avoid the detected obstacle. Reaching this waypoint, the vehicle may return to the desired Transit waypoint.*

(b) *Temporary avoiding waypoint is defined by a user-specified ahead distance and angle. These parameters could ideally be calculated in the agent when more information from the stereo camera is postprocessed.*

Figure 3.4: *Illustration of the state OA*

Figure 3.4a shows the performance of the vehicle when an obstacle is detected in the stereo camera video stream. The blue path shows the original Transit path, while the red path shows the actual path, i.e. the obstacle-avoiding path. As it appears in the figure, the obstacle is left oriented relative to the ROV when it detects the obstacle. As a result, the vehicle turns right, and the temporary avoiding waypoint, η^{OA}_{temp}, is to the right of the object. Figure 3.4b defines the parameters of this temporary waypoint; the avoiding angle, β, and the distance ahead, d, before the vehicle switches back to the originally desired waypoint in Transit. These parameters are currently predefined in the autonomy layer, but they could also be taken as an input from the computer processing the stereo images. Notice that if the obstacle still appears in the cameras when the desired heading, β, is reached, the vehicle turns additionally $\beta°$. This state has the opportunity of disabling information from the stereo cameras and manually warning the agent about obstacles, in case the information from the stereo images somehow is false.

The avoiding method presented in this state is rather simplified, and more complex avoiding paths should be considered for implementation. An example is the work done by Steffen Kørte in [25]. However, the aim of this thesis is not to study the path planning when avoiding

obstacles, but rather the autonomous implementation of such a reactive behavior. The agent architecture can be further used as an interface between complex avoiding algorithms, the camera vision output, and the actual control system.
The conditions for the vehicle to be in this state is that the computer vision output detects an object which is within a certain threshold distance. Today, the computer vision is only able to detect objects at a distance of maximum 4 meters, and as a result, the vehicle will avoid any obstacle that appears. When the vehicle has turned the desired amount of degrees, β, and obstacles does not longer appear in the cameras, the control execution layer switches back to Transit.

3.9 Mission Planning

The purpose of the hybrid agent architecture is to sequentially steer the ROV using the deliberate states above, while reactively take action when unexpected situations occur, such as avoiding obstacles. The mission of localizing a SOI on the sea bottom is illustrated in Figure 3.5.

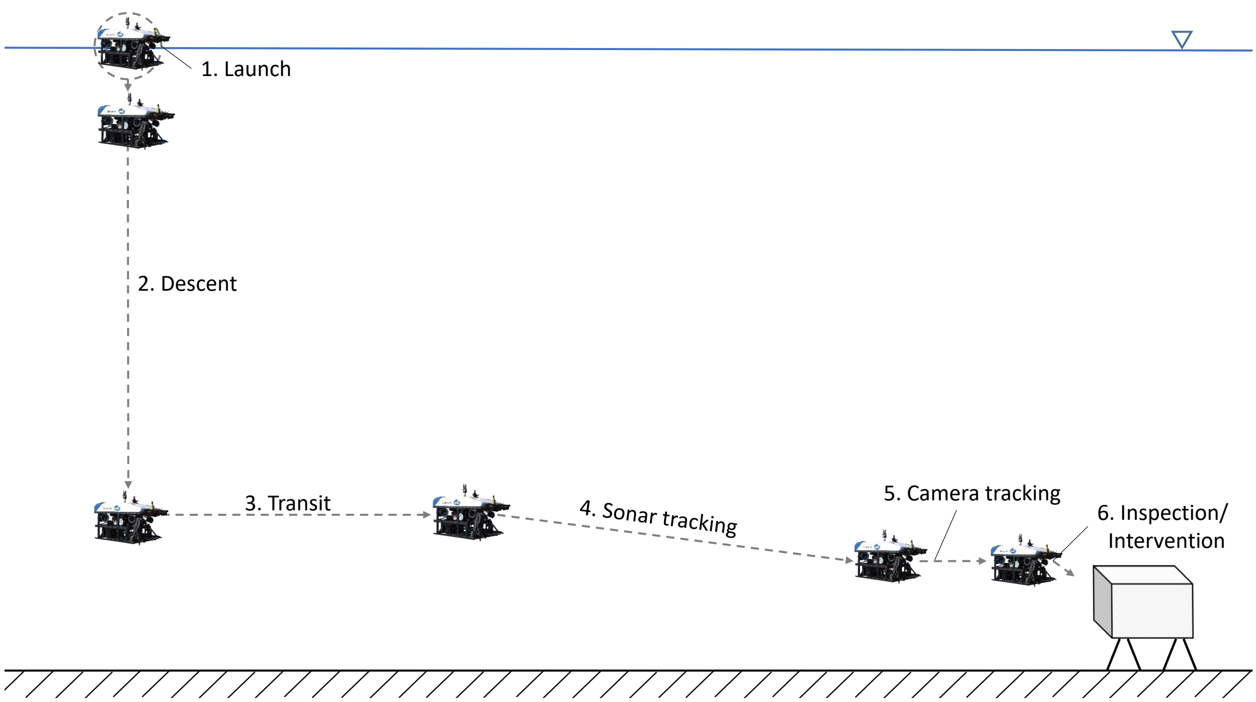


Figure 3.5: *Autonomous mission: approach and localization of SOI*

The agent sequentially switches the deliberative states as their conditions, waypoints and references are satisfied. While doing so, the agent continuously waits for messages from the computer vision output and is ready to avoid a collision as soon as anything is detected in the cameras. If an obstacle is detected, the agent instantly switches to the state OA, and when the obstacle is avoided, the vehicle continues with the autonomous mission. The mission is complete when the vehicle is located in front of the SOI, and the cameras can recognize the SOI. Due to reasons discussed in Section 3.6 and 3.7, the two latter deliberative states are merged into stationkeeping in front of the SOI. Appendix C applies Huffman's theory [49] for organizing the mission in a structural manner.

The mission is tested through software simulations and field trials in the Trondheim fjord. Implementation and experimental set-up is presented in Chapter 4, and the results are presented in Chapter 5 and 6.

Chapter 4

Experimental Setup

This chapter presents the implementation, equipment and experimental set-up of the testing of the mission control system. The agent architecture was tested through software simulations and field trials in the Trondheim fjord using the ROV 30k. The NTNU owned research vessel R/V Gunnerus was used as a surface vessel with a control container placed on deck. Figure 4.1 presents the information flow and equipment connection for the testing, inspired by [5] and [47].

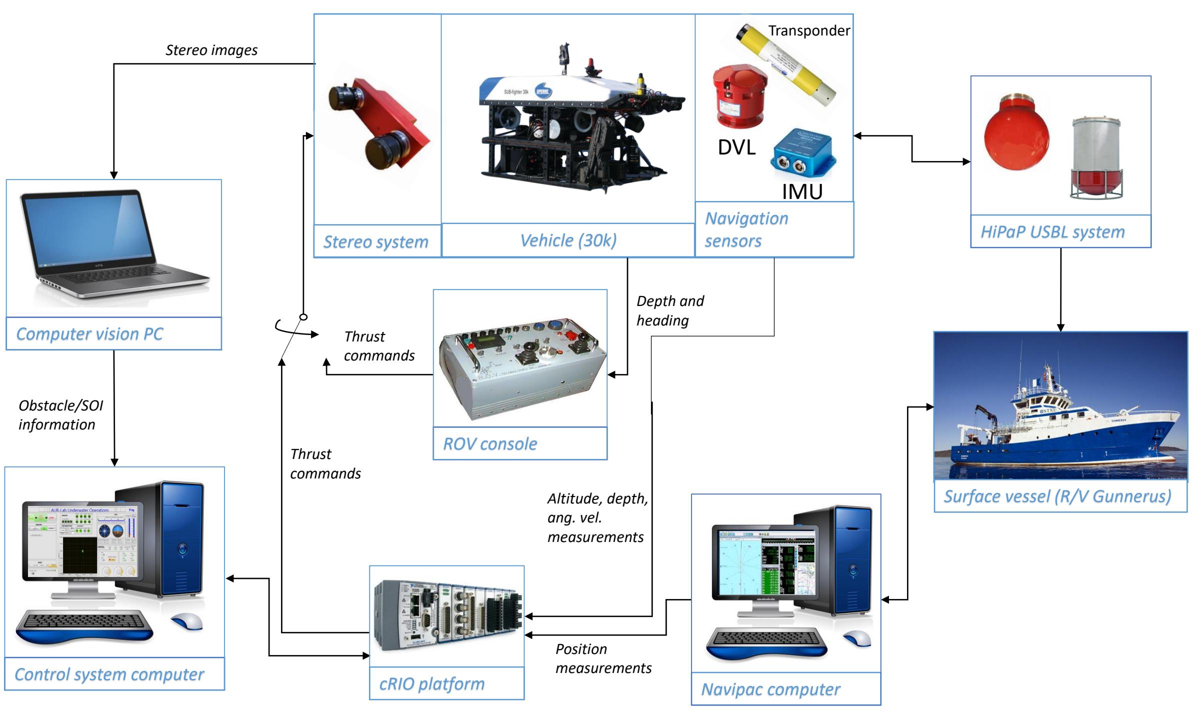


Figure 4.1: *Information flow between the equipment used for testing the agent architecture. Each block is described in detail in the following sections.*

When performing simulations, the cRIO platform (Section 4.1.1) is substituted with a software simulator (Section 4.1.2). When performing Hardware-In-the-Loop (HIL) simulations, the cRIO platform is connected to the same software simulator instead of the ROV. The switch between the thrust commands is a switch between manual/remote control and automatic control of the vehicle. The equipment in Figure 4.1 will be further described in the following sections.

4.1 ROV Control System

The ROV Control System has been developed by a team of MSc students, Ph.D. candidates, Professors, and postdoctoral researchers since 2010 [45]. The objective of the work was to develop a control system with user interface for DP, and tracking for use in real ROV missions, and the work is described in [47]. The system was deployed and tested on ROV Minerva during monthly cruises with R/V Gunnerus and tested in HIL simulations before deployment. The development of the control system is explained in detail in [5].

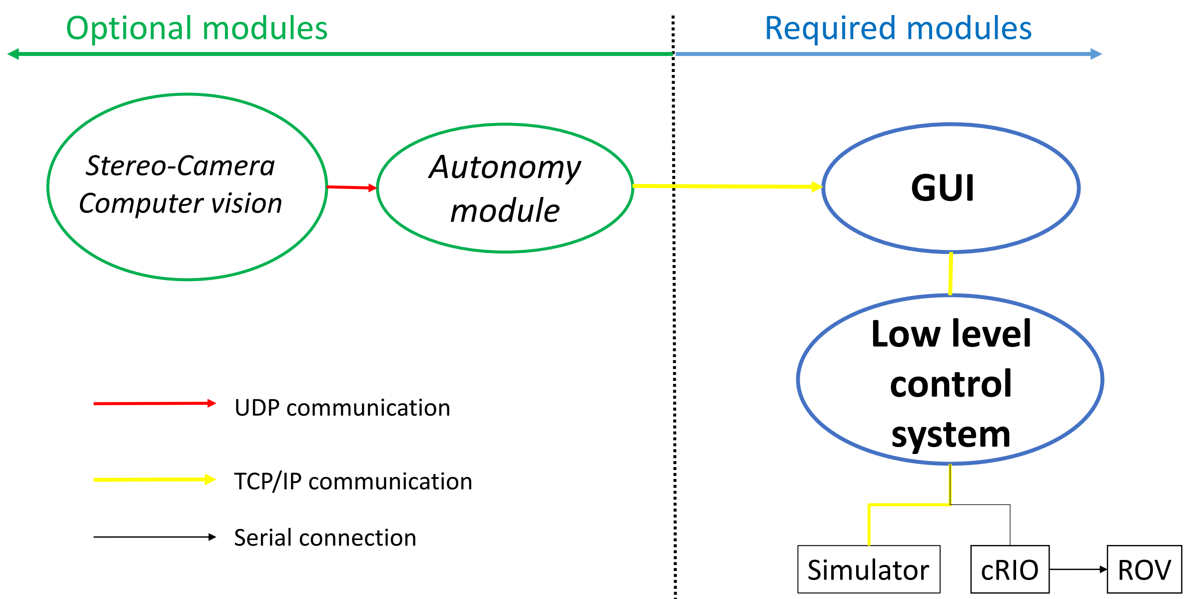


Figure 4.2: *Module configuration of (1)-(4).; Basic modules GUI and Control System, connection with optional Autonomy module and its additional module for computer vision. Communication between modules is stated with color coding, and further described in Section 4.1.3 and 4.1.4. Based on Figure 2.4 in [5].*

As illustrated in Figure 4.2, the DP system is built on two basic modules; *(1) Control System* and *(2) Graphical User Interface (GUI)*, and several optional modules added gradually during the development. The basic modules are necessary to run the system, while the optional modules present different applications of the control system, and can be activated and deactivated. This thesis implements the Hybrid Agent Architecture as an optional module, *(3) Autonomy module*, which is programmed separately from (1) and (2), and is input to (2) when activated. The *autonomy module* is enabled through (2) when the control system is initialized. (1) is connected to the hardware and simulator. In addition, an optional *(4) Stereo Vision (SV) module* is added for specific applications of (3), and is communicating only with (3). The connections and information flow between the modules use TCP/IP and/or UDP communication protocols (Section 4.1.3 and 4.1.4). Module (4) is further explained in Section 4.2.4 and Appendix B.

4.1.1 Software and Hardware Platforms

The DP control system is implemented on the real-time hardware (HW) platform Compact Reconfigurable Inputs and Outputs (cRIO), delivered by National Instruments (NI). The cRIO platform has two main processing targets: one real-time processor for communication and signal processing, and one Field-Programmable Gate Array (FPGA) for high-speed control [29]. The FPGA is a reprogrammable silicon circuit designed to deliver good performance in every part of the application when adding more processing [27]. The cRIO platform is programmed with Labview software (SW). The control SF is deployed on the cRIO platform, but a host computer provides the GUI. Labview is an FPGA development environment using graphical programming syntax, and in that manner allowing for more visuality and simplicity when programming a system with NI standards [28]. The advantages of using Labview are the high-speed, efficiency, high integration capacity with all HW and the possibility of modifying the code easily even during testing and simulation [47]. The control system is currently updated with the latest version of the SF.

4.1.2 *Verdandi* simulator

To test the control system before field trials, an ROV simulator (*Verdandi*) has been developed in parallel with the control system using the same SF. By deploying the control system and simulator on the cRIO platform, HIL simulations can be performed. The HIL simulator method incorporates hardware components in the numerical simulation environment, which give results with better credibility than pure numerical simulations [18]. *Verdandi* is a detailed, high fidelity, mathematical model of the ROV, disturbances, and measurements. Software simulations are possible by connecting the control system directly to the simulator using

TCP/IP connection. Given the accurate mathematic model in the simulator, SF simulations give realistic responses of the vehicle when testing different modules. This thesis uses SF simulations to verify and test the hybrid agent architecture before deploying the system on the cRIO for full-scale testing. The simulations are convenient for creating new modules in the control system since the SF does not need to be compilated between modifications. This opens for troubleshooting and debugging of the *autonomy module* without significant consumed time nor processing. The simulations have been a necessary tool for implementing the module in the control system, and the results are presented in Chapter 5.

4.1.3 Transmission Control Protocol

The simulator, control system, and its GUI are dependent on reliable communication to cooperate properly. Transmission Control Protocol (TCP) is used for that purpose, due to its reliability when transferring information. TCP guarantees that every message is received and that no information is lost on the way, by assigning every symbol of information a sequential value that is obtained throughout the transfer [9]. The TCP-module receiving the message must send a confirmation that the entire package containing the message has been received, and if it is not received within a certain amount of time, the system is restarted [30].

4.1.4 User Datagram Protocol

The *Autonomy module (3)* relies on crucial information from the stereo cameras (Section 4.2.4) through the *SV module (4)*. As soon as the camera detects an object, the computer running the control system should be notified to instantly create a new path. In this thesis is such a communication obtained by using User Datagram Protocol (UDP). UDP is a simple message-oriented network protocol for transferring information quickly. No message transfer is guaranteed, nor saved in the receiver platform. This makes UDP less reliable than TCP, but, on the other hand, fast and with low bit consume [9].

UDP is, therefore, easier to implement, and the control system is less dependent of the computer receiving information from the camera. UDP was chosen for OA purpose since UDP does not require a restart when a connection is lost, but rather a timeout. The system returns to sending and receiving messages after this timeout. TCP would require reinitialization of the control system which is inconvenient because the connection could be lost easily when two computers are communicating independently [30]. For further improvement of the system, one could consider the TCP instead. However, UDP connection is simpler and makes the systems less dependent on each other.

4.2 ROV SUB-Fighter 30k

The ROV used for testing the mission control system is the NTNU owned Sub-Fighter 30k (30k). 30k was designed by Sperre AS for tough conditions and is especially suited for marine archeology research [44]. See [45] for specifications of the vehicle and its equipment. The vehicle is operated from an onboard container carrying the necessary equipment for manual and automatic control. The container is placed in the stern of R/V Gunnerus (Section 4.3), together with a remotely operated winch, providing cable for the vehicle, and a manual console. The vehicle is operated and maintained by AUR-lab (Section 1.1.4) with help from the crew on Gunnerus.

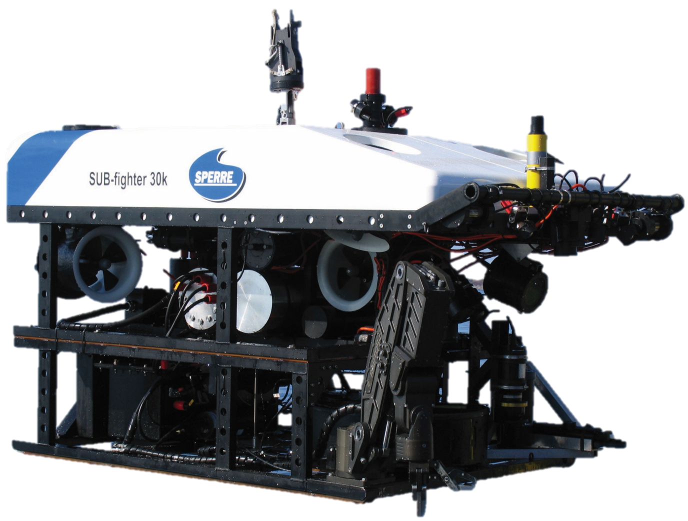


Figure 4.3: *ROV SUB-fighter 30k [53] ready for deployment in the Trondheim fjord*

Figure 4.3 shows the vehicle ready for deployment with equipment specified in [45]. The yellow transponder is attached for communication with the High Presicion Acoustic Positioning (HiPAP) system on Gunnerus (Section 4.3.1). The ROV currently has three auto functions; auto depth/altitude, auto heading and Dynamic Positioning (DP). Specific sensors used for this thesis are explained in Section 4.2.1-4.2.3.

4.2.1 Sonar

Sonars use sound propagation for underwater navigation or scanning and detection. Sonars consists of a transducer, a device that can transmit and receive acoustic pulses ("pings"). The transducer sends out single-beam pulses and listens to the reflections/echoes of the pulse. The sonar can measure the distance to an object, by converting the time for the sound echo to return, into a range, knowing the speed of sound. The forward-looking sonar on 30k used for scanning of the seafloor is a Kongsberg Simrad Mesotech made sonar, which represents high-resolution single-beam scanned sonars [24]. The high resolution is achieved by increasing the sampling period, improving the signal-to-noise ratios, by using small pulse lengths and widen the bandwidth of the transducer. Parameters for the sonar can be found in [24]. The Kongsberg Simrad sonar requires MS 1000 processing software to operate equipment and process data. A range of maximum 30 meters is used for the autonomous mission to assure the quality of the sonar scanned objects when using it for feedback.

4.2.2 Doppler Velocity Log

DVL is a sensor sending out 4-beam "pings" and measuring the resulting response regarding frequency shift, i.e. Doppler shift [20]. This shift is translated to a velocity, relative to a reflection point. The DVL purpose is to calculate the ROV velocity relative to the seabed or the water layer. By doing so, the ROV underwater position is possible to estimate. The DVL has two main conditions; Bottom-Lock (BL) and Water-Lock (WL). BL is when the vehicle is sufficiently close to the sea bottom and can measure its velocity relative to the seafloor. WL is the alternative to BL, i.e. when the vehicle measures its velocity relative to the water. BL is used as a condition for some of the states during the mission since it is crucial for the ROV to know its altitude as it reaches a certain depth. BL is not achieved before the vehicle reaches this certain depth, and until the DVL is in BL condition, the ROV is positioned using the HiPAP system (Section 4.3.1) on R/V Gunnerus. The HiPAP system becomes inaccurate as the ROV distance to the operating vessel becomes large, which will cause wear and tear on the thrusters if the distance does not decrease, or the vehicle achieves BL for DVL navigation.

4.2.3 Inertial Measurement Unit

Inertial Measurement Units (IMUs) are devices that use accelerometers and gyroscopes to give a measure on linear and angular motion [57]. They can be used for dead reckoning when the position measurements from other sensors are lost, but they are primarily used for giving a measure on the orientation of a body. On 30k, the IMU calculates the pitch and roll rotations using the accelerometers, and the yaw rotation (heading, ψ) using the gyroscope. Notice that accelerometers are based upon integrating the measured acceleration, which may give deviations in positions due to accumulations of small initial errors [33]. As a result, IMUs are not used for direct positioning of the ROV, but rather a measure on the orientation and dead reckoning.

4.2.4 Stereo Cameras

AUR-lab has access to two stereo cameras for use on ROV 30k. Stereo cameras use multiple lenses with separate image sensors to capture three-dimensional (3D) images [56]. As a result, stereo cameras may be used to detect the width of, and length to, objects in the camera range. Computer vision techniques, combined with information from stereo cameras, allow for use in cars and vehicles with integrated autopilot. In this thesis, stereo cameras are used for recognizing objects, and detecting an obstacle in the camera range. The stereo system is considered for both detection of objects in the cameras, and object tracking for position reference. The two concepts are explained in more detailed in the Appendix B. A description of the processing of the images for detection of obstacles is given in Section 3.8.

4.3 R/V Gunnerus

R/V Gunnerus is an NTNU owned and operated research vessel from 2006. The vessel is suitable for research within a variety of fields, such as biology, geology, archeology, oceanography and technology [2]. The vessel is equipped with a DP system as well as a HiPaP 500 unit system (Section 4.3.1), providing perfect conditions for ROV operations for different purposes. See [2] for further details and specifications on the vessel. In this thesis, Gunnerus is a crucial part of the mission control system.

4.3.1 HiPaP system

To control an underwater vehicle, a precise positioning system is required such that both the vehicle and the onboard station knows the position of the vehicle at any time. Minerva and 30k are positioned using HiPAP when operated from R/V Gunnerus [32]. The HiPAP-system is delivered by Kongsberg and can utilize software for both Ultra-Short BaseLine (USBL) and Long BaseLine (LBL) principles [23].

LBL is a positioning technique that uses several transponders located at different, known coordinates on the seafloor. The term *long* represents the distance between the transponders on the seabed, and allows for a target within the group of transponders to determine its position, using the distance between itself and each transponder in the group. The term *baseline* represents the distance between the transponders. The advantages of using an LBL positioning system is its accuracy and the fact that LBLs are independent of water depth [43]. The disadvantages, however, are the time-consuming installation and calibration of the baselines, as well as the spatial constraint in fixed transponder positions. As a result, LBL networks are typically used for areas where there will be several surveys throughout a long time.

ROV operations require considerable independence of spatial limitations, which make LBL positioning systems highly inconvenient considering its spatial limitations as well as the high installation time. USBL systems, on the other hand, are significantly more practical for ROV operations, since the transceiver is mounted on the operating vehicle, and the transponder on the ROV (see Figure 4.3). This makes USBL positioning systems independent of the operation area and installation time. In a USBL system, an acoustic pulse is transmitted by the transceiver and detected by the subsea transponder (mounted on the ROV), which replies with its acoustic pulse [17]. This return pulse is detected by the shipboard transceiver. The time from the transmission of the initial acoustic pulse until the reply is detected is measured by the USBL system and is converted into a range. This range is further used for calculating the ROV position [16]. The HiPAP system installed on Gunnerus is a HiPAP 500 USBL positioning system with a propagated accuracy in position measurement of 0.85 m [26]. USBL becomes more inaccurate with increasing water depth, and the measurement update frequency decreases. Nevertheless, USBL is a better-suited positioning system than LBL for ROV operations [26].

4.4 Full-scale Experiments

The full-scale experiments were performed in the Trondheim fjord during the fall semester of 2016. The mission anatomy explained in Chapter 3 requires a specific Structure Of Interest (SOI) for approach and localization. In addition, the mission requires an obstacle on the path. During the field trials, two SOIs and obstacles have been used for this purpose; a shipwreck (Figure 4.4a) and a transponder tower (Figure 4.4b) located in the Trondheim fjord. When testing the reactive layer of the hybrid agent, the objects were used as obstacles. The vehicle was sent towards the objects purposely to test the detection and avoidance of the obstacle. When testing the deliberative part of the hybrid agent, the above objects were used as SOIs to approach and localize through sensor feedback (see Chapter 3 for further details). When testing the hybrid agent with both reactive and deliberative components, the objects were used as obstacles, and the SOI was simulated with coordinates. Figure 4.4 shows the two

(a) *3D model of the Herkules wreck produced from the stereo system. Herkules is a tug that sank in 1957 due to poor seamanship [1]. The wreck is what divers call a Donald Duck wreck; a rather intact wreck standing on the keel.*

(b) *Transponder attached to tripod forming a transponder tower [3]. The tower carries one of four transponders in the LBL network deployed outside TBS in 2013 [3]. The transponders are located 250 and 400 meters away from land.*

Figure 4.4: *SOIs and obstacles during full-scale experiments in the Trondheim fjord.*

objects.

Figure 4.5 shows the Herkules wreck in the sonar scan. As can be seen from the scan, the object is quite evident, and the lines represent the distance to the object. Thus, it is believed that with the right post-processing software, the SOI would be recognizable in the sonar scan and use for feedback navigation. This thesis, however, simulates the location of the sonar scanned SOI manually.

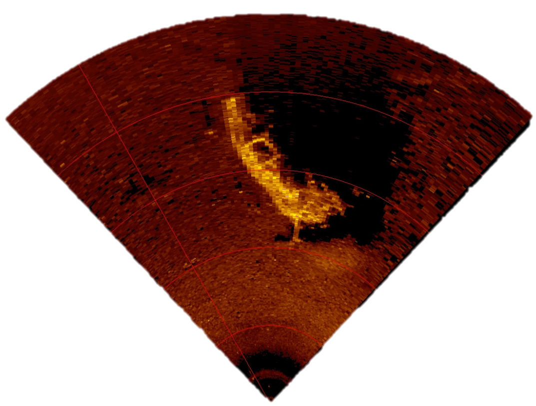
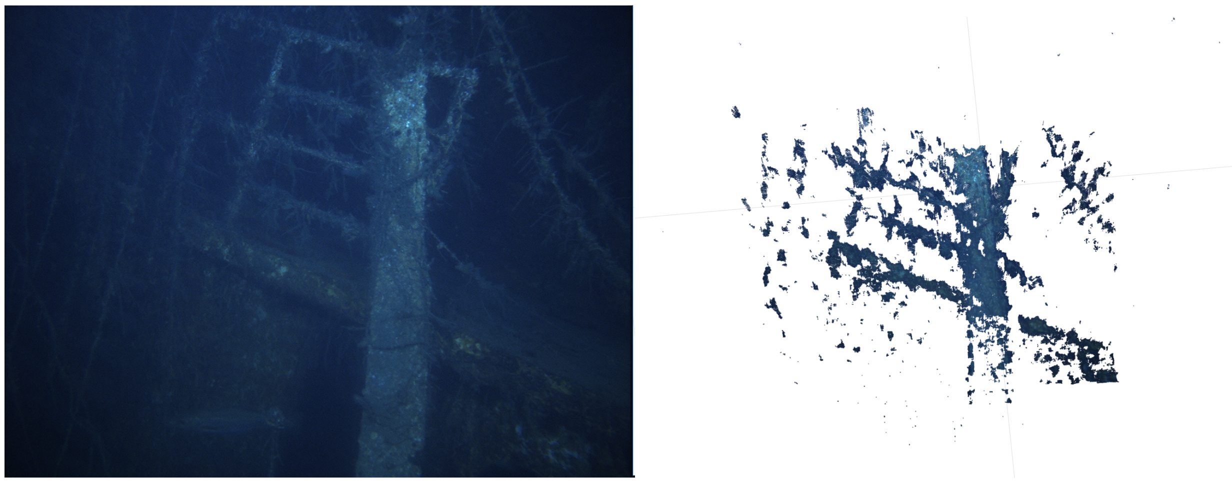


Figure 4.5: *The Herkules wreck in the forward looking sonar scan from the ROV. This is what the SOI looks like in the state ST (Section 3.5)*

Figure 4.6b is a 3D point cloud of the Herkules wreck when it appears in the stereo cameras. The 3D point cloud is created through postprocessing of the stereo image (Figure 4.6a), and is used to determine whether there is an obstacle ahead or not.



(a) *Stereo image of the stern of the Herkules wreck during OA* **(b)** *3D point cloud constructed from the image in (a) after postprocessing the image*

Figure 4.6: *Structures of Interest and obstacles during full-scale experiments in the Trondheim fjord.*

The mission anatomy in Chapter 3 is tested through software simulations (Chapter 5) and full-scale experiments (Chapter 6) using the equipment and set up explained above.

Chapter 5

Simulations

Pre-mission software simulations were performed of the vehicle, testing deliberative and reactive control both separately and simultaneously. Simulations were performed using *Verdandi* (Section 4.1.2), and manual or simulated input from the stereo vision system. This section presents the simulations in three separate trials, and Appendix D presents additional results from the simulations. The three trials tested the hybrid agent in the following manner:

Section 5.1: Reactive control; Collision avoidance

Section 5.2: Deliberative control; autonomous mission without collision avoidance

Section 5.3: Hybrid control; autonomous mission with collision avoidance

All sub-behaviors were tested, and the result of the trials is presented in the following.

5.1 Reactive control

Collision avoidance was simulated in several trials, all of which yielding identical results. This was done first of all to be prepared for field trials, assuring no untested scenarios would occur. Secondly, the experimental testing of such an algorithm is somewhat limited, since the vehicle must be purposely sent towards an obstacle. Testing of several sequent obstacles would, therefore, be difficult, since there would have to be two or more obstacles placed on one path. As a result, the avoidance algorithm is thoroughly tested in the simulations, and a simple collision avoidance test is performed in the full-scale experiment. Correlation in performance during simulations and field trials of one scenario would imply correlation in performance in every simulated scenario.

To simulate the warning of an obstacle, a manual switch was implemented in the agent, as well as a manual input defining the orientation of the simulated obstacle. In addition, the parameters defining the temporary avoiding waypoint had to be predefined (see Section 3.8 for parameter definitions). These were varied through several tests. Figure 5.1 shows position estimates ($\hat{x}$, $\hat{y}$, $\hat{\psi}$), measurements (x_m, y_m), and desired positions (x_d, y_d) of the vehicle in North-East (NE) plane when the obstacle is left oriented. The obstacle is avoided with an angle of 30° and an ahead distance of 5 meters.

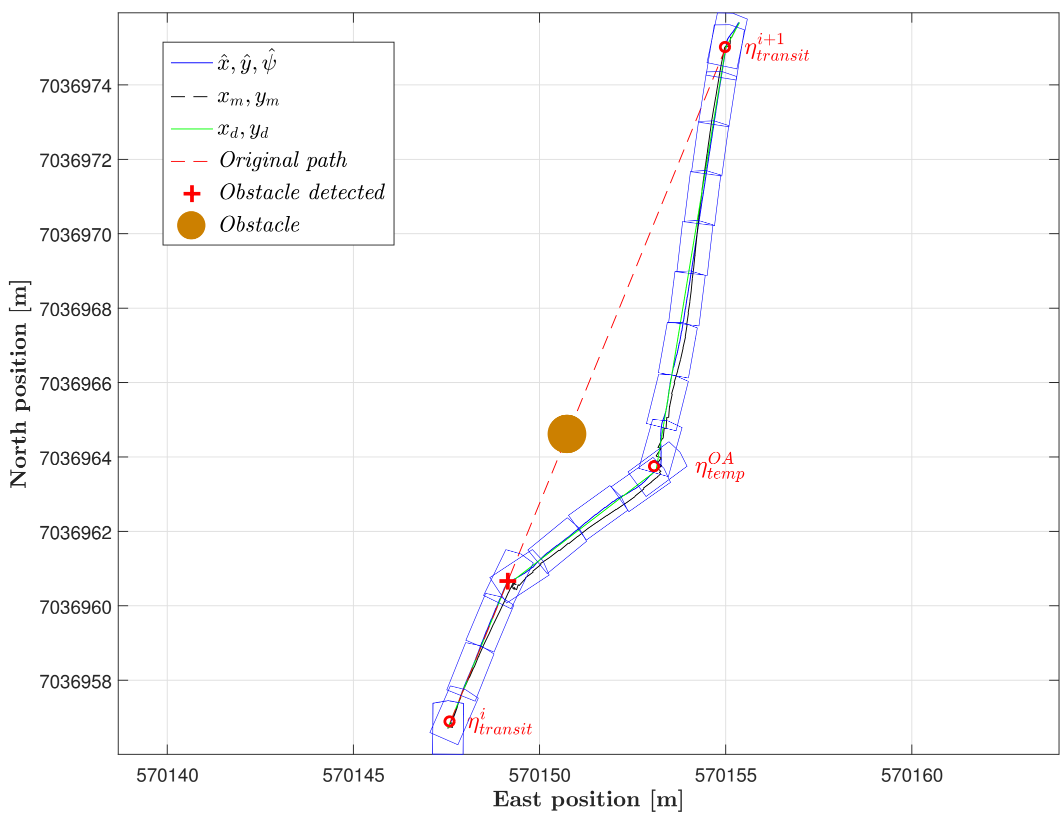
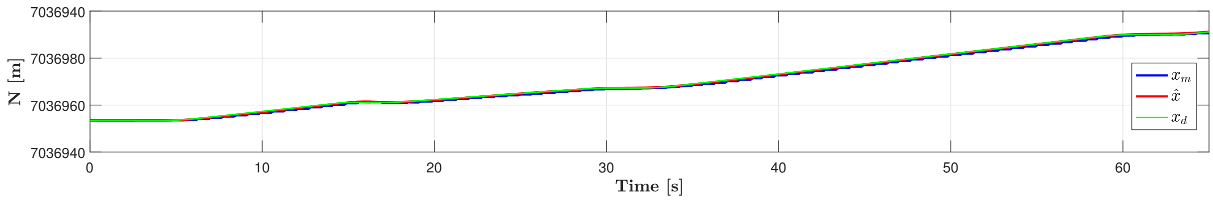
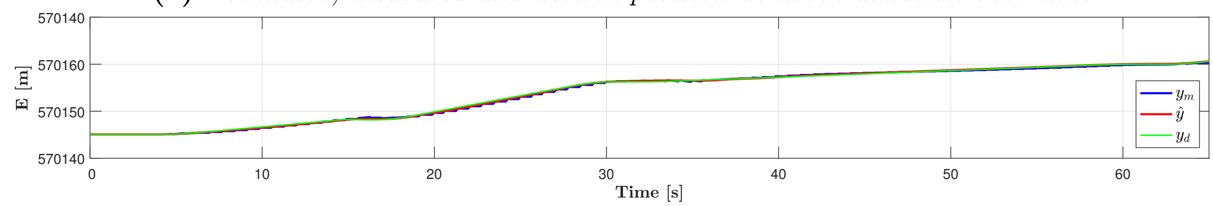


Figure 5.1: *Estimated, measured and desired ROV position in NE plane, $\beta = 30°$ and $d = 5$ m*

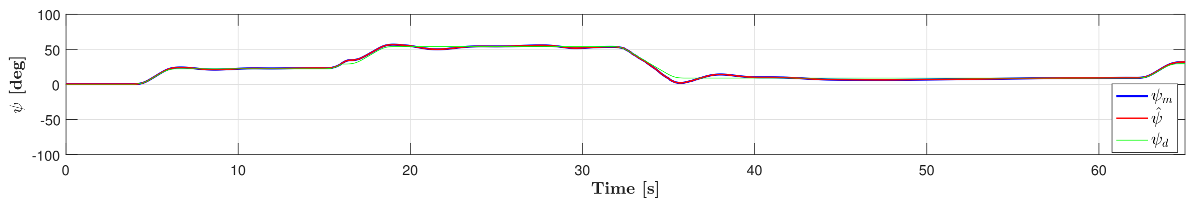
The vehicle is in the deliberative behavior Transit when an obstacle is manually detected. As a result, η^{OA}_{temp} is created as a temporary waypoint using the predefined parameters. The ROV trajectory is therefore modified from the original path to the avoiding path through this temporary waypoint. As soon as this waypoint is reached, the vehicle can return to the original Transit waypoint. As seen in the plot, the control system chooses the shortest path from the current position to the desired setpoint. The Transit waypoints are denoted by i and $i+1$ because it does not matter which waypoint it is, OA will be active through the entire mission and could be activated between any Transit waypoints. The state transitions can be further studied plotting the position and heading in time-domain, as presented in Figure 5.2.



(a) *Estimated, measured and desired position in north direction over time*



(b) *Estimated, measured and desired position in east direction over time*

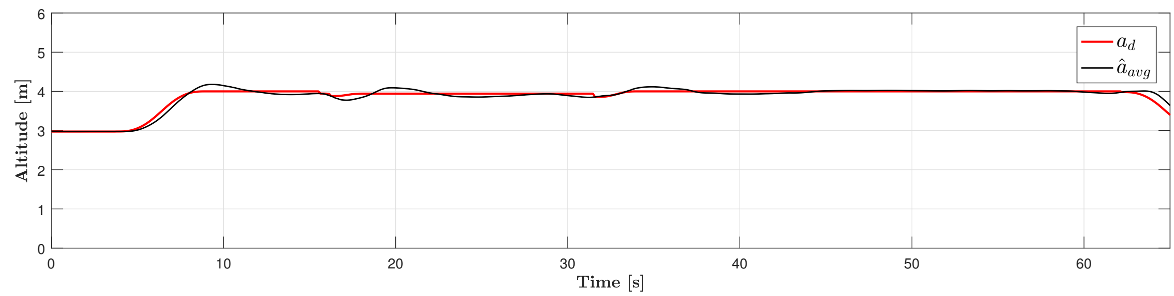


(c) *Estimated, measured and desired heading over time*

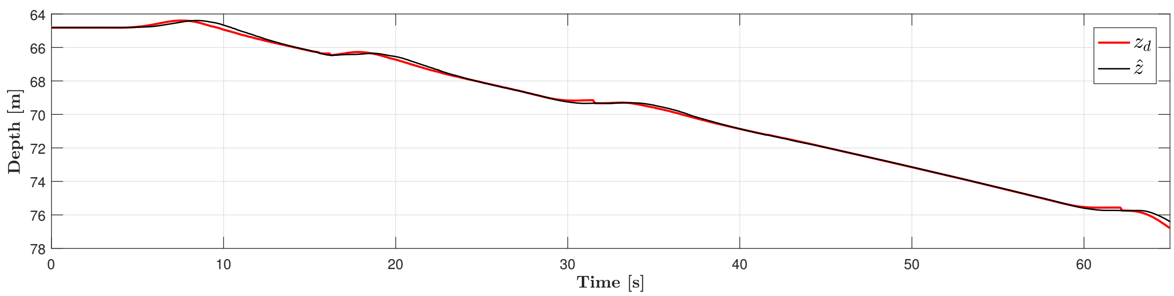
Figure 5.2: *Estimated, measured and desired ROV position and heading over time*

The vehicle is kept constant for the first 5 seconds in $\eta^i_{transit}$, which is the timer in Transit mode that requires the vehicle to stationkeep for 5 seconds as soon as it reaches a waypoint. After 5 seconds, the vehicle starts moving towards $\eta^{i+1}_{transit}$, which is seen from the increase in both position and heading. After approximately additional 5 seconds, the vehicle is kept constant in position, and the heading starts to increase due to the warning about an obstacle. The small constant level in the heading at time 17 seconds is due to the vehicle receiving information that the obstacle is no longer in front of the cameras. This is done by turning off the same switch as when detecting the obstacle.

Further, the vehicle turns until it reaches the desired heading defined by β, in this case, 30° in body reference frame. As soon as the vehicle has turned away from the obstacle, it starts moving towards the temporary waypoint and reaches this at time 30 seconds. After stabilizing here, the vehicle turns back to the Transit waypoint $\eta^i_{transit}$. Looking at Figure 5.2a and 5.2b, it can be seen that the position stabilizes, and thus the desired waypoint is reached at time ∼60 seconds. The altitude and depth during collision avoidance are plotted in Figure 5.3 as a function of time.



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure 5.3: *Estimated and desired ROV altitude and depth over time*

As can be seen in Figure 5.3a, the altitude is kept at 4 meters throughout the mission. The mission starts with an initial altitude of 3 meters defined as the desired altitude in $\eta^i_{transit}$. Further, the OA waypoint is defined using the same altitude as already desired, but it could easily be altered to desire a different altitude. The depth is increasing due to the seafloor, and the waypoint transitions are easier to see in Figure 5.3b, since the depth is constant when the vehicle is kept in a steady position. Overall, the simulation proves that the agent architecture can replan real-time while being on a predefined path. The vehicle reacts instantly when an obstacle is detected and continues to the desired setpoint as soon as the obstacle is avoided. The complete collision avoidance is illustrated in a 3D plot in Figure 5.4.

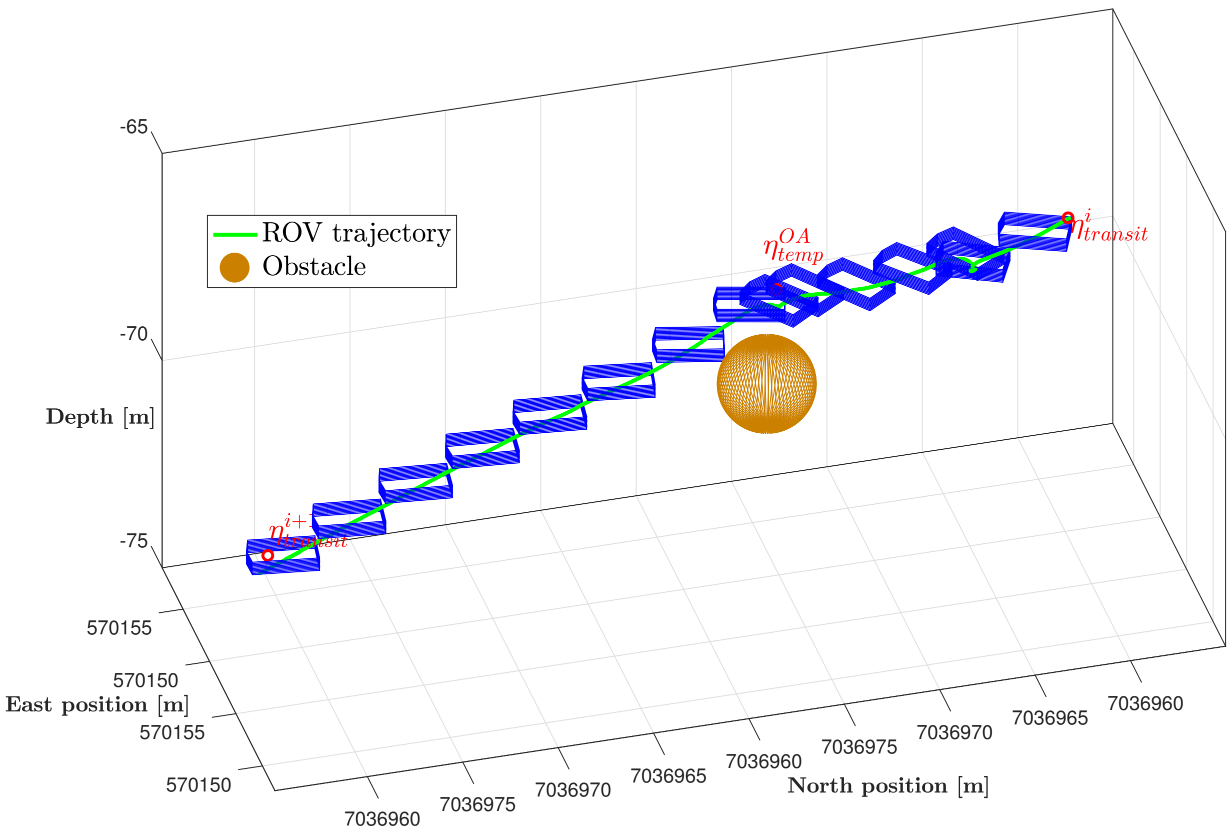


Figure 5.4: *3D illustration of the vehicle during mission control*

5.2 Deliberative control

The deliberate behaviors of the agent architecture were tested through several trials, both individually and sequentially. The deliberative behavior Transit was also tested in the previous trial, and the agent has the possibility of starting the mission in any of the states described in Sections 3.2-3.7. This section presents the entire autonomous mission from Launch to CT and Intervention. The mission is started close to the surface and takes the vehicle sequentially through all the deliberative states until CT, where the vehicle is stationkept in front of the imaginary SOI. All waypoints are predefined in the autonomy layer, and the mission is not interrupted between start and stop. Figure 5.5 shows an NE-plot of the vehicle during the simulation.

η_{init} is where the mission is started, and the vehicle is kept at this coordinate until it switches to state Transit. Due to the properties of the state Launch and Descent, the vehicle is kept constant to the north and east axes during these states. As a result, only the Transit

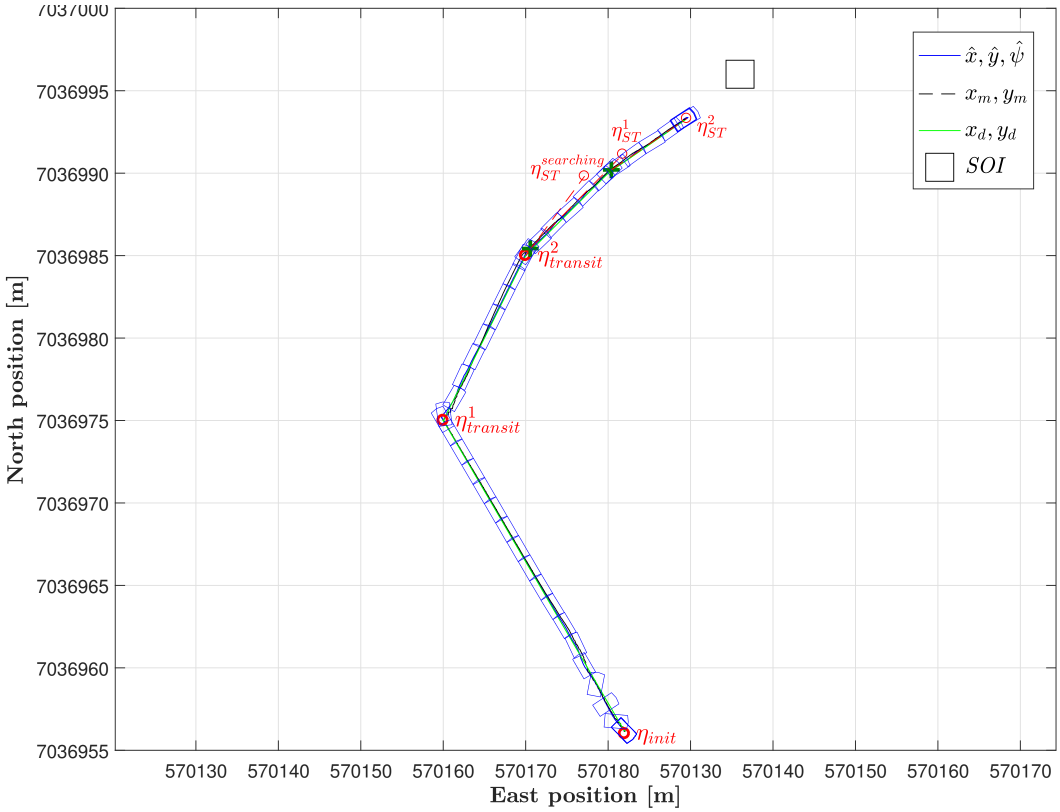
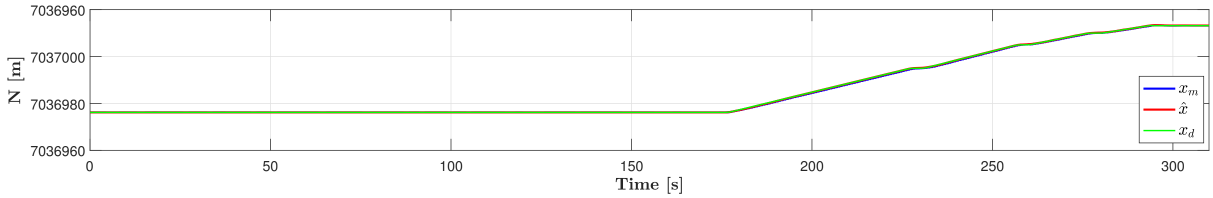


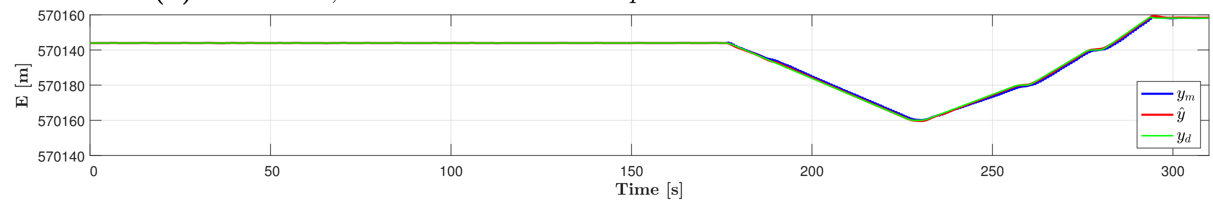
Figure 5.5: *Estimated, measured and desired ROV position in NE plane*

and ST states are visible in the Figure 5.5. Transit takes the vehicle through two waypoints, $\eta_{transit}^{1}$ and $\eta_{transit}^{2}$, before ST takes over control. Two simulated sonar updates are made after the sonar starts searching for the object, creating η_{ST}^{1} and η_{ST}^{2}. At the latter waypoint, the vehicle is located in front of the SOI with a distance of 3 meters. The autonomy layer then switches to stationkeeping in front of the SOI, as a merger of the two latter deliberative states CT and Intervention. The state transitions can be further studied in position and heading plots in Figure 5.6.

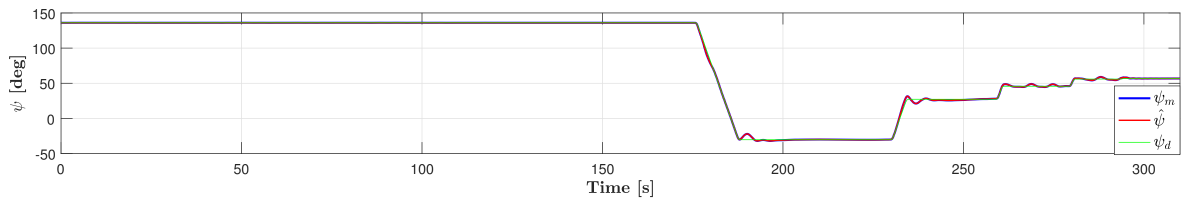
It is observed that the position and heading of the vehicle are kept constant for the first 175 seconds when the vehicle is sent through Launch and Descent. The transition between these states can only be seen when studying the ROV depth (Figure 5.7b). The transition between Descent and Transit, however, is clearly after 175 seconds, when the vehicle position and heading start to change. The vehicle reaches $\eta_{transit}^{1}$ after additionally 110 seconds, where the heading changes again (Figure 5.6c), and the vehicle starts moving towards $\eta_{transit}^{2}$. The transition between Transit and ST occurs at time ∼260 seconds, whereupon the sonar sends an update on the SOI position and the heading changes with additionally 20 °. The new sonar update is sent after additionally 20 seconds, and the heading is changed again, this



(a) *Estimated, measured and desired position in north direction over time*



(b) *Estimated, measured and desired position in east direction over time*



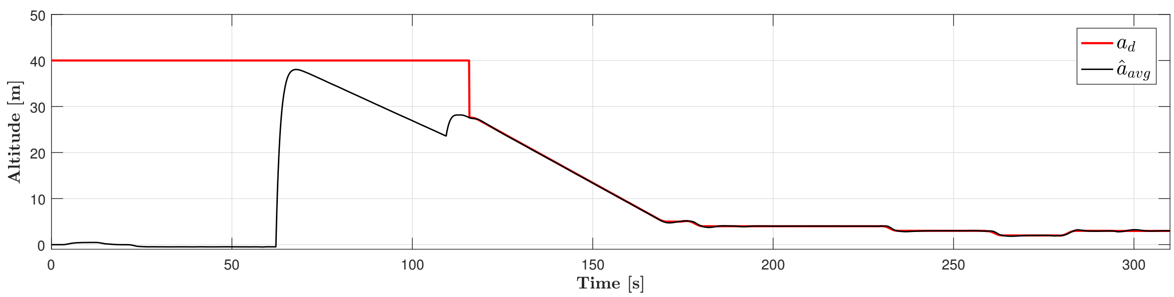
(c) *Estimated, measured and desired heading over time*

Figure 5.6: *Estimated, measured and desired ROV position and heading over time*

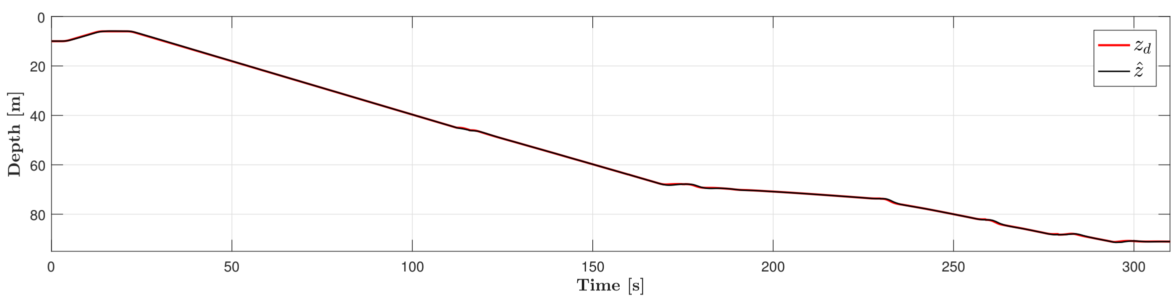
time with 10 °. The last ST waypoint is reached at almost 300 seconds, as can be seen from the positions Figures 5.6a and 5.6b, that stabilizes at this time. Some oscillations in the heading are observed as the vehicle reaches the desired heading in a new setpoint. The small oscillations in the heading result in small deviations in the east position (Figure 5.6b).
The state transition between Launch and Descent are visible in the depth plot of the vehicle (Figure 5.7b).

When the mission starts, the Launch state is activated. The vehicle is then taken to 5 meters depth to stabilize before descending towards the seabed. The depth plot shows that the mission is initialized after a couple of seconds, and reaches 5 meters after about 10 seconds. It stabilizes here for 10 seconds until the autonomy layer switches to state Descent. The remaining state transitions can be seen in both Figure 5.7a and 5.7b in the same time instants as discussed above for Figure 5.6. The mission from 60 meters depth until the end is illustrated through a 3D plot in Figure 5.8.

Similar simulations were tested using different predefined waypoints and parameters. Additional results are presented in Appendix D.



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure 5.7: *Estimated and desired ROV altitude and depth over time*

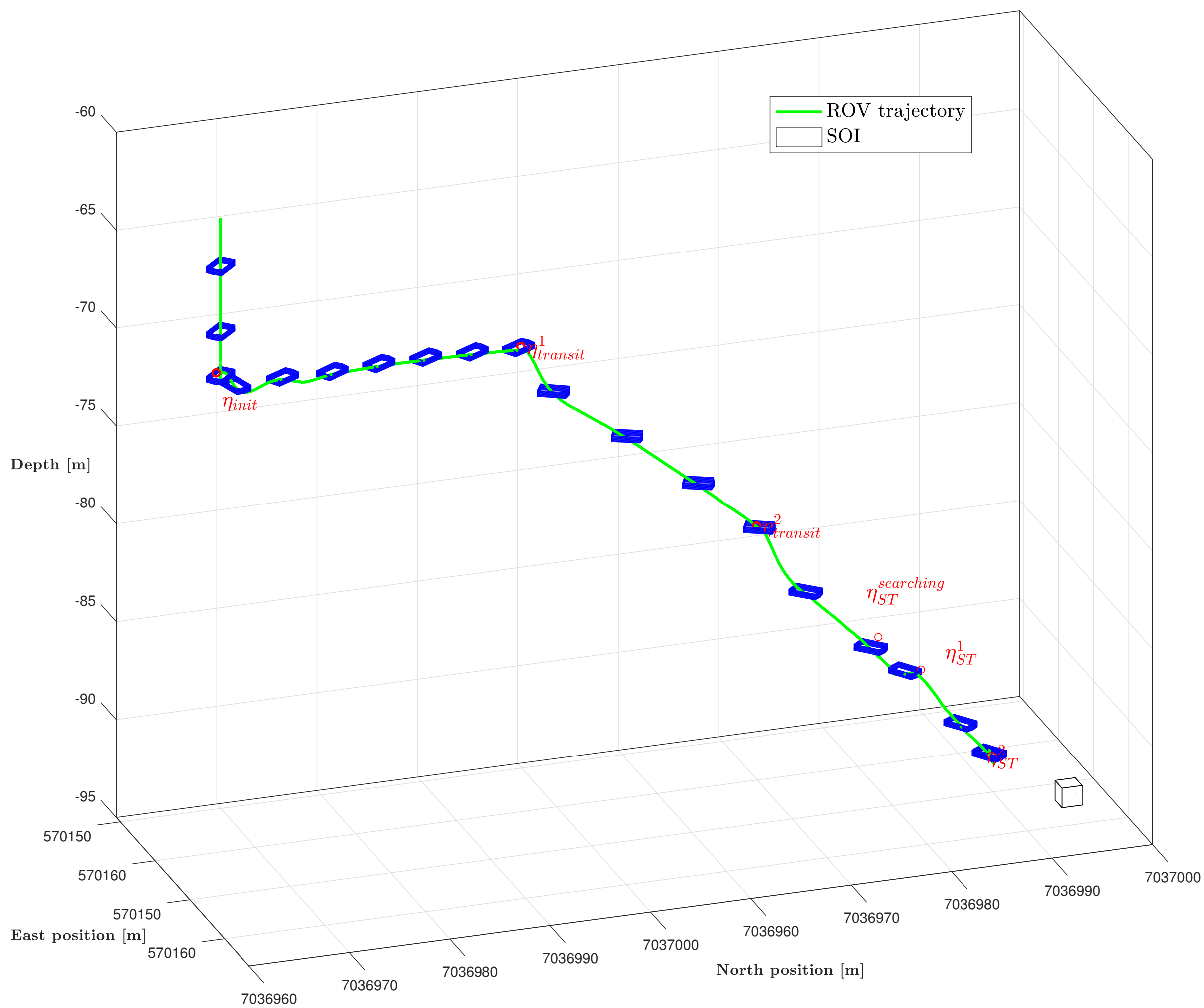


Figure 5.8: *3D illustration of the vehicle during mission control*

5.3 Hybrid control

The tests described in Section 5.1 and 5.2 are also performed using the hybrid agent architecture with deliberative and reactive behaviors in parallel. The vehicle is sent sequentially through the deliberative states to locate and approach an imagined SOI located on the sea bottom. While doing so, the vehicle avoided an obstacle on its path, through replanning the original path as a result of a manual obstacle warning. Figure 5.9 shows the vehicle in the NE plane during this trial.

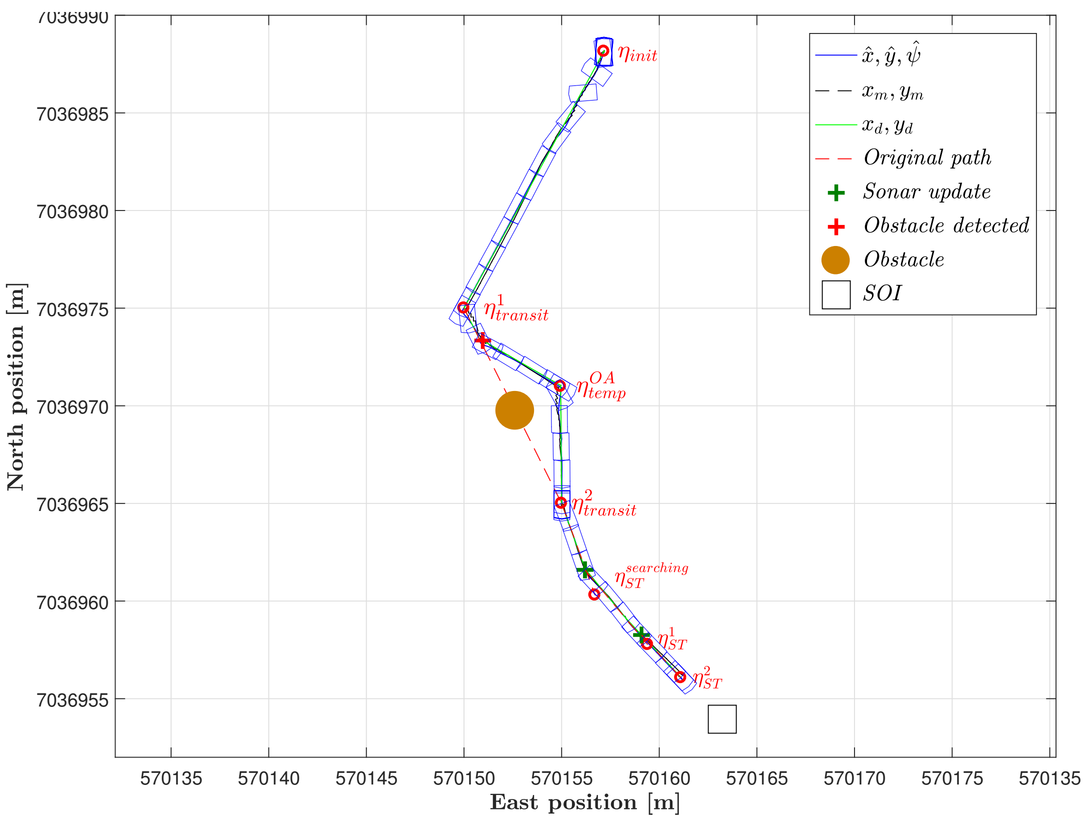


Figure 5.9: *Estimated, measured and desired ROV position in NE plane*

This time, the predefined waypoints are placed south of the initial waypoint, η_{init}, to test the system for different orientations. Reaching a certain altitude in Descent, the vehicle is sent towards the first Transit waypoint $\eta^1_{transit}$ using altitude control. Reaching this waypoint, the desired position is changed to the next Transit waypoint, $\eta^2_{transit}$. On the way towards this waypoint, the autonomy layer receives a warning about an obstacle in front of the vehicle. As a result, the vehicle turns $\beta = 30°$ to the left, and moves $d = 5$ meters ahead with this heading until it reaches the temporary waypoint η^{OA}_{temp}. Reaching this waypoint, the obstacle has been avoided, and the vehicle is safe to approach $\eta^2_{transit}$. This is the last Transit waypoint, and the autonomy layer thus switches to ST as soon as this waypoint is reached. A searching waypoint, $\eta_{ST}^{searching}$, is defined and the vehicle moves towards this while waiting for updates from the sonar data. The green crosses illustrate an updated sonar input to the autonomy layer, which results in an updated waypoint, η_{ST}^{1}. Further, as the vehicle approaches this new waypoint, it receives a new update from the sonar, which creates a corresponding waypoint, η_{ST}^{2}. This waypoint is within a distance of 3 meters from the SOI according to the sonar, and thus is no further update is necessary. Reaching the last waypoint, the vehicle switches to stationkeeping in front of the SOI, to facilitate for CT and Intervention. The parameters for the sonar waypoints are gathered in Table 5.1 and are relative to the ROV position when the update is received in the autonomy layer. The parameters are further described in Section 3.5.

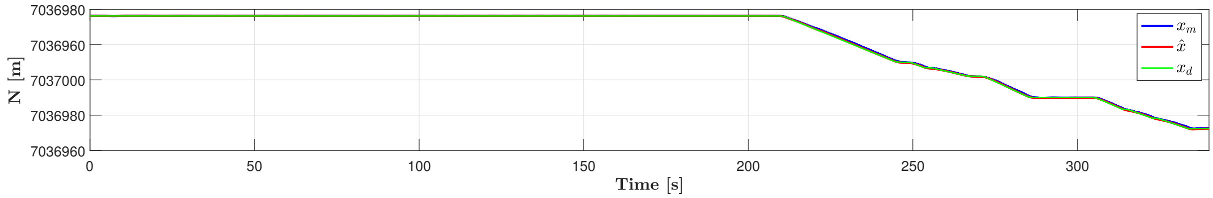
WP	α [°]	d [m]
$\eta_{ST}^{searching}$	-20	5
η_{ST}^{1}	-20	5
η_{ST}^{2}	-5	5

Table 5.1: *Parameters for ST waypoints relative to the ROV position when the sonar update is recieved in the autonomy layer*

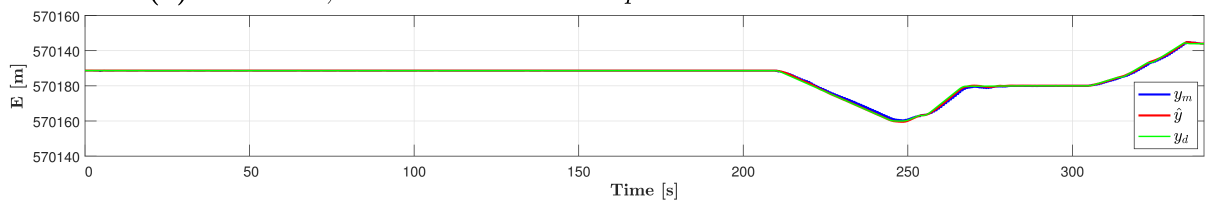
The mission is illustrated through position and heading plots in Figure 5.10, and depth and altitude plots in Figure 5.11.

Looking at the position plots for the first 220 seconds, it is clear that the vehicle is kept constant while descending towards the sea bottom. Figure 5.11b shows that the vehicle ascends to 5 meters depth during the Launch state. After stabilizing here, the vehicle starts descending continuously using depth control. When achieving BL (approximately at time ∼170 s), the autonomy layer switches to altitude control, and Figure 5.11a presents the input to the control system. When reaching an altitude of 5 meters, the vehicle stabilizes there before Transit is activated.

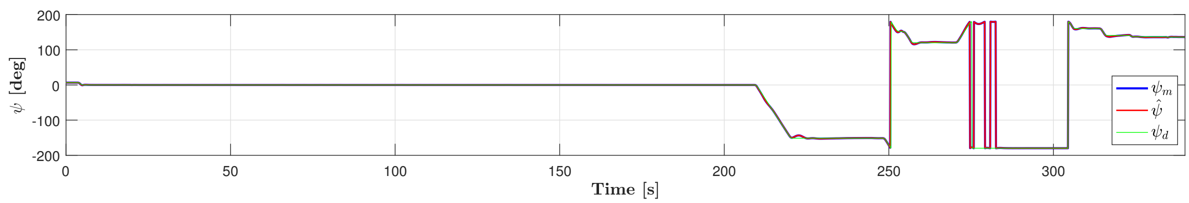
The vehicle starts approaching $\eta_{transit}^{1}$ after stabilizing at 5 meters altitude. In this case, $\eta_{transit}^{1}$ requires an altitude of 4 meters, and $\eta_{transit}^{2}$ requires an altitude of 3 meters. The waypoint transitions are thus easy to see when looking at the desired altitude in Figure 5.11a. Notice the jumps in the heading angle in Figure 5.10c after 250 seconds. These jumps are caused by the conversion in radians to the interval [-π π], which is done in a feedback control system to avoid discontinuities. The many jumps at time ∼270 seconds are due to the desired heading being 180°, and the estimated heading is slightly oscillating around the desired value. The ST transitions can be seen in the desired heading, at time ∼270, ∼310 and ∼320 seconds.



(a) *Estimated, measured and desired position in north direction over time*

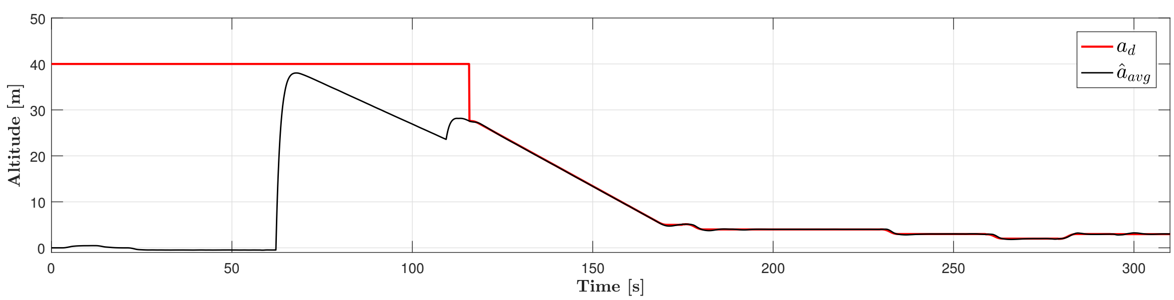


(b) *Estimated, measured and desired position in east direction over time*

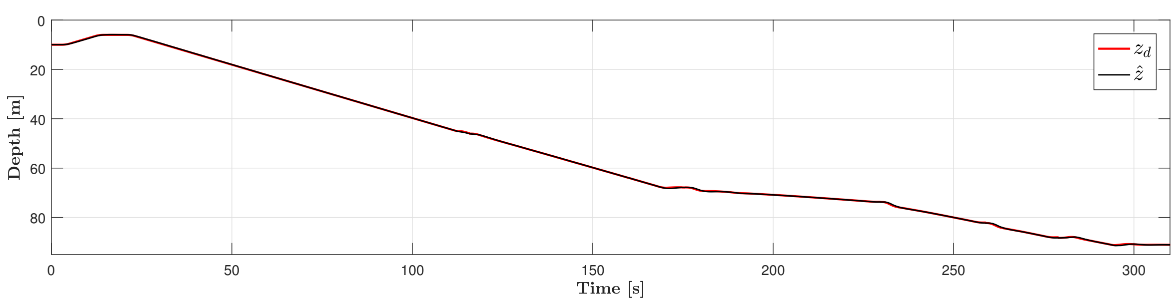


(c) *Estimated, measured and desired Heading over time*

Figure 5.10: *Estimated, measured and desired ROV position and heading over time*



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure 5.11: *Estimated and desired ROV altitude and depth over time*

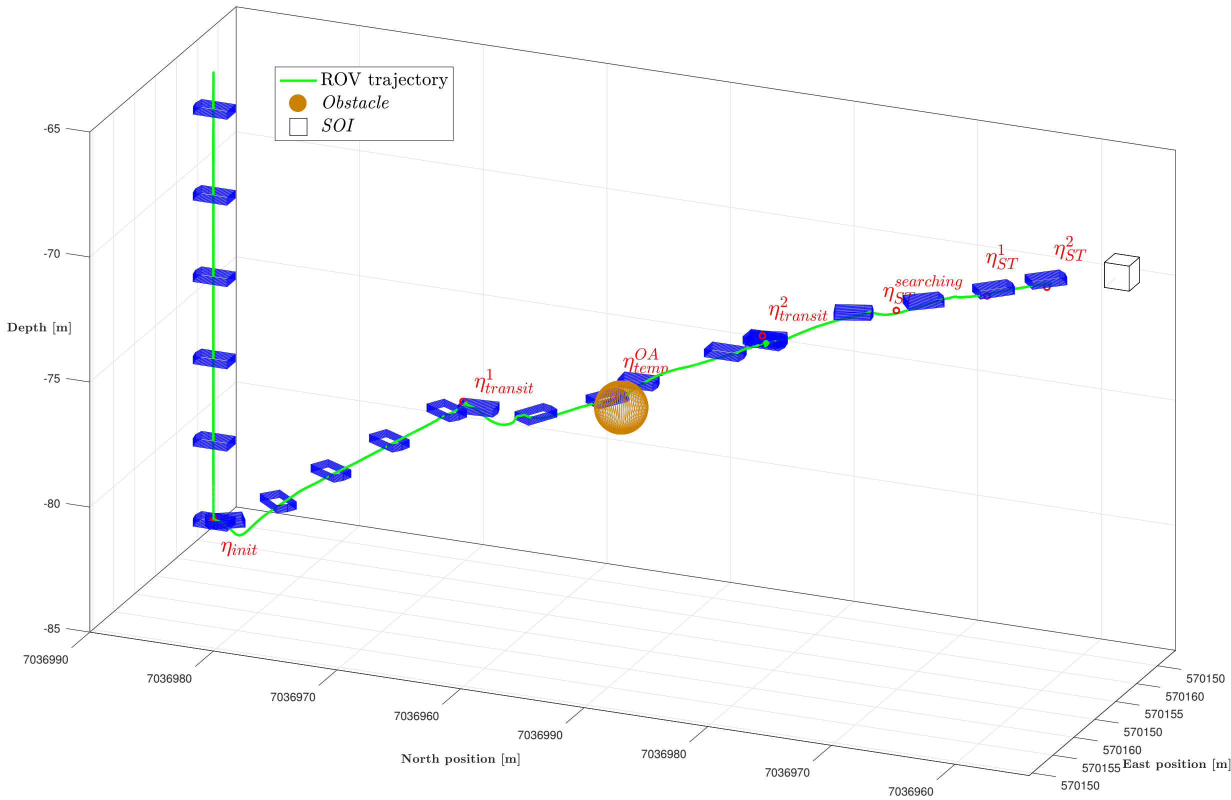


Figure 5.12: *3D illustration of the vehicle during mission control*

The mission from 65 meters depth until localization of the SOI is illustrated through a 3D plot in Figure 5.12.
To recap, the simulation trials have tested the deliberative, reactive and hybrid properties of the hybrid agent architecture. The results make a good foundation for testing the agent in full-scale experiments. The architecture is tested using equipment and set-up explained in Chapter 4, and the result is presented in Chapter 6.

Chapter 6

Full-Scale Experiments

The agent architecture was tested through real-time experiments in the Trondheim fjord using 30k using the experimental setup explained in Chapter 4. Raw data from the experiments were post-processed to obtain the following results. The architecture was tested through the same trials as in Chapter 5, thus, this chapter is divided into the same three sections.

6.1 Reactive Control

In this trial, the vehicle was sent towards the Herkules wreck on purpose to test the collision avoidance system. The vehicle was set to avoid obstacles with parameters $\beta = 35°$ and $d = 5$ m. The results are presented through the ROV position estimates ($\hat{x}$, $\hat{y}$, $\hat{\psi}$), measurements (x_m, y_m) and desired values (x_d, y_d) in Figure 6.1.

The waypoints are explained in Section 3.8. Notice that the size of the obstacle is not corresponding to the true size of Herkules, but it illustrates the position on the wreck that triggered the obstacle warning. In reality, the computer processing information from the stereo cameras can detect an object at a distance up to 5 meters.

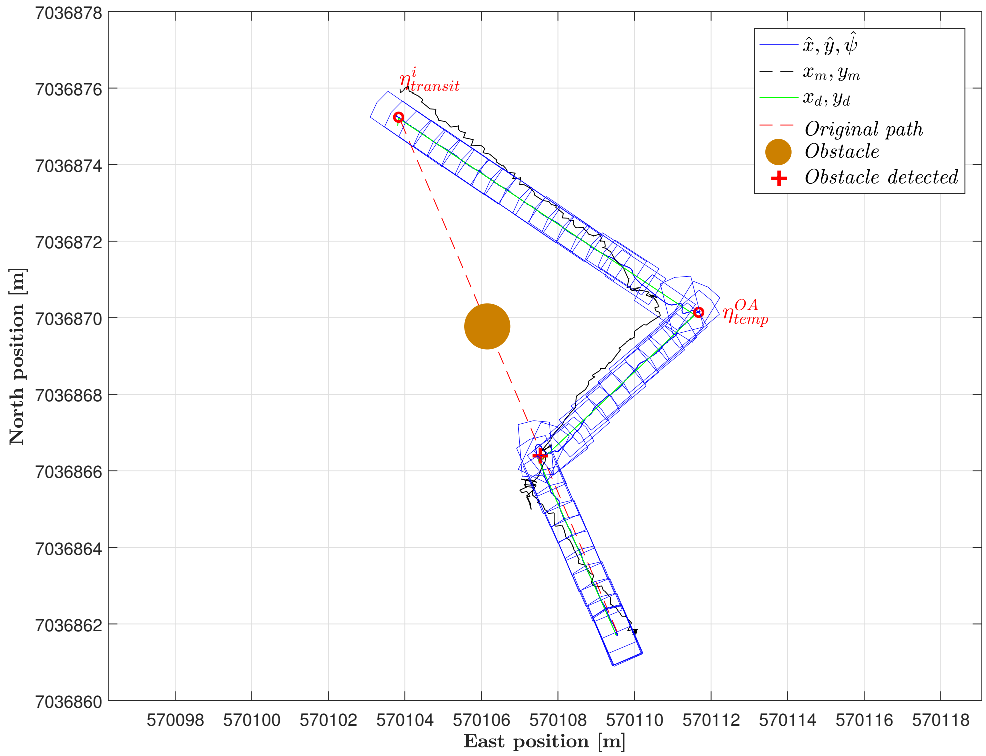
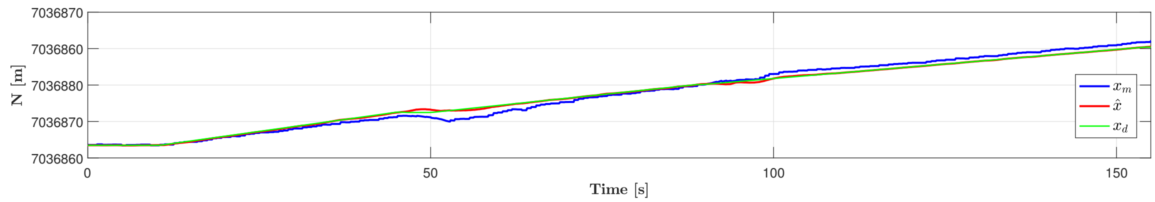


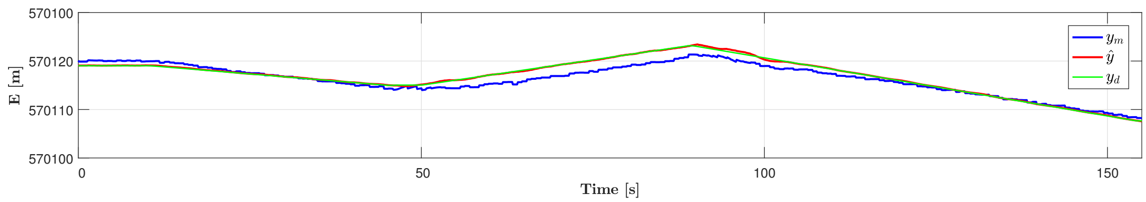
Figure 6.1: *ROV position in NE plane, $\beta = 30°$ and $d = 5$ m.*

To further invest the state transitions and behavior of the vehicle, the north and east position, as well as the heading angle, are plotted over time in Figure 6.2.

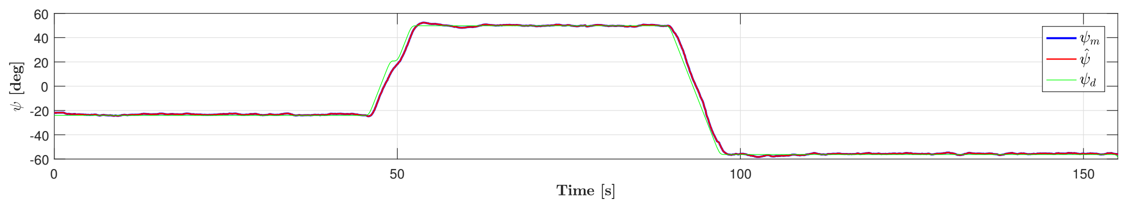
From the figure, it can be seen that the vehicle behaves as desired; at approximately 45 seconds, the autonomy layer receives a message about an obstacle in front of the ROV, left oriented. As a result, the vehicle stops moving in all directions and changes its heading with 35°. Reaching this heading, the ROV can still see the object, and the heading is therefore changed with additionally 35°. With this angle, the ROV can no longer see the object, and it is ready to approach the temporary waypoint η_{OA} at approximately 55 seconds. The vehicle starts moving in the north and east direction with a constant heading. Reaching η_{OA} and with no warning about other obstacles, the vehicle is safe to return to the initial transit waypoint at time 90 seconds. The heading is changed, so the ROV is looking in the forward moving direction, and is then kept constant until the waypoint is reached.



(a) *Estimated, measured and desired position in north direction over time*



(b) *Estimated, measured and desired position in east direction over time*



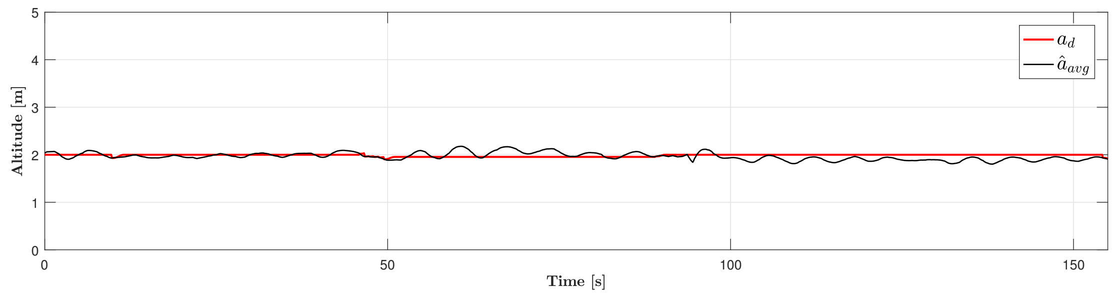
(c) *Estimated, measured and desired Heading over time*

Figure 6.2: *Estimated, measured and desired ROV position and heading over time*

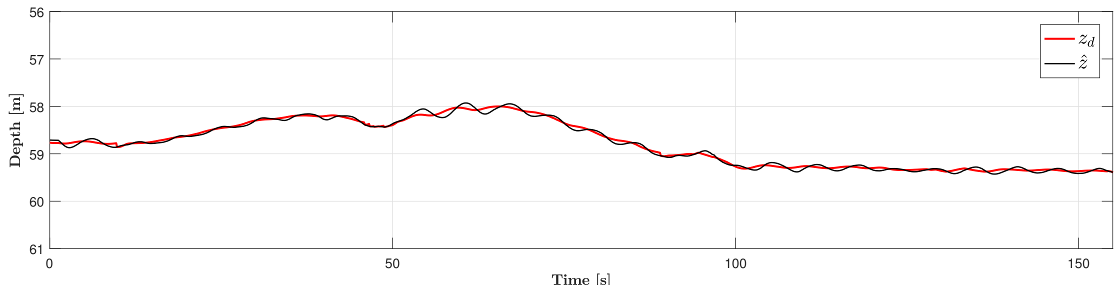
During the entire mission, the vehicle is set to have a constant altitude from the seafloor. Figure 6.3a shows the desired and estimated altitude and depth over time. As expected, the altitude is kept constant, and the depth is varying with the seafloor. The oscillations in the estimated values are caused by variations in the DVL signals and noise. The average value, however, shows that the desired altitude is kept constant throughout the mission.

The test is also illustrated through a 3D plot in Figure 6.4, where the obstacle is illustrated as a sphere. It is now easy to see how the vehicle performs during collision avoidance. Notice that the depth is varying with the seafloor, but the altitude is, as mentioned above, kept constant throughout this mission.

To recap, the collision avoidance has been tested, and all orders from the autonomy layer were upheld. The result was a reactive vehicle able to avoid collisions with unexpected objects in a quick manner, without predefining the situation. Collision avoidance was tested twice modifying only the heading angle when avoiding the obstacle. The results using an angle of 45 degrees is presented in Appendix E.1 using the same plots as above.



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure 6.3: *Estimated and desired ROV altitude and depth over time*

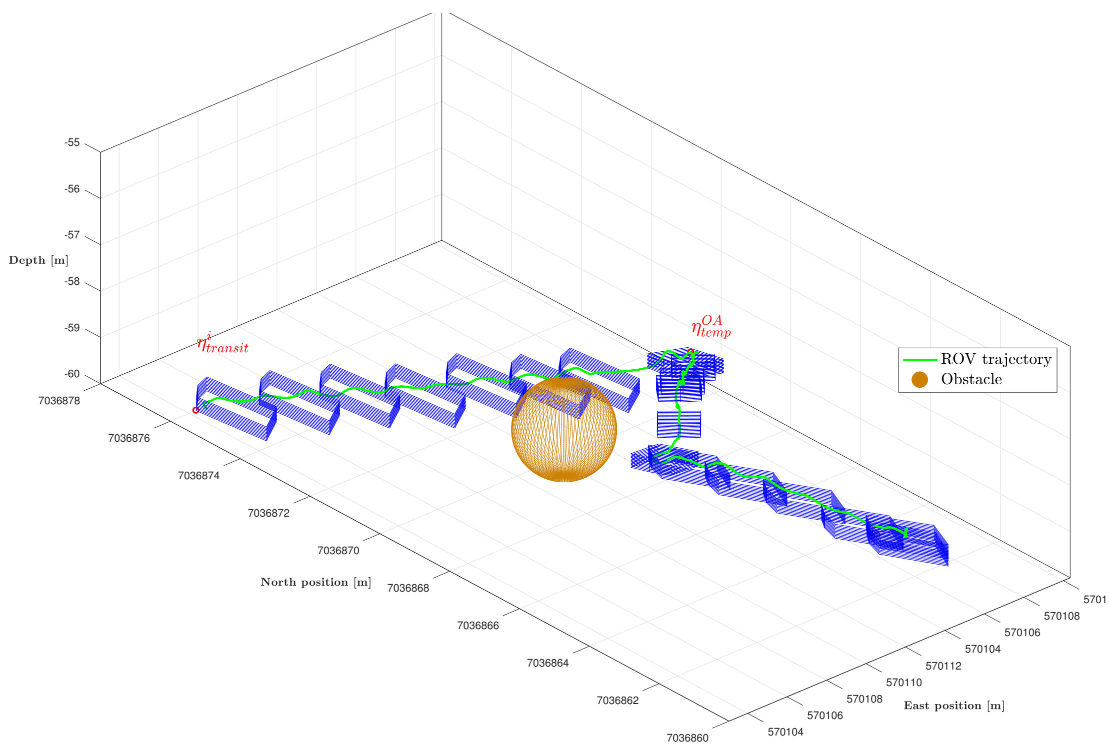


Figure 6.4: *3D illustration of the vehicle during mission control*

6.2 Deliberative control

This trial tested the deliberative behaviors implemented in the agent architecture using the Herkules wreck as a SOI. The Launch state was not included in this test because the ROV was already located in a position adequate for Descent. The ST state was tested separately to avoid collision with the wreck and entanglement of the cable. The mission was started in Descent at an 18 meters depth. Figure 6.5 shows the result of the autonomous mission in a NE plane of the vehicle, representing the position estimates ($\hat{x}$, $\hat{y}$, $\hat{\psi}$), measurements (x_m, y_m) and desired values (x_d, y_d).

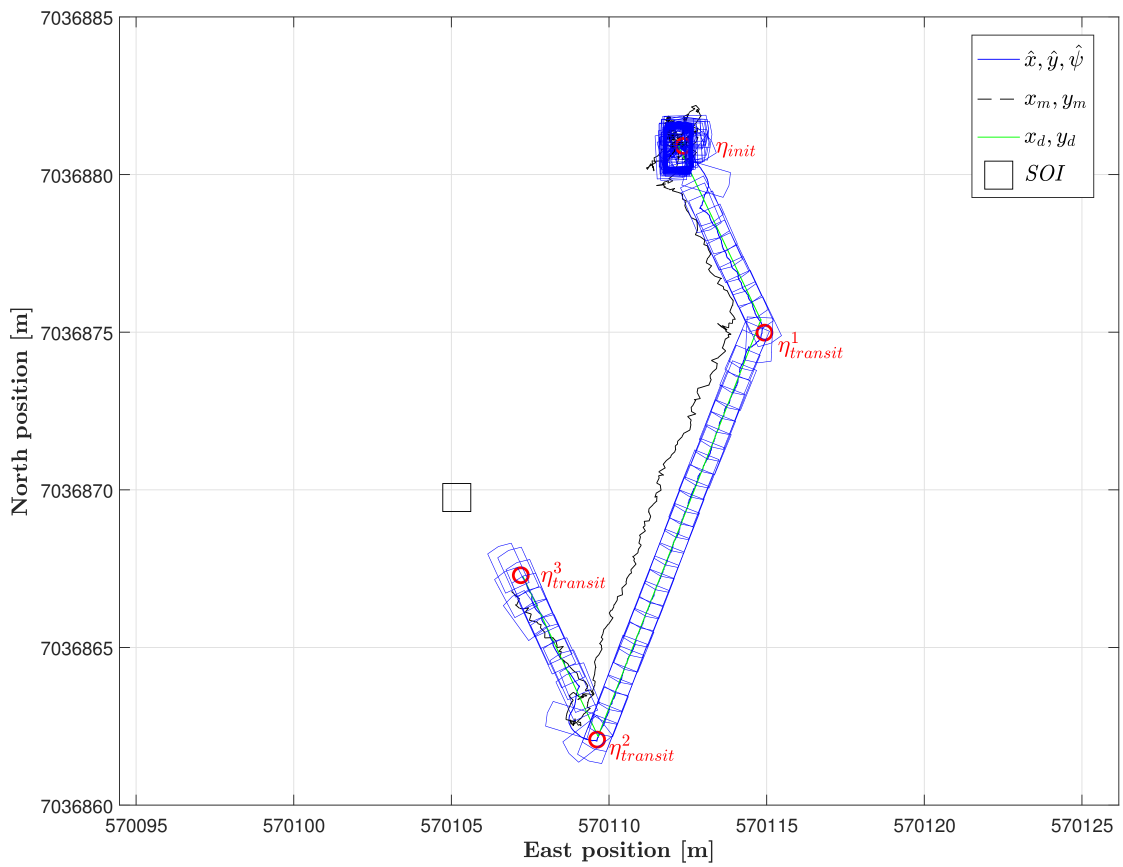
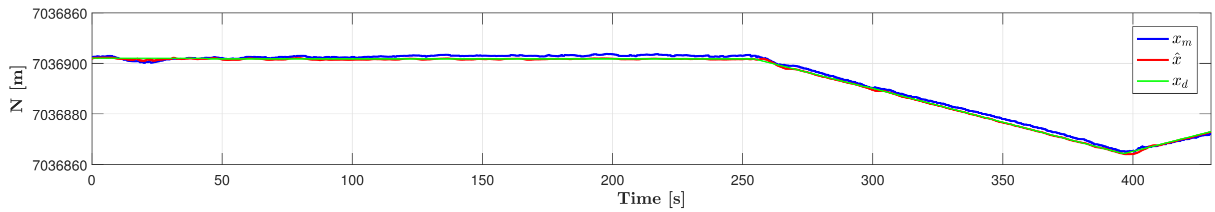


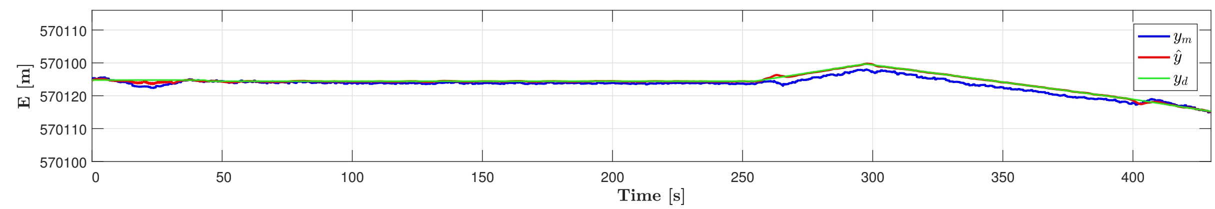
Figure 6.5: *Estimated, measured and desired ROV position in NE plane*

η_{init} represents the setpoint from the Descent-state. In this test, three transit waypoints are defined, $\eta^1_{transit}$, $\eta^2_{transit}$ and $\eta^3_{transit}$, respectively. The last waypoint is set sufficiently close to the SOI to perform ST.

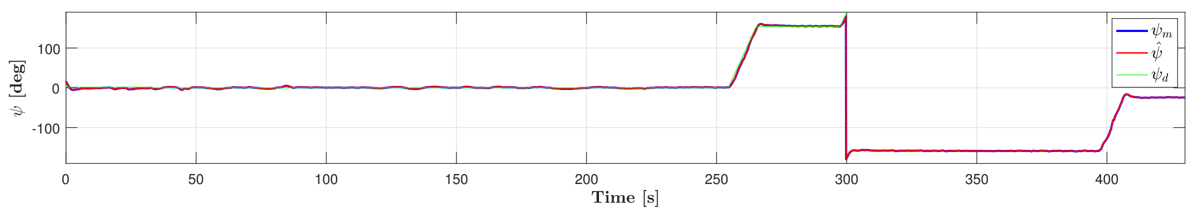
To further study the state and waypoint transitions, the position in north and east, as well as the heading of the ROV, are plotted over time in Figure 6.6. Notice the deviations between the measured and estimated positions in both Figure 6.5 and 6.6.



(a) *Estimated, measured and desired position in north direction over time*



(b) *Estimated, measured and desired position in east direction over time*

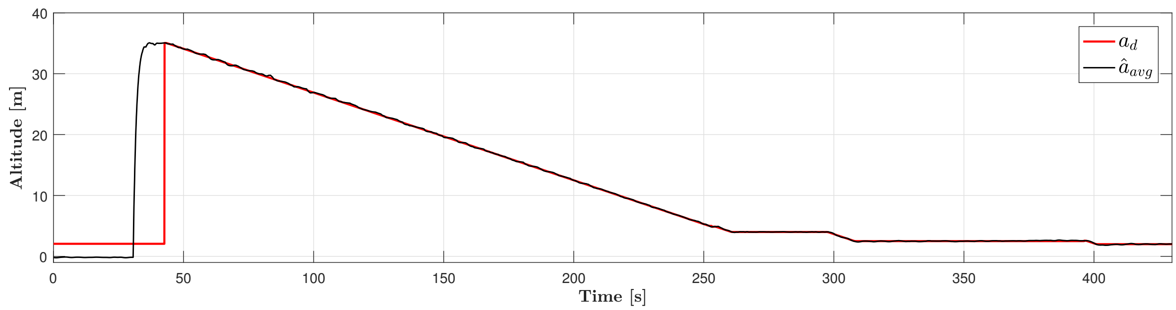


(c) *Estimated, measured and desired Heading over time*

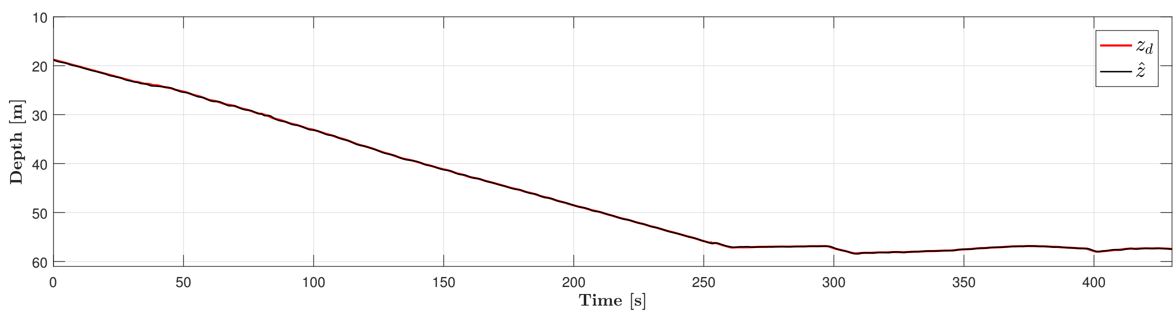
Figure 6.6: *Estimated, measured and desired ROV position and heading over time*

It is evident from the figure that the state transition from Descent to Transit occurs at approximately 255 seconds, due to the sudden change in all DOFs. The transitions between the transit waypoints happen at time 230 seconds, 300 seconds and 395 seconds, respectively, which is seen by the change in heading (Figure 6.6c). To study the transition between WL and BL in particular (during Descent), the altitude and depth are plotted over time in Figure 6.7.

Initially, the altitude is zero as long as the DVL sensor is in WL (Figure 6.7a), and the vehicle is controlled using the depth (Figure 6.7b). At approximately 30 seconds, the sensor starts receiving signals from the sea bottom and switches to BL. The estimated altitude experiences a sudden jump from zero to 35 meters. After 5 seconds, the desired altitude is set to 5 meters in the autonomy layer. A small transient effect is observed in the depth plot (Figure 6.7b) as the control system switches from depth to altitude control.
Further, the state and waypoint transitions explained above can also be seen in Figure 6.7, as the desired altitude is also changed at these specific time steps.
Finally, the entire autonomous mission is illustrated in a 3D plot in Figure 6.8, where the



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure 6.7: *Estimated and desired ROV altitude and depth over time*

same waypoints are plotted as in Figure 6.5. The plot shows the ROV behavior from 45 meters depth until it completes the mission in front of the shipwreck.

This section has presented results from testing of the deliberative states Descent and Transit. ST will be particularly studied in the following.

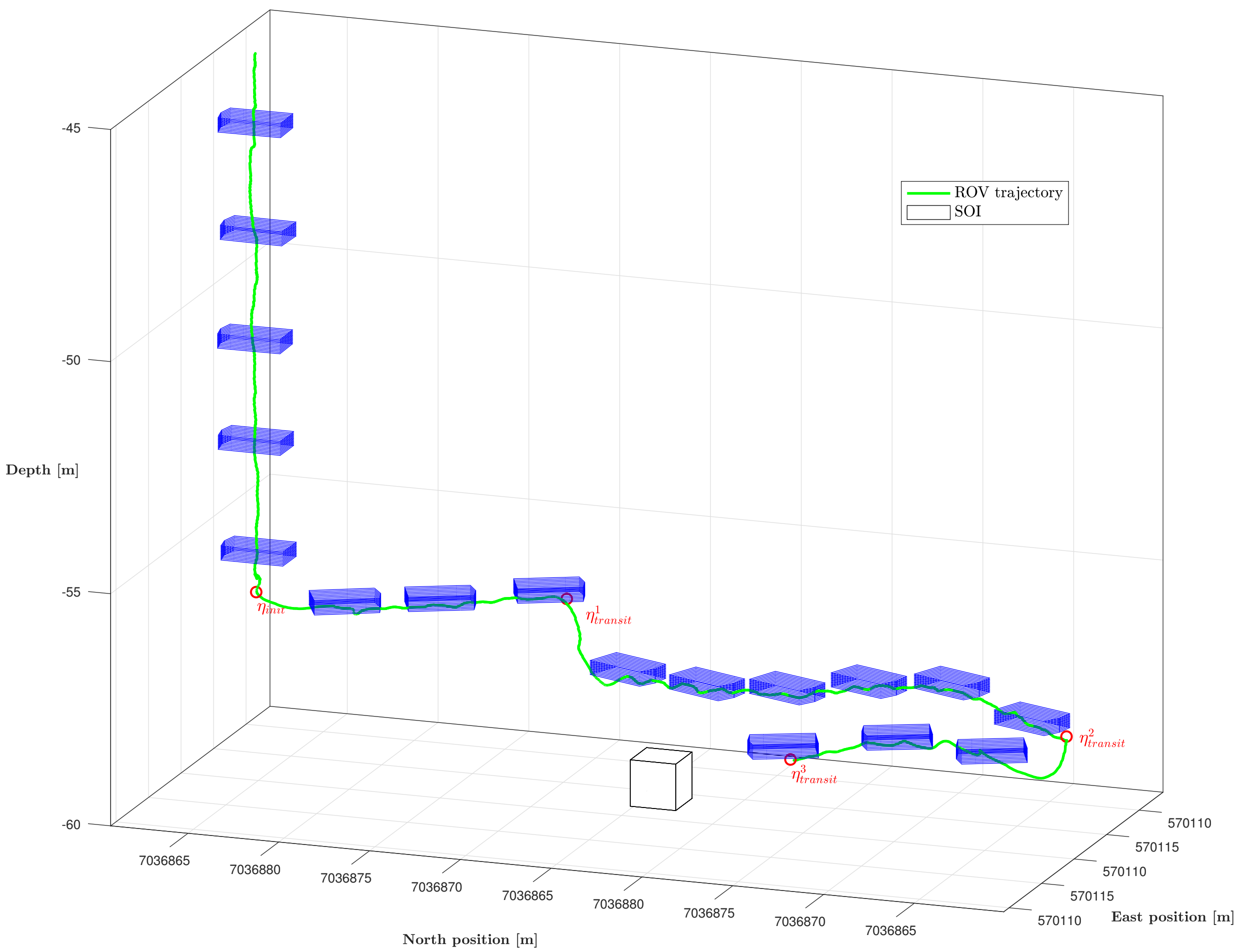


Figure 6.8: *3D illustration of the vehicle during mission control*

6.2.1 Sonar Tracking

ST was tested individually during the full-scale testing. Description of this state is given in Section 3.5, and it is tested with two manual sonar updates with $\alpha = 45°$ and $d = 5$ m. Position estimates ($\hat{x}$, $\hat{y}$, $\hat{\psi}$), measurements (x_m, y_m) and desired values (x_d, y_d) are plotted in Figure 6.9.

From the figure, it is clear that the vehicle can change the heading and ahead distance as soon as an update is registered, and finally, to approach the SOI. The result shows that the agent can trigger a change in behavior using manually simulated sonar feedback. The feedback can be updated an infinite amount of times, and the vehicle will change its heading as soon as an update is registered. When the vehicle reaches the SOI with a distance of more or less 2 meters, it is located in an adequate distance for CT.

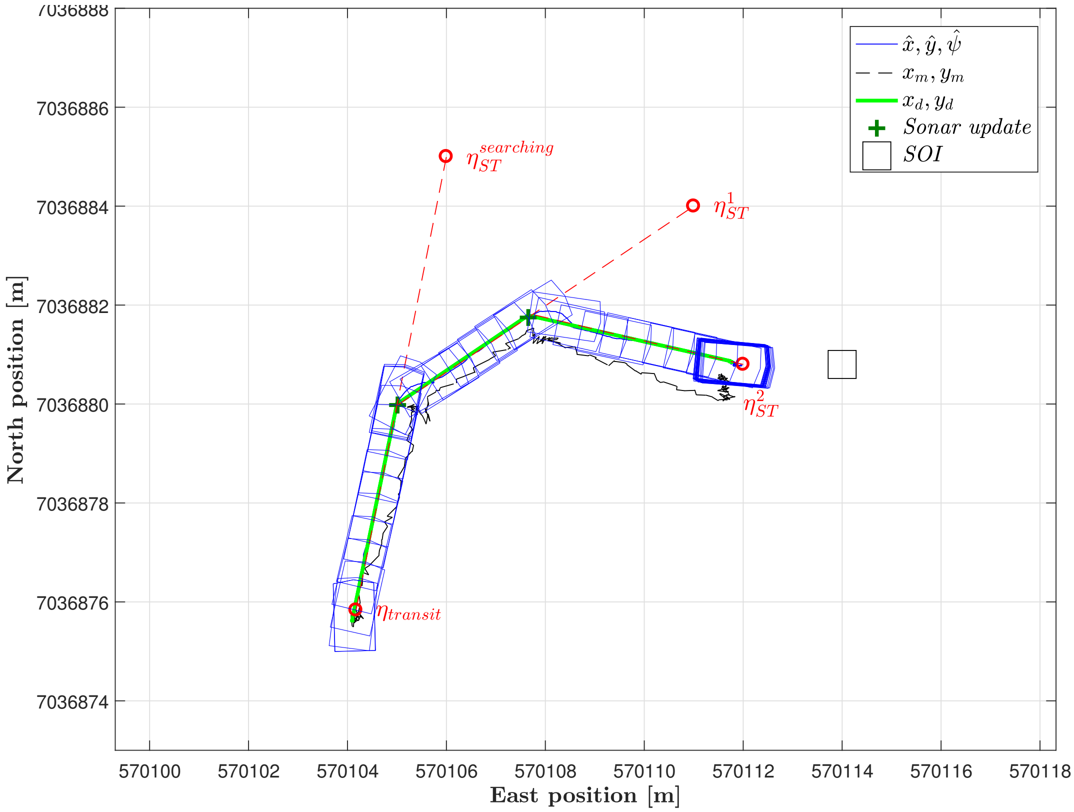


Figure 6.9: *Estimated, measured and desired ROV position in NE plane*

6.3 Hybrid control

Hybrid control is tested using the Herkules wreck as an obstacle, and the SOI is simply envisioned by a predefined coordinate. This trial tests all the deliberative and reactive behaviors of the agent architecture, from Launch to CT/Intervention, avoiding an obstacle on the path. The trial also proves the opportunity of modifying the Transit waypoints on the fly, since $\eta_{transit}^{3,original}$ turned out to be located too close to the obstacle. As a result, this waypoint is modified during the mission, and the vehicle approaches $\eta_{transit}^{3,modified}$ instead. The result is presented as a NE plane in Figure 6.10.

It is observed that the vehicle can behave as desired from the hybrid agent; sequentially performing the deliberative states, being able to replan on the fly when obstacles appear. The north and east position over time are plotted in Figure 6.11a and 6.11b, and the heading in Figure 6.11c.

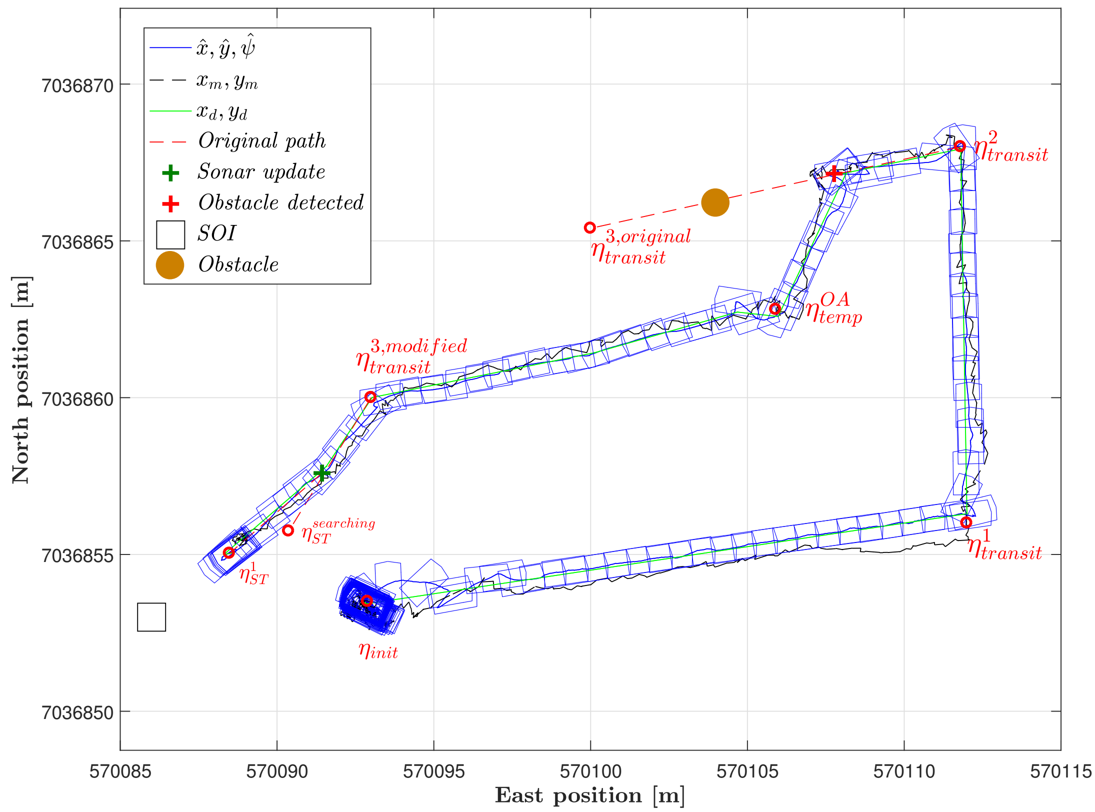
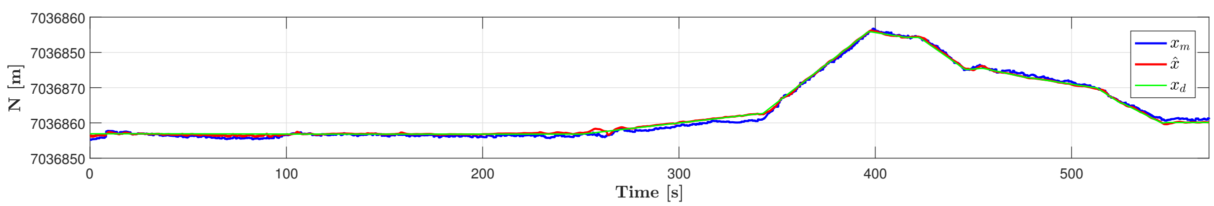
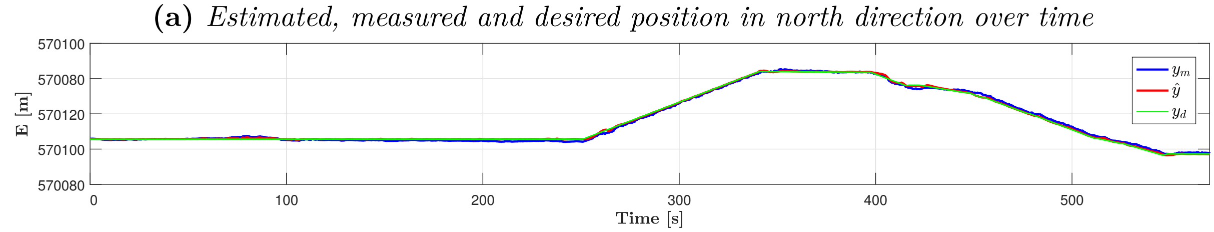


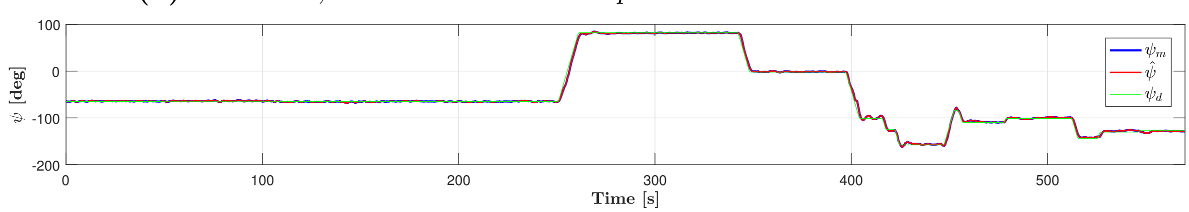
Figure 6.10: *Estimated, measured and desired ROV position in NE plane*



(a) *Estimated, measured and desired position in north direction over time*

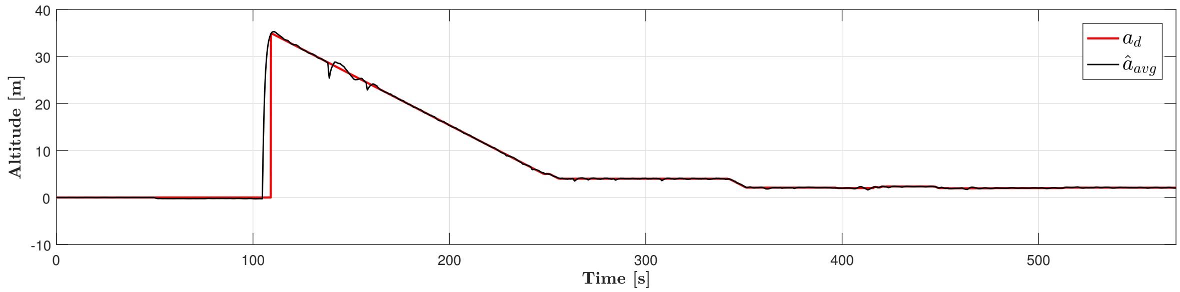


(b) *Estimated, measured and desired position in east direction over time*

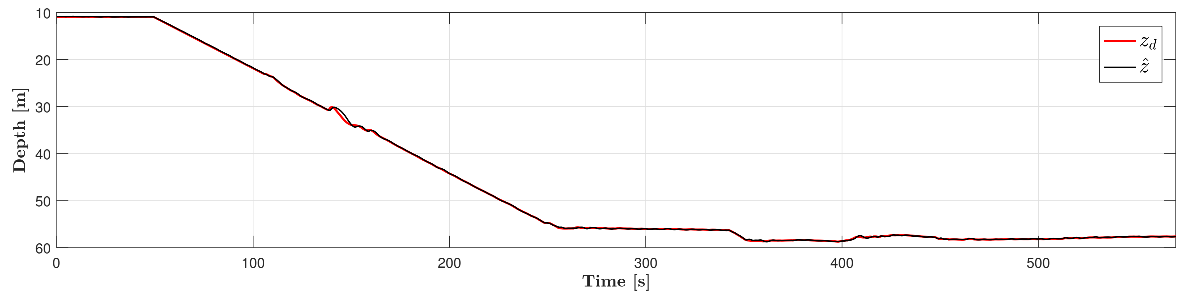


(c) *Estimated, measured and desired Heading over time*

Figure 6.11: *Estimated, measured and desired ROV position and heading over time*



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure 6.12: *Estimated and desired ROV altitude and depth over time*

The state transitions can be seen in the same way as previous trials, as the heading suddenly changes. The altitude and depth are plotted in Figure 6.12, where the same transitions are visible. Notice the bump in the altitude and depth as the vehicle switches from depth to altitude control after 150 seconds. The constant depth in the start is when the vehicle is kept in Launch at 10 meters depth before the autonomy layer switches to Descent. The mission is plotted in a 3D illustration in Figure 6.13 from a depth of 40 meters.

From all the presented results it can be observed that the hybrid control trial has been successful, taking the vehicle from the surface to the SOI through the implemented states, avoiding obstacles on the way. Additional results from the experimental testing is given in Appendix E, and the digital Appendix A.2 is a video from the onboard ROV camera when avoiding collision with the transponder tower.

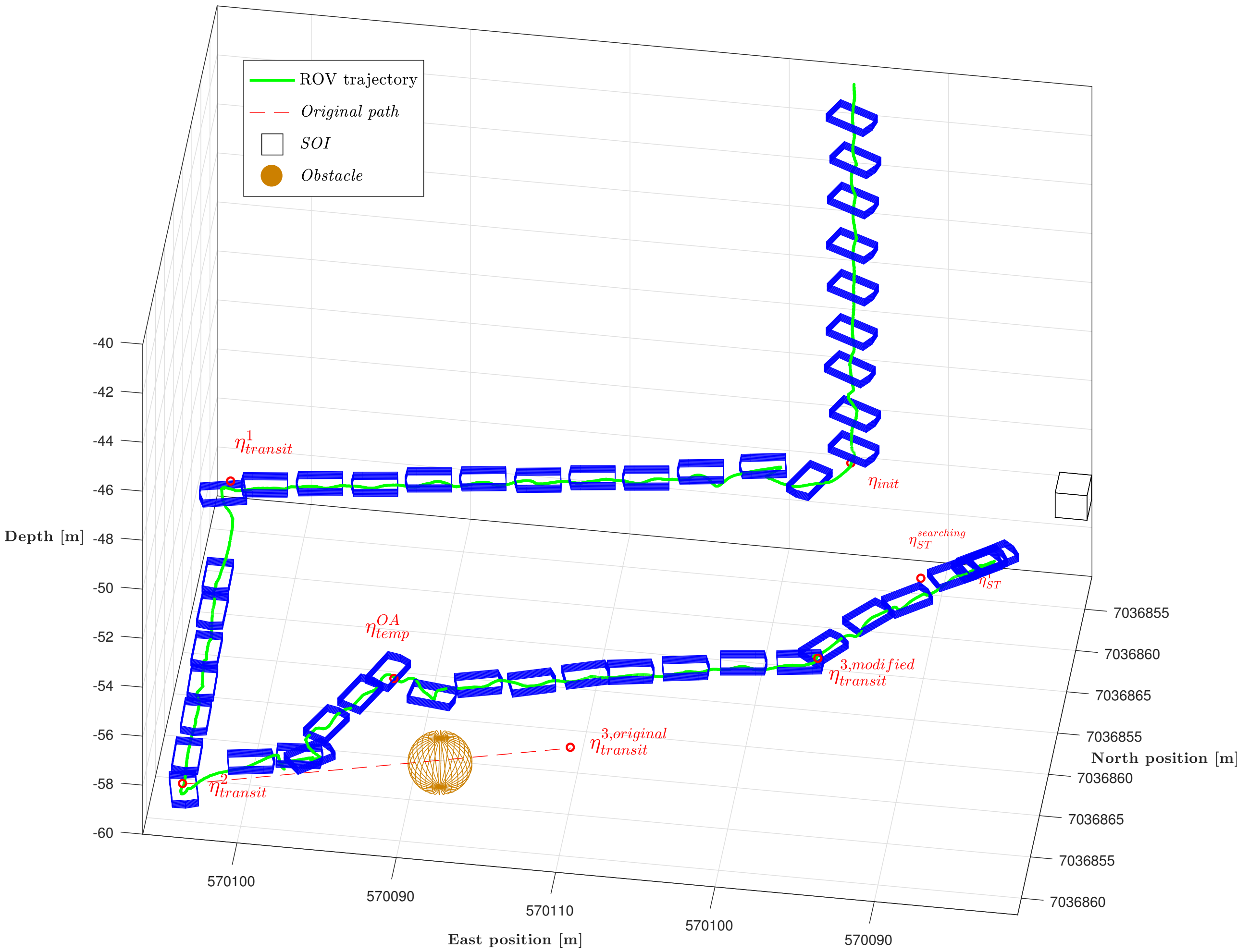


Figure 6.13: *3D illustration of the vehicle during mission control*

Chapter 7

Discussion

To recap, both simulation and full-scale results proved satisfactory behavior of the agent architecture. All deliberative and reactive states have been tested, and the behavior of the vehicle corresponds to the desired behaviors in the different states. The result is discussed in terms of deliberative, reactive and hybrid control in Sections 7.1-7.3. Lastly, uncertainties in the results are discussed in Section 7.4.

7.1 Deliberate control

Results show that the agent can control the vehicle from surface until approach and localization of a SOI. The agent can start the mission in any of the deliberative states, which makes the mission control system flexible and applicable to both complete missions and specific tasks. An example is if the vehicle is already located close to the seafloor, and the agent can start the mission in Transit instead of taking the ROV to the surface before starting the mission. Another advantage with the agent is that it uses the most suitable sensors for navigation dependent on the active deliberative state. When the DVL sensor is not in BL, the vehicle is controlled using depth control, and when achieving BL, altitude control is activated. The sonar is not actively used before ST is active, as is the case for the stereo cameras and CT. The agent is therefore not requiring all sensor information available in every state, but rather the opposite. The sequential steering and division into sub-behaviors focus on navigation using only one sensor at a time, in addition to those sensors necessary for estimating the vehicle translations and rotations.

Some transient oscillations are observed in the altitude and depth when the vehicle is in Descent, and the agent switches from depth to altitude control. It is believed that these transients occur due to the assumption of zero velocities in the altitude controller whenever it is enabled. Since the vehicle is moving in depth, and hence the velocities are nonzero, this assumption creates a wrong initial condition for the controller. As a result, the control output oscillates slightly, but after receiving measurements of the altitude, it can stabilize and control the vehicle again. This could be improved by using the current velocities as initial conditions in the altitude controller instead of assuming zero velocities. However, since this would require configurations in the low-level control system, it was not further considered for the scope of this thesis.

Some oscillations are observed in the heading angle during the waypoint transitions in Transit and ST. These oscillations are caused by the desired change in heading and position at the same time. The vehicle is trying to move in another direction than the direction to which it is headed, due to the sudden change in the desired waypoint. As a result, all the thrusters are working to control 4 DOF's simultaneously, and the heading oscillates slightly before stabilizing at the desired value. These oscillations could be removed by requiring a change in heading before the vehicle should move in north and east position. This way, one would assure the vehicle is directed towards the new waypoint before moving. The control purpose would be divided into two: first controlling the heading to the desired value, and then controlling the north and east position to the desired value, keeping the heading constant. This improves the performance of the control system and the vehicle and is proposed for further work.

Overall, the deliberative testing has been successful, and each state has been tested thoroughly through both simulations and full-scale experiments.

7.2 Reactive control

The reactive properties of the agent architecture were tested separately during the state Transit. The results show that the vehicle can replan the original mission when either a manual or a real-time stereo camera detection is triggered. The agent has the opportunity of adding other reactive behaviors, and the implementation of collision avoidance shows that the control execution layer can prioritize the reactive behaviors as soon as they are triggered. It is clear from the results that as soon as the *SV module* detects an object, the vehicle starts turning. This implies that the functional reactive layer is in fact constantly waiting for sensor information, as opposed to the deliberative layer that is triggered by state transitions. Collision avoidance would not be active unless som manual or sensory information triggered the state.

The collision avoidance is rather simplified, but knowing that the agent can react and replan the mission on the fly, more complex avoidance algorithms could be introduced. An advantage with the OA state is that it has the opportunity of turning off information from the *SV module*. This turned out to be beneficial when the *SV module* was not working properly. For instance, the module had not implemented that fish were not obstacles, which resulted in obstacle warnings every time a fish appeared close to the stereo cameras. By manually turning off the information from the *SV module*, the vehicle could proceed the mission without being interrupted. With more post-processing of the stereo images, filtering of uncritical objects would be possible, so such a switch would not be required. In addition, to reach level 3 of autonomy (Section 1.1.6), the vehicle itself should be able to decide which obstacles are threatening and not. Nevertheless, this manual switch turned out to be useful when testing with a not fully developed *SV module*.

A disadvantage with the OA state is that many parameters had to be set manually, such as the heading and ahead distance when avoiding the obstacle. When using other avoiding algorithms, smaller steps could be used, and several setpoints could be set while constantly listening to information from the stereo cameras. The need for predefined parameters is thus highly dependent on the method for avoiding the obstacle. Since this thesis does not study the different methods in detail, it has not been considered a significant disadvantage that these two parameters had to be predefined. In addition, it is reasonable to believe that a *SV module* could also provide the distance to and relative angle to the object if more post-processing of the images were performed. As a result, the OA state is considered quite successful, considering the constraints are in modules that are not included in the scope of this thesis. It is believed that with further research regarding these modules, it is believed that the vehicle will be able to perform quite accurate and optimal collision avoidance.

Overall, the results imply a hybrid agent able to replan the mission real-time when receiving information about an obstacle. The agent opens for implementation of additional reactive behaviors, and the results prove that the reactive layer will be activated with correct sensor input.

7.3 Hybrid control

Considering both simulation and full-scale trials on deliberative and reactive control separately showed favorable results, it is not surprising that the hybrid control was successful. The result is an agent architecture able to bring the vehicle from surface to a SOI while avoiding obstacles detected in the stereo images. The aspect discussed in Section 7.1 and 7.2 does obviously apply for the hybrid control since it is comprised of the deliberative and reactive control. The vehicle can perform the mission starting from any initial coordinate in the surface, as long as there is an available cable. The vehicle may pass through infinitely many predefined waypoints, and the agent is also open for modification of these predefined waypoints after mission start. Hybrid control does not have any restrictions of waypoints, which makes the cable the only spatial restriction for the ROV. Nor is there any restriction in the amount of obstacles the vehicle should avoid on the way, as long as an obstacle is detected, the agent will avoid it. The mission can be started in any deliberative state, making the *autonomy module* flexible with respect to the initial position of the vehicle.

7.4 Uncertainties

Some uncertainties have influenced the result of this thesis, not unexpected when using an experimental approach. Some of the uncertainties are mentioned in Section 7.1 and 7.2. In addition, some general uncertainties were observed in the results. The simulator is quite realistic, but some aspects are not predictable and possible to implement even in a high-fidelity mathematical model. These aspects are described in the following.

In the full-scale experiments it is observed that the position estimates has quite large deviations from the desired measured position. These deviations are considerably smaller when the mission is tested using the transponder tower as an obstacle (Appendix E.3) than when the Herkules wreck is used. Based on the latter observation, it is believed that the deviations are caused by the Herkules wreck. The deviation tends to increase as the vehicle changes the heading, and the measured position tends to deviate as the vehicle comes closer to the shipwreck. Therefore, it is believed that the wreck, as a large steel structure, causes disturbances in the ROV compass, and the measurements will not be true. As a result, the observer in the control system has a nonzero estimation error, and the vehicle is controlled to a position deviating from the true desired position. Another possible reason for the deviation is that the position of the assisting ship, Gunnerus, might have a small deviation from the initial position when the control system was started. Being directly proportional to the distance between the ship and the vehicle (see Section 4.3.1), the vehicle position may experience deviations when the vessel is moved.

Another uncertainty is that the HiPaP positioning system does not give a strictly accurate position of the ROV when it is close to the surface. This causes the autonomous control to deviate slightly in Launch, since the measurements can be fluctuating slightly, and thus the position estimates may deviate. As soon as the DVL sensor achieves BL, this is no longer a problem, but since the aim of this thesis is to perform a complete autonomous mission, it does somewhat prevent the vehicle to have an accurate performance in Launch.

The control systems try to position the vehicle by supplying it with thrust, but the false oscillating positions received from the HiPAP system results in sudden thrust inputs that. Such transients can cause wear and tear on the thrusters and equipment, and other redundant position measuring sensors could reduce this risk.

Further, the environmental conditions in the operational surroundings create another uncertainty when deploying the agent architecture on a real system. The full-scale results are quite accurate and successful due to the calm environmental conditions in the area of operation. Field work with ROV Minerva at Gunnerus earlier this year showed that when the current becomes large in the operational area, the vehicle is not able to withstand the current loads. It will, therefore, experiment a deviation in the position depending on the magnitude and direction of the current. Luckily, there were not much current during the testing of the autonomous operation, but it is still considered a severe uncertainty, given that autonomous ROVs should also aply to deep waters and preferably independently of the environmental conditions. However, the uncertainty lies in the capacity of the equipment, in this case, the maximum provided force from the thrusters, and an ROV should be suited with equipment according to its applications. 30k is a research ROV, and thus focuses on the quality of the sensors and equipment best suited for the research carried out. For the purpose of autonomous ROV interventions, the focus can be applied on the robustness of the equipment and propulsion system instead.

Overall, all the above uncertainties were observed after deploying the agent on a real system for full-scale experiments. Although simulation results are close to reality and suitable for testing and debugging the system, some unexpected aspects will always occur when performing full-scale results. These uncertainties demonstrate the importance of thorough testing in the development of a new design, which is advocated in the approach of this thesis (Section 1.3).

Chapter 8

Conclusion and Recommendations for Future Work

Based on the result in Chapter 5 and 6, and the discussion of the result in Chapter 7, a conclusion will be formulated in this chapter (Section 8.1). In Section 8.2, recommendations for future research on autonomous ROV operations are presented.

8.1 Conclusion

The research of this thesis is motivated by the desire to increase the level of autonomy in ROV operations. This has been a red thread throughout the thesis, and the work done by Pere Ridao [36] has been a fundament for this research. Ridao suggests a behavior-based, hybrid agent architecture consisting of three layers: a deliberative, a functional reactive and a control execution layer. Further, he proposes several approaches for the three layers, some of which are discussed in Section 2.1. Based on this previous research, the author has developed an architecture to show the autonomy potential for intervention type ROV missions. The deliberative layer uses a sequential steering approach, dividing the mission into sub-behaviors, planning the mission in detail. The reactive layer demonstrates the handling of non-deterministic events, i.e. OA, using computer vision for identification and navigation. The control execution layer is implemented as a hierarchical switch, by assigning each state a priority relative to the other states.
Results show that the agent architecture is able to maneuver the ROV without human intervention from surface until approach and localization of a SOI. In Section 1.1.6 the following is stated:

"The aim of this thesis is to study the possibility of increasing the level of autonomy in

current ROV operations from level 2 to 3 according to [8]."

Given the variety of points of view regarding autonomy, it is not simple to argue the above statement in definite terms. However, focusing on the definitions of the levels presented in Section 1.1.6, the author, believes that the proposed agent architecture is compliant with the third level of autonomy: *Management by exception*. The mission control system does not need to prompt the operator for specific information or decision making, but the operator may override or alter parameters during the mission. Of course, many aspects must be considered to achieve an autonomous system, and the agent considers a selection of these aspects. Nevertheless, the development of such an architecture is a groundbreaking contribution to increasing the autonomy of ROV operations, and more aspects of the autonomy can be added easily. Further, the following is stated in Section 1.2:

"This thesis should introduce an innovative contribution to state of the art autonomous ROVs, and open for further research on the subject."

Previous research focuses on the development of autonomous ROV operations and mission control architectures using mainly a theoretical approach [36, 58, 38, 19, 6, 34, 40, 4, 7, 48]. Other research focuses on the detailed path planning and replanning techniques of UUVs [5, 25, 47]. This thesis applies theory from both fields by developing a control architecture with integrated replanning techniques and presenting it through theoretical and experimental results. The author considers the *innovative contribution* of this thesis to be the combination of previous research to test and increase the level of autonomy in current ROV operations. The result of such a combination is a new hybrid agent architecture, tested and verified through reliable simulations and experiments.
Lastly, the following assertion is stated in Chapter 1:

"An autonomous approach on the ROVs could reduce the need of human intervention, and thus increase the accuracy, quality, and safety of such operations."

First of all, the reduction of human intervention is considered implicit in the increase in level of autonomy. Secondly, it can be argued that the safety around ROV operations is improved just by reducing the level of human interaction. The safety is considerably increased by replacing human divers with ROVs, moving the risk of damage on human to damage on equipment. Further, the accuracy of an automatically controlled vehicle is higher than the precision of a manually operated vehicle, since the latter is highly dependent on the sensing and responsiveness of the operator. Steering the ROV using a control system will move the sensing elements from the assisting vehicle to the actual ROV, and increase the responsiveness of the vehicle. Lastly, the increase in quality of the operation is considered implicit from the increase in accuracy, and the fact that ROVs can be equipped with better-suited actuators

than a human diver. This again would reduce the risk of damaging the equipment both located on the ROV and the SOI.
Based on the statements and argumentation above, it is concluded that the work presented in this thesis does certainly provide a possible solution to the initial objectives. Although there remains much to be done within the focused field of study, this contribution is considered a step towards the future in autonomous ROV operations. The architecture framework is recommended to use as a base for further research on the subject, with the focus on the content in Section 8.2.

8.2 Further Work

As of now, the mission control system is a preliminary framework for the development of a fully autonomous ROV. It is a future goal that the level of autonomy should be increased additionally, in order to maximize accuracy, quality, and safety in ROV operations. There was a time when ROVs were considered a hazard [55], and today they are integrated and considered crucial in a variety of research areas and operations. It is, therefore, believed that the future of ROVs is bright, and the author's suggestions for future focus are given in the following. Specific proposals for improvement discussed in Chapter 7 are listed in the following. Additional improvements are suggested in Section 8.2.1-8.2.3.

- Altering the altitude control using the current velocities as initial conditions in controller and observers as opposed to assuming zero velocities.
- When sending a new waypoint to the control system from the *Autonomy module*, the desired heading should be changed before moving in surge and sway, assuring the vehicle is always directed towards the desired waypoint.

8.2.1 Deliberative Improvement

The improvement of the deliberative layer should mainly focus on optimizing those states that are simulated or dormant in this thesis: CT and Intervention/Inspection. The use of stereo images for accurate positioning should be implemented in the low-level control system as an additional observer, using the work of Stud. Tech. Michele Gazzea (Briefly described in Appendix B) as a base. Further, the *SV module* should feed the distance from the stereo cameras to the object in the cameras into the *Autonomy module*, a work that is already in progress (Ph.D. Marco Leonardi collaborates on this as part of the contribution in Appendix

B). When the above steps are possible, the author advises using the distance to the object for control purpose in CT, while estimating the position using the new observer. When the distance to the object is less than a threshold value, the *Autonomy module* can switch to Intervention/Inspection, which leads to the next proposed step for improvement.

Inspection and intervention of SOIs form an enormous field of study but is something that needs to be studied and applied to underwater operations, in order to provide a fully autonomous vehicle. Using the example of opening a valve on a subsea installation, the camera recognition of the valve needs to be studied, but also the autonomous control of the manipulator arm. The author advises to use approaches discussed in [50] as a base for the former, and [41] as an inspiration to the latter.

Lastly, the deliberative layer could be improved by studying the possibility of adding new states. The author suggests doing so by familiarizing with the applications of a future autonomous ROV and specializing the deliberative layer according to the specific operations.

8.2.2 Reactive Improvement

Although the deliberative layer may be improved as suggested above, it is only a matter of applying different theories to control the vehicle using a known dynamic environment and predefined mission objectives. The real complexity and intelligence of an autonomous UUV lie in the functional reactive layer. This layer is what makes the vehicle truly autonomous (sense $\rightarrow$ act), and not only automatic (sense $\rightarrow$ plan $\rightarrow$ act). First of all, the author recommends to study different approaches regarding replanning techniques and collision avoidance, and implement them in the reactive state OA for testing. The work done by [25] should be used as a starting point. There exists much current research on collision avoidance techniques. The main work will be to implement it in the agent architecture. The *SV module* should be further improved as part of the OA state. It is crucial that this module can distinguish obstacles from fish and other non-threatening objects. In addition, it is necessary to make sure the distance to, and orientation of the obstacle is precise since a slight deviation could cause severe consequences.

Further, the author advises focusing research on additional reactive behaviors. A reactive state triggered by the thrust exceeding a maximum limit could be a start for such research. The vehicle should be able to slow down or stop if the control system somehow is requiring more thrust than what can be provided. This phenomenon can be caused by several events: the cable could be entangled, stuck, or the winch operator does not apply sufficient cable, and the vehicle will try to move forwards with no cable available. The thrust could also be

exceeding a limit due to large current effects and harsh environmental conditions. In the former case, the agent should slow down the vehicle and somehow detect what problem is causing the unavailable cable. The agent should notify the operator that cable is not available, in case it is caused by the operator handling the cable winch. This indicates another field of study for future operations: the automatic handling of winch for releasing cable. This could be implemented using the position of the ROV relative to the ship, and a control system for the onboard winch regulating the cable according to this position. Such a system could also reduce the risk of cable entanglement since it could prevent cable slack. In case the large thrust is caused by environmental conditions, the *Autonomy module* should slow down, but most important is the need for replanning of the mission according to the direction of the current. The bias model in the observer could be used for estimating the mean direction of the current, and the optimal direction of the ROV could be calculated from this, to assure the thrusters can control the vehicle. The *Autonomy module* should be able to replan the mission by using this optimal heading direction.
Overall, the reactive layer should be improved by adding new reactive states, and each of these could be studied individually by different students or researchers, as long as one researcher remains focus on the agent architecture for testing and improvement of the states.

8.2.3 Mission Control System

The mission control system could be improved by both reactive and deliberative research, but the author also recommends to study the optimization of the *Autonomy module* itself and its information flow. The control executive layer is currently comprised by a hierarchical switch, and other approaches could be studied, such as fuzzy-implemented behaviors [10]. Different approaches on the control executive layer could be compared and optimized to enhance the correlation between deliberative and reactive behaviors.
Through further research and study of the deliberate and reactive layers, an improvement of the autonomy level would be achievable. In addition to studying the different operations relevant for AROVs, the possibility of having two or more ROVs working together in tandem or groups should be studied. Interaction and cooperation between different UUVs could also be a cost reducing, optimizing alternative. Such an approach would require studying the management and optimization of each UUV during the operation. Of course, these proposals are time-consuming, and the research could take years, but increasing the autonomy of any subsea operation is a large step towards the future. The result of all the work could be an optimized subsea operation carried out with minimized human interaction, reduced costs,

and high safety. [38] and recommends several focus areas to increase the autonomy in ROV operations, and the author supports the suggested focus when further developing the agent architecture.

Overall, based on satisfactory results of this thesis, the author recommends continuing carrying out research on autonomous ROV interventions with the focus on the above aspects.

Bibliography

[1] (2011). *Herkules*. http://dykkepedia.com/wiki/Herkules [Accessed: 15.12.2016].

[2] (2016). *R/V Gunnerus*. https://www.ntnu.edu/oceans/gunnerus [Accessed: 15.12.2016].

[3] AUR-Lab (2013). Deployment av lbl nett ved tbs. Technical Report 1, NTNU.

[4] Barnett, D., McClaran, S., Nelson, E., McDermott, M., and Williams, G. (1996). Architecture for the texas a&m autonomous underwater vehicle controller. pages 231–237.

[5] Candeloro, M. (2016). *Tools and Methods for Autonomous Operations on Seabed and Water Column using Underwater Vehicles*. PhD thesis, NTNU.

[6] Carreras, M., Batlle, J., Ridao, P., and Roberts, G. N. (1999). An Overview on Behaviour-Based Methods for AUV Control. *Behaviour*, pages 141–146.

[7] Carreras, M., J.Batlle, Ridao, P., and Roberts, G. (2000). An overview on behavior-based methods for auv control. *Computer Vision and Robotics Group, University of Girona*.

[8] Committee on Autonomous Vehicles in Support of Naval Operations, N. R. C. (2005). *Autonomous Vehicles in Support of Naval Operations*.

[9] Corner, D. E. and Stevens, D. L. (1995). *Internetworking with TCP/IP: Col1 : Principles, protocols and architecure*.

[10] Evans, J., Patrón, P., Smith, B., and Lane, D. M. (2007). Design and evalutation of a reactive and deliberative collision avoidance and escape architecture for autonomous robots. *Springer Science+Business Media*.

[11] Flynn, A. M. and Brooks, R. A. (1989). Building robots: expectations and experiences. pages 236–243.

[12] Fossum, T. O., Ludvigsen, M., Nornes, S. M., Rist-Christensen, I., and Brusletto, L. (2016). Autonomous robotic intervention using rov: an experimental approach. *Ocean's 2016 Conference*.

[13] Goto, Y. and Stenz, A. (1987). The cmu system for mobile robot navigation. pages 99–105.

[14] Guizzo, E. (2010). Hiroshi ishiguro: The man who made a copy of himself. *IEEE Spectrum*.

[15] Hall, W. D. and Adams, M. B. (1992). Autonomous vehicle taxonomy. pages 49–64.

[16] Heis, J., Morgado, M., Batista, P., Oliveira, P., and Silvestre, C. (2016). Design and experimental validation of a usbl underwater acoustic positioning system. *Sensors 2016*.

[17] Henriksen, E. H. (2014). Rov control system for positioning of subsea modules. Master's thesis, Norwegian University of Science and Technology.

[18] Hwang, T., Roh, J., and Park, K. (2006). Development of hils systems for active brake control systems. *SICE-ICASE, 2006, International Joint Conference*.

[19] Insaurralde, C. C., Cartwright, J. J., and Petillot, Y. R. (2012). Cognitive control architecture for autonomous marine vehicles. *IEEE International Systems Conference (SYSCON)*.

[20] Kenny, A. (2016). *Recent Advances in Doppler Sonar Technology for Subsea Navgation and Current Profiling*. https://goo.gl/kPwJSW [Accessed: 10.05.2016].

[21] Kirkeby, M. (2010). Comparison of controllers for dynamic positioning and tracking of rov minerva. Master's thesis, Norwegian University of Science and Technology.

[22] Kongsberg (2016a). *Autonomous Underwater Vehicles AUV/Marine Robots*. https://goo.gl/dMge4p [Accessed: 20.04.2016].

[23] Kongsberg (2016b). *HiPAP*. https://goo.gl/lqS8gL [Accessed: 09.05.2016].

[24] Kongsberg (2016c). *MS1000/1007 Altimeter Briefing*. http://www.emepc.pt/images/pdf/KongsbergMesotech1007.pdf [Accessed: 10.05.2016].

[25] Korte, S. O. (2011). Guidance & control strategies for uuvs. Master's thesis, Norwegian University of Science and Technology.

[26] Ludvigsen, M. (1977). *An ROV toolbox for optical and acoustical seabed investigations*. PhD thesis, NTNU.

[27] NI (2016a). *FPGA Fundamentals*. `http://www.ni.com/white-paper/6983/en/` [Accessed: 07.05.2016].

[28] NI (2016b). *LabVIEW System Design Software*. `http://www.ni.com/labview/` [Accessed: 07.05.2016].

[29] NI (2016c). *The CompactRIO Platform*. `http://www.ni.com/compactrio/` [Accessed: 07.05.2016].

[30] Nornes, S. M. (2016). Verdandi simulator. E-mail correspondance.

[31] NTNU (2016a). *Mer om AUR-Lab*. `https://www.ntnu.no/aur-lab/om-aur-lab` [Accessed: 30.04.2016].

[32] NTNU (2016b). *Technical specifications and instrumentation of ROV Minerva*. `http://www.ntnu.edu/oceans/minerva/spesifications` [Accessed: 07.05.2016].

[33] Olsen, O. A. (2015). *Industrielle målemetoder*.

[34] Py, F., Rajan, K., and McGann, C. (2010). A Systematic Agent Framework for Situated Autonomous Systems. *Scientist*, pages 583–590.

[35] Repubblica, L. (2016). *FPGA Hiroshi Ishiguro: "Vi racconto il mio gemello androide"*. `https://goo.gl/A4vKvg` [Accessed: 22.12.2016].

[36] Ridao, P., Battle, J., Amat, J., and Roberts, G. (1999). Recent trends in control architectures for autonomous underwater vehicles. *International Jornal of Systems Science*.

[37] Ridao, P., Yuh, J., Batlle, J., and Sugihara, K. (2000). On AUV control architecture. *Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, pages 855–860 vol.2.

[38] Schjølberg, I. and Utne, I. B. (2015). Towards autonomy in rov operations. *IFAC-PapersOnLine*.

[39] Seto, M. L. (2013). *Marine Robot Autonomy*.

[40] Simmons, R. G. (1994). Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, pages 34–43.

[41] Åsmund Pedersen Hugo (2016). Kinematic control of underwater robotic system. Master's thesis, Norwegian University of Science and Technology.

[42] Society, M. T. (2015). *What is an ROV?* `http://www.rov.org/rov_overview.cfm` [Accessed: 30.03.2016].

[43] Sonardyne (2016). *Sonar*. `http://www.sonardyne.com/products/positioning/lbl.html` [Accessed: 09.05.2016].

[44] Sperre (2016). *SUB-FIGHTER 30K*. `http://sperre-as.com/portfolio/sub-fighter-30k/` [Accessed: 28.04.2016].

[45] Sørensen, A. (2011). *NTNU Applied Underwater Robotics Laboratory AUR-Lab*. `https://goo.gl/DctZeY` [Accessed: 15.05.2016].

[46] Sørensen, A. J. (2013). *Marine Control Systems*.

[47] Sørensen, A. J., Dukan, F., Ludvigsen, M., Fernandes, D. A., and Candeloro, M. (2012). Development of dynamic positioning and tracking system for the rov minerva.

[48] Sørensen, A. J. and Ludvigsen, M. (2015). Towards integrated autonomous underwater operations. *IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles*.

[49] Stavdahl, ., Malvig, K. E., Pettersen, O., Gravdahl, J., Gravdahl, I., From, P., and Eielsen, A. A. (2015). *Tilpassede datasystemer*.

[50] Sveier, A. (2016). Primitive shape detection in point clouds. Master's thesis, Norwegian University of Science and Technology.

[51] U.S. Department of Transportation, F. H. A. (2005). *Clarus - Concept of Operations*.

[52] Vlahavas, I. and Vrakas, D. (2005). *Intelligent Techniques for Planning*. IGI Global.

[53] Volden, F. and Torres, P. (2016). Aur-lab server.

[54] Website, T. W. (2011). *Halifax LK789 crash, April 1944, Colony Farm*. `https://goo.gl/5jaPP9` [Accessed: 13.12.2016].

[55] Wernli, R. L. (1998). *Operational Effectiveness of Unmanned Underwater Systems*.

[56] Wöhler, C. (2013). *3D Computer Vision*.

[57] Xsens (2016). *IMU Inertial Measurement Unit*. https://www.xsens.com/tags/imu/ [Accessed: 07.06.2016].

[58] Yavuz, H. and Bradshaw, A. (2002). A new conceptual approach to the design of hybrid control architecture for autonomous mobile robots. *Journal of Intelligent and Robotic Systems*.

Appendix A

Electronic Attachments

The files in this appendix are included in electronically submitted versions.

A.1 Autonomy_framework.vi

LabVIEW program-subroutine for the ROV Control System: this VI is the hybrid agent architecture implemented in this thesis. Deploying it in the Mission Control level of the ROV Control System will enable simulations and full-scale testing.

A.2 Transponder_avoidance.mp4

Video from the HD cameras on 30k during Collision Avoidance of transponder tower. The video corresponds to the mission of which results are presented in Appendix E.3.

Appendix B

Abstract for Paper Contribution to Ocean's 17

Vision based obstacle avoidance and motion tracking for autonomous behaviors in underwater vehicles

Marco Leonardi[1], Annette Stahl[1], Martin Ludvigsen[2]
Michele Gazzea[1], Ida Rist-Christensen[2], Stein M. Nornes[3]

[1]Department of Engineering Cybernetics, NTNU, Trondheim, Norway.

[2]Department of Marine Technology,
Applied Underwater Robotics Lab, NTNU, Trondheim, Norway.

[3]Centre for Autonomous Marine Operations and Systems,
Department of Marine Technology, NTNU, Trondheim, Norway.

marco.leonardi@itk.ntnu.no, micheg@stud.ntnu.no, idaris@stud.ntnu.no

Performing reliable underwater localization and maneuvering of Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs) near historical sites, nature protection areas or even man-made structures is a difficult task. Different sensing techniques have been exploited with sonar being the most often used sensing element to extract depth information or to avoid obstacles. However, little has been published on complete control systems that use robotic vision for such underwater applications.
This paper provides a proof of concept regarding a series of experiments investigating the use of stereo vision for underwater obstacle avoidance and position estimation. The overall aim of this concept study is to present developed methods that increase the autonomous capabilities of ROVs in order to make ROV missions safer and build the required knowledge for incorporating vision for ambient awareness and mapping in the most recent AUVs.
We propose a vision based obstacle avoidance method exploiting rectified stereo image pairs from a calibrated system to compute respective disparity maps. From each disparity map a 3D point cloud representing the ambiance has been extracted. The point cloud, as well as the original disparity map in general, contain noise due to false correspondence matches. These false matches have been eliminated using a modified version of the Density-based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [Ester et al. 1996]. In general, one of the assumptions within the DBSCAN algorithm states that all clusters defined should have the same density distribution. However, this assumption is inconsistent for a point cloud extracted from a stereo vision system, since the cluster density might be represented as an inverse function of the distance between the object and the camera (object pointing far from the camera have a lower average density than points close to the camera). In order to model this relation, the respective epsilon parameter has been modified to represent a function of the actual depth of the points.
In addition, corresponding stereo image pairs were utilized to extract and track features within consecutive image frames, in order to compute the apparent motion of the vehicle. This movement was then translated into a motion estimate with respect to the camera reference. Features were detected with the *Shi-Tomasi* corner detector [Shi and Tomasi 1994], with the Scale-Invariant Feature Transform (SIFT) [Lowe 1999] and with Speeded Up Robust Features (SURF) [Bay et al. 2008]. Once good features were found, we deployed a feature tracker to register the displacement between vectors; different trackers have been tested and their performance compared. From this procedure

we obtain a series of displacement vectors containing information about apparent feature motions. The obtained shifts are then combined to obtain the overall image motion. Outlier rejection has been performed using a median filter and a RANdom SAmple Consensus (RANSAC) circle fit on the data points.
The stereo vision based collision avoidance system was implemented as a reactive part of an autonomy layer in the mission control system used for the autonomous ROV intervention [Fossum 2016]. The autonomy layer contains a deliberative module switching between several predefined behaviors in parallel with a system of reactive behaviors taking control of the vehicle in case of unwanted and unforeseen events, like an obstacle in the path of the vehicle. The system provides guidance input to the control system on the ROV. In our experiment, the vehicle was sent towards an obstacle known to the operator, in order to purposely test the detection of obstacles and to provoke reactive behavior for obstacle avoidance using real time computer vision. When the obstacle is detected in the camera images, the orientation of the vehicle, and distance to the obstacle is sent to the autonomy layer through User Datagram Protocol (UDP) communication. As a result, the reactive behaviour demands a change in heading of the vehicle. The vehicle turns until the obstacle is no longer detected in the stereo images, and a new way-point is defined straight ahead. Figure 1 is a xy-plot of the vehicle during the testing of collision avoidance.

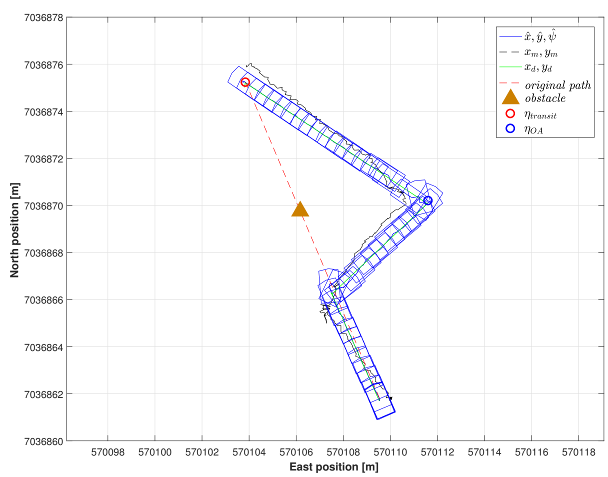


Figure 1. ROV position in xy-plane during collision avoidance.

The experiments were conducted in the Trondheimsfjord using the ROV SPERRE SUB-fighter 30K that has been equipped with a straight forward looking stereo vision system.

References

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231.

Fossum, e. a. (2016). Autonomous robotic intervention using rov: An experimental approach. OCEANS 2016 MTS/IEEE – Monterey.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2.

Shi, J. and Tomasi, C. (1994). Good features to track. IEEE Conference on Computer Vision and Pattern Recognition, Seatte.

Appendix C

Huffman Table for Sequential Steering

When developing a sequential process consisting of different states, it may be convenient to sort the states and their transitions in a structural manner. This creates a systematic overview before implementation and programming. The states are therefore sorted in a so-called Huffman table, inspired by [49]. A Huffman table is prepared as a three-dimensional matrix with $n \cdot m \cdot 2$ elements, where n represents the amount of inputs, and m the amount of outputs. Inputs and outputs meaning conditions for the states to be triggered and completed, respectively. For each value $i \leq m, j \leq n$, the element (i,j) contains information about the next state, s_i, and action/output, w, respectively. The Huffman table is organized as in Table C.1.

States s	Inputs					
	p_1	p_2	...	p_i	...	p_m
s_1	-/-	-/-		-/-		-/-
s_2	-/-	-/-		-/-		-/-
s_3	-/-	-/-		-/-		-/-
...						
s_j	-/-	-/-		q_s/w		-/-
...						
s_n	-/-	-/-		-/-		-/-

Table C.1: *General Huffman table [49]*

The complete autonomous mission is organized in a Huffman table given in Table C.2. The inputs and outputs are listed below, and the state transitions are represented by the bold elements in the table. The transitions are triggered by their respective inputs, and as a result, the respective output occurs. All deliberative state transitions are triggered by timeout in a timer that is reset when the vehicle reaches the desired position/altitude/depth of the current state. State transitions between deliberate states and OA are triggered by the detection of an obstacle in the cameras. The result is the obstacle avoiding algorithm explained in Section 3.8. More states can be added to the table as the mission becomes more complex.

State	Inputs								
	Init	Com/Sen	BL	Pos	Alt	Depth	Timeout	UDP/Cam	OD
1. *Launch*	1/DP	1/DP	1/-	1/-	1/-	1/res	**2/DP**	1/L	**6/OA**
2. *Descent*	2/-	2/DP	2/AltC	2/-	2/res	2/BL!	**3/DP**	2/L	**6/OA**
3. *Transit*	3/-	3/DP	3/AltC	3/res	3/-	3/-	**4/DP**	3/L	**6/OA**
4. ST	4/-	4/DP	4/AltC	4/res	4/-	4/-	**5/DP**	4/L	**6/OA**
5. CamT	5/-	5/DP	5/AltC	5/SK	5/SK	5/SL	5/L	5/FB	**6/OA**
6. OA	**6/-**	**6/-**	**6/-**	**6/-**	**6/-**	**6/-**	**6/-**	6/L	6/-

Table C.2: *Huffman table for autonomous mission*

Inputs:

- *Init*: The autonomous system is initialized by operator
- *Com/Sen*: Communication and sensor functionality is verified and available
- *BL*: BL is achieved in the DVL sensor
- *Pos*: ROV position is within a certain range
- *Alt*: Altitude is within a certain rangewith respect to the desired altitude
- *Timeout*: Time has elapsed since timer was reset
- *Depth*: Depth is within a certain range, 5 meters for *Descent*
- *UDP/Cam*: UDP connection is available and functioning. Camera is functioning, calibrated and feedback enabled
- *OD*: Obstacle detected in stereocameras

Outputs:

- *DP*: New DP message is sent to the control system with new desired coordinate, depth or altitude
- *AltC*: Altitude control is activated
- *res*: Timer is reset

- *BL!*: Message to operator: 'BL Idle'
- *SK*: Stationkeep vehicle
- *FB*: Feedback from cameras is recieved
- *L*: Listening to stereo cameras for feedback

An example on how to read the table is presented using the column with red content. It applies when the input "Pos", and the state *Transit* are true. "Pos" implying that the vehicle has reached the desired *Transit* coordinates, which means that the vehicle must stabilize at that position before switching to the next state. The number 3, in the column, implies that the new desired state for the current input is still *Transit*. The output "res" is a result of the input, which is to reset the timer. When the time has elapsed in the timer, the new input, "Timeout", is active, and the state is switched to the next desired state.

Appendix D

Additional Results from Simulations

D.1 Autonomous Mission without OA

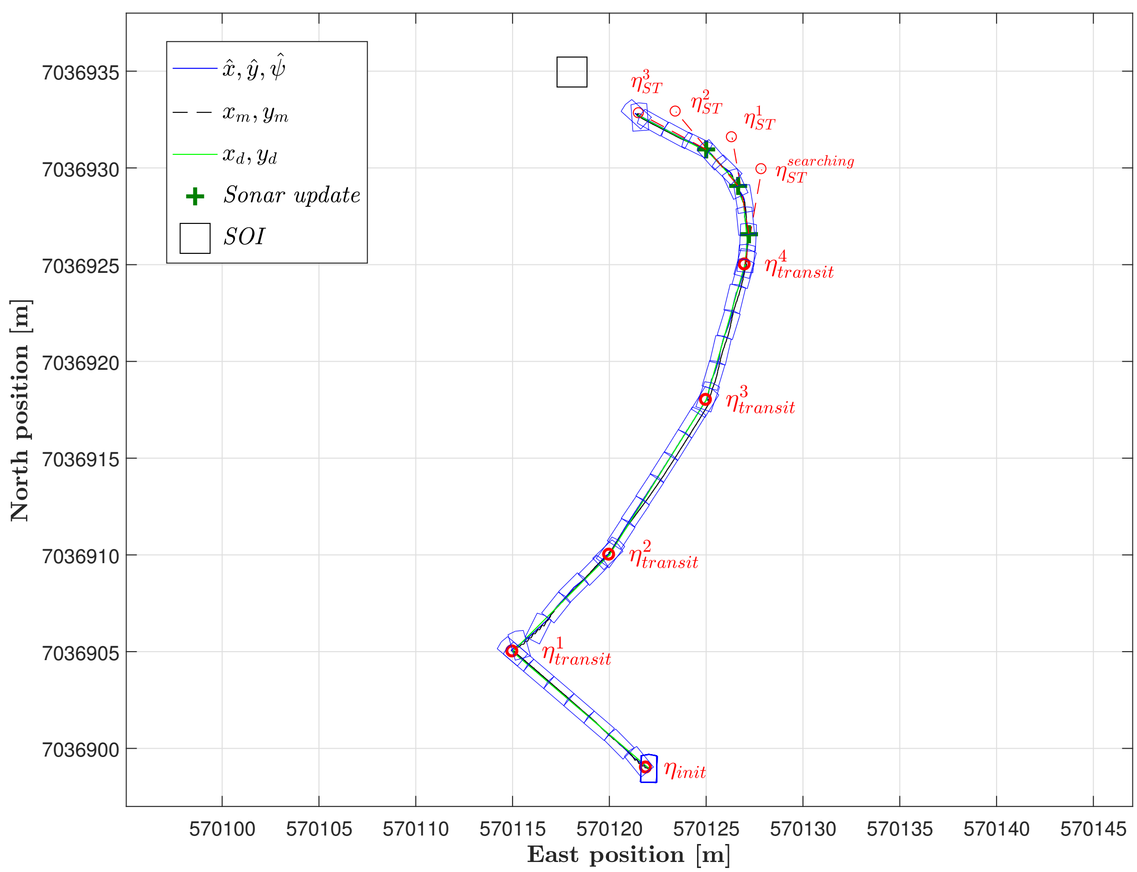
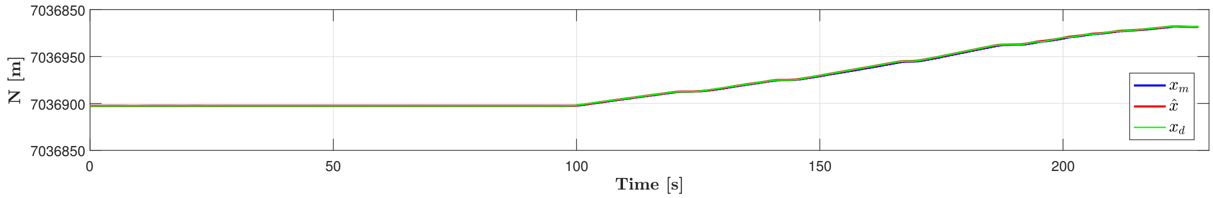
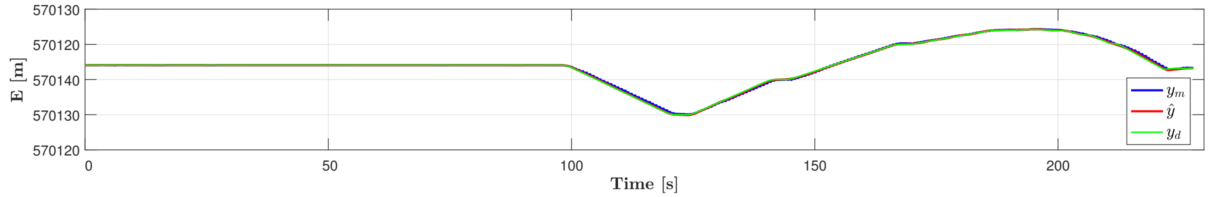


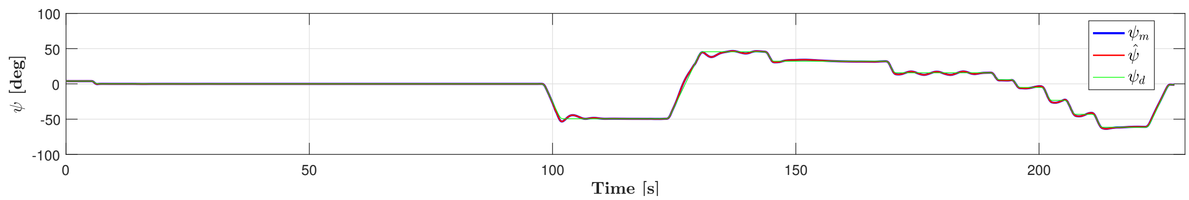
Figure D.1: *Estimated, measured and desired ROV position in NE plane*



(a) *Estimated, measured and desired position in north direction over time*

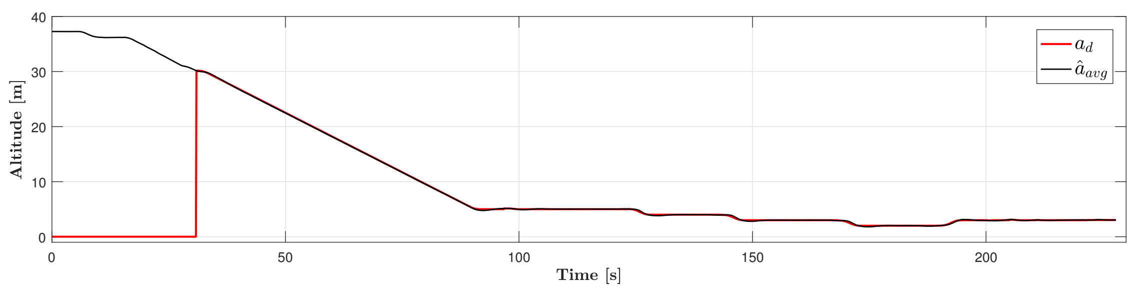


(b) *Estimated, measured and desired position in east direction over time*

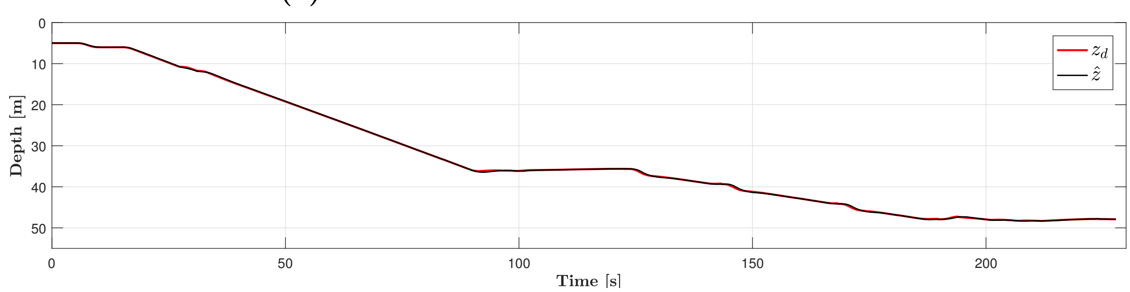


(c) *Estimated, measured and desired heading over time*

Figure D.2: *Estimated, measured and desired ROV position and heading over time*



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure D.3: *Estimated and desired ROV altitude and depth over time*

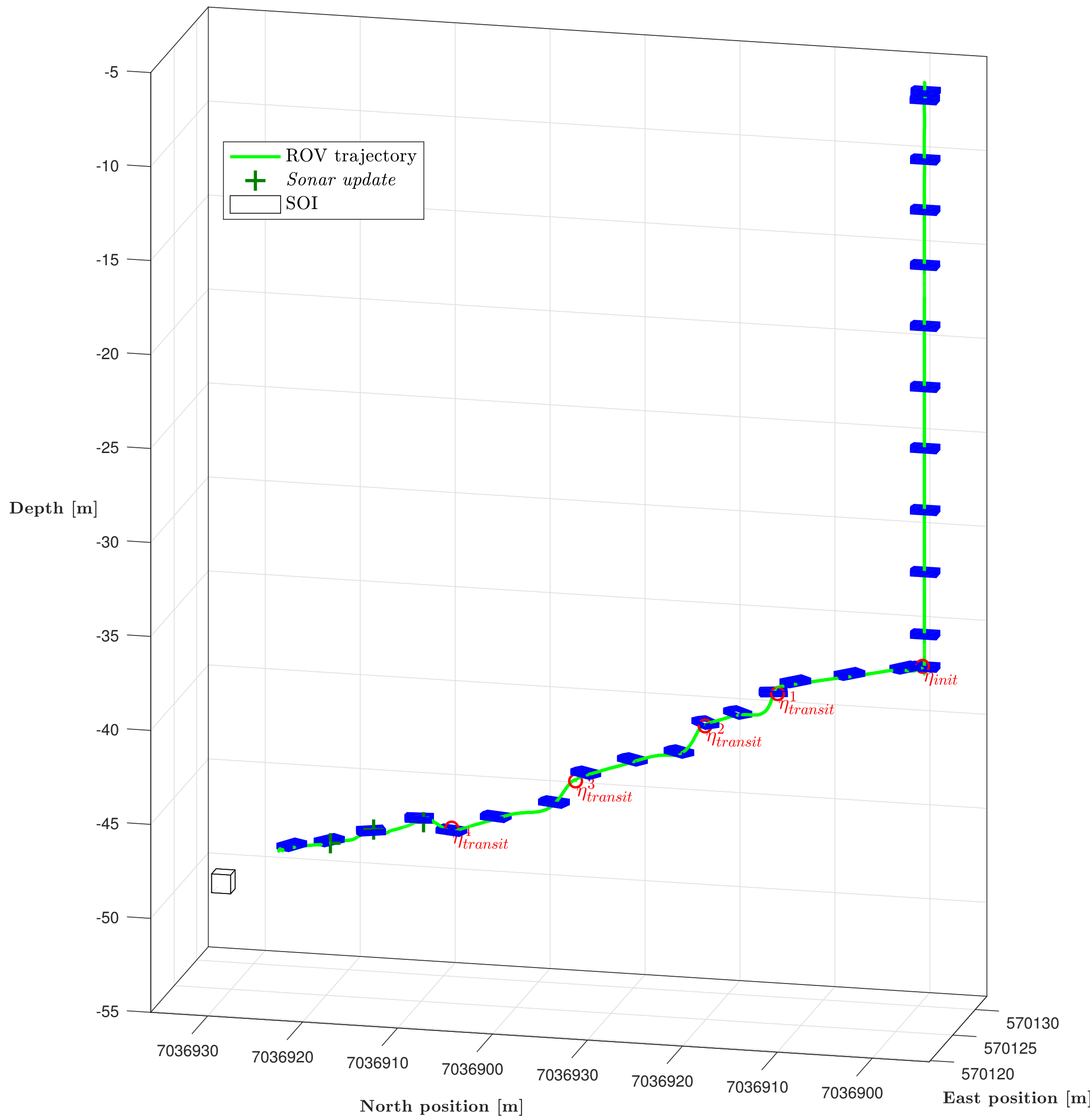


Figure D.4: *3D illustration of the vehicle during mission control*

Appendix E

Additional Results from Full-scale Experiments

E.1 OA with $\beta = 45°$

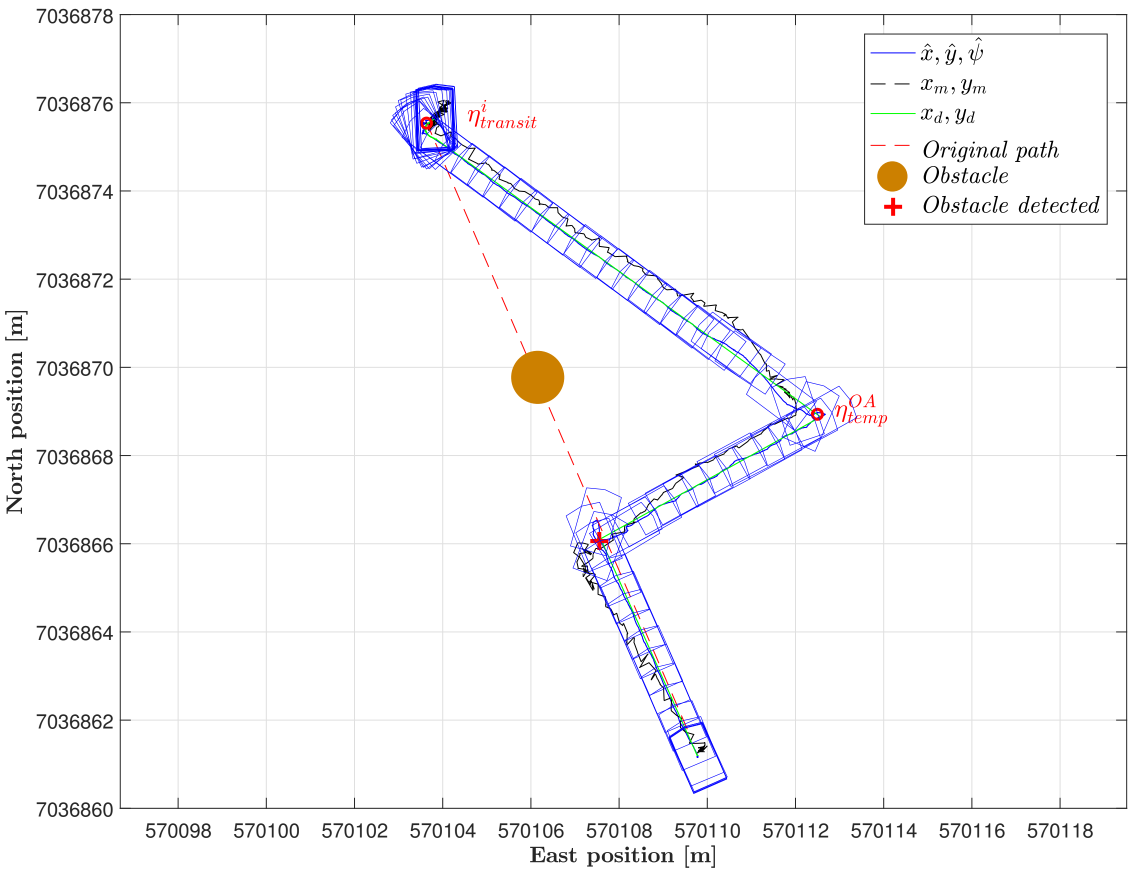
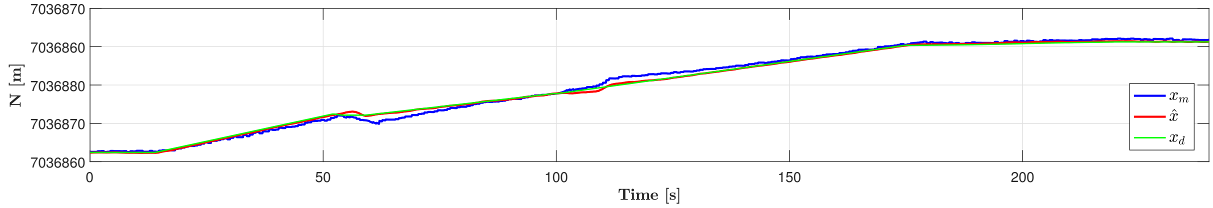
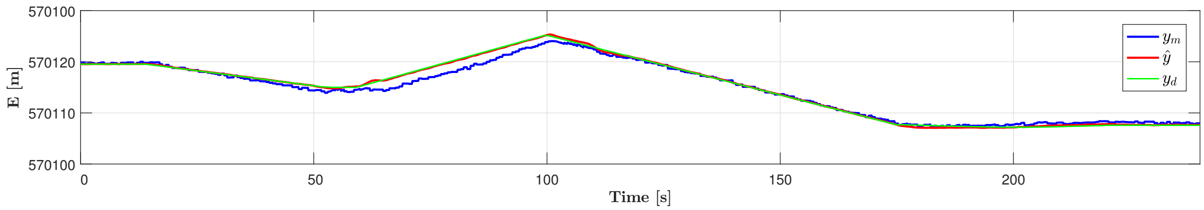


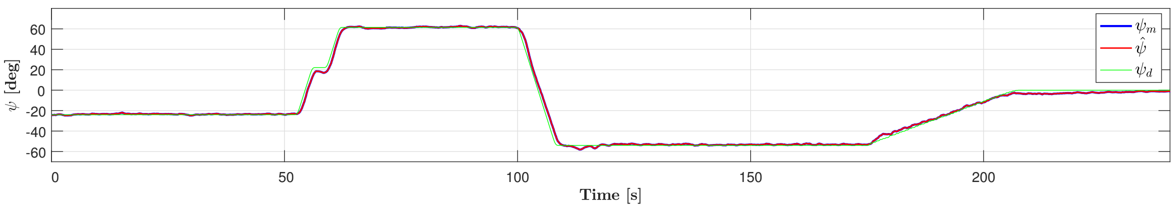
Figure E.1: *Estimated, measured and desired ROV position in NE plane*



(a) *Estimated, measured and desired position in north direction over time*

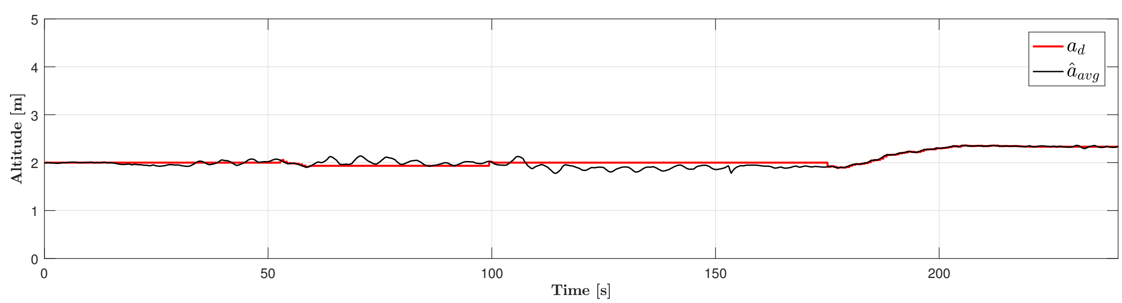


(b) *Estimated, measured and desired position in east direction over time*

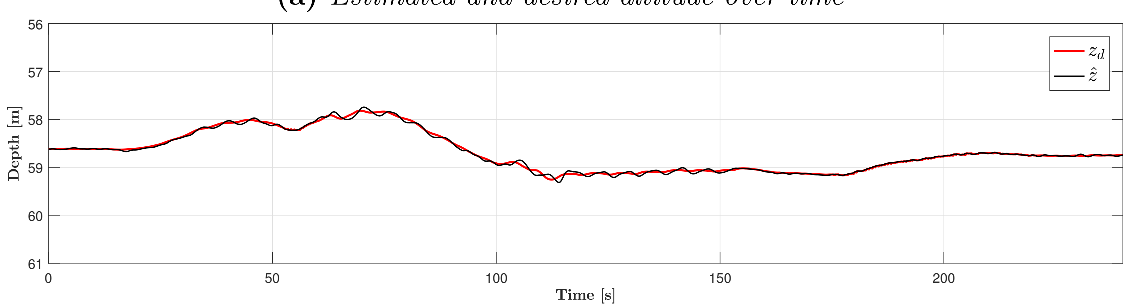


(c) *Estimated, measured and desired heading over time*

Figure E.2: *Estimated, measured and desired ROV position and heading over time*



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure E.3: *Estimated and desired ROV altitude and depth over time*

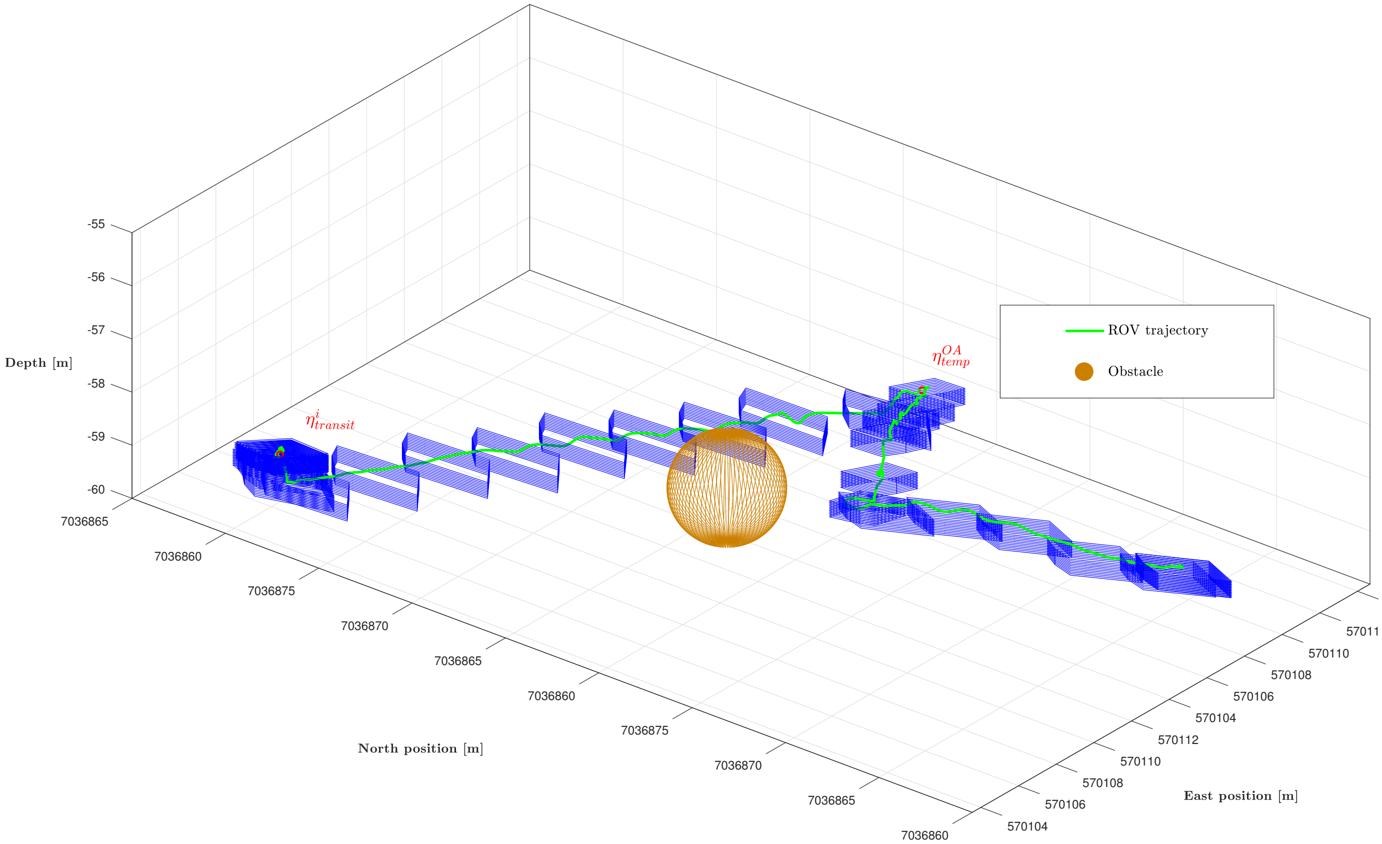


Figure E.4: *3D illustration of the vehicle during mission control*

E.2 Autonomous Mission without OA

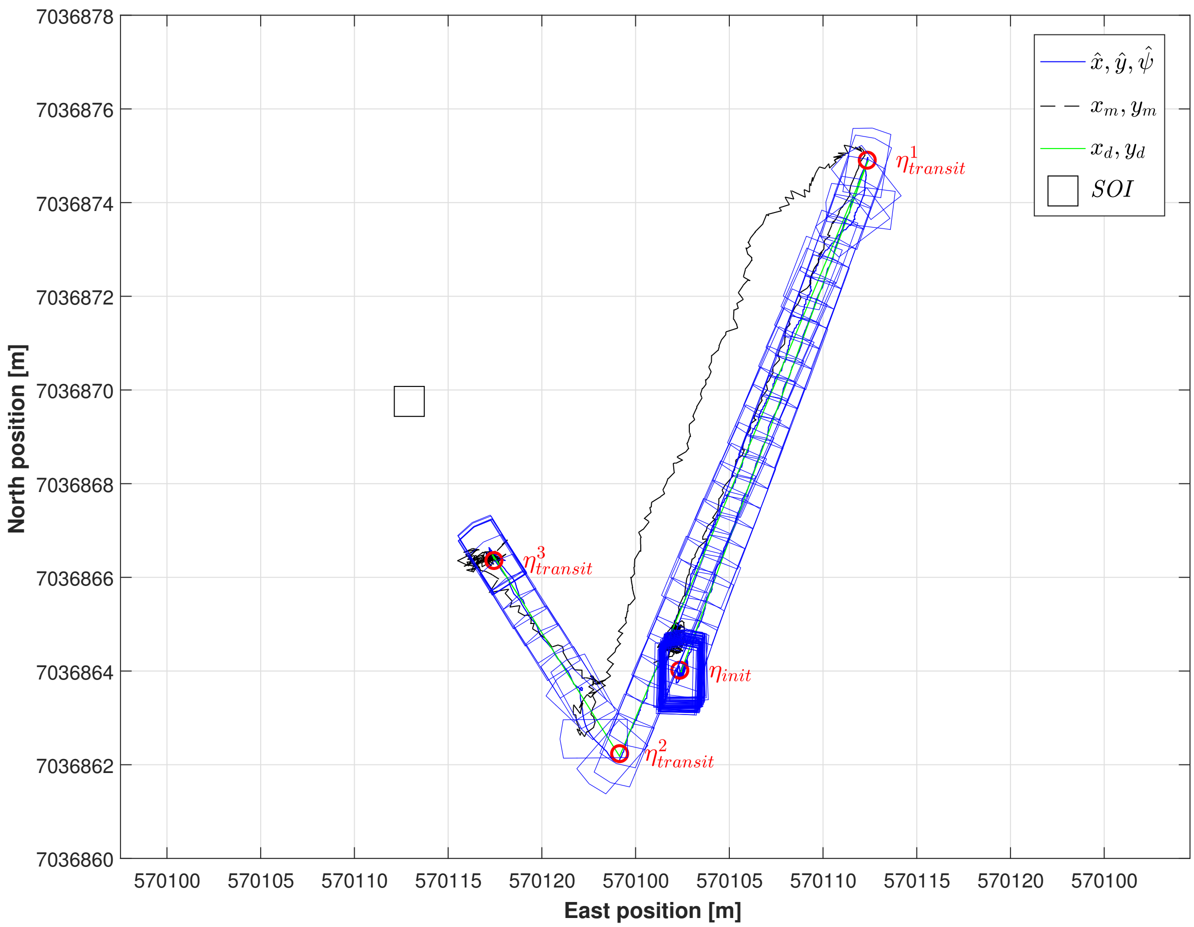
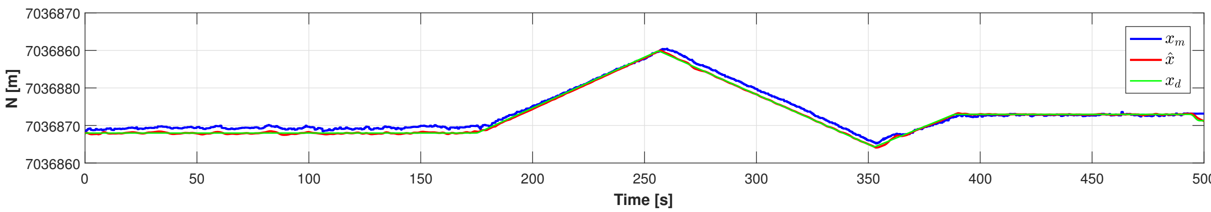
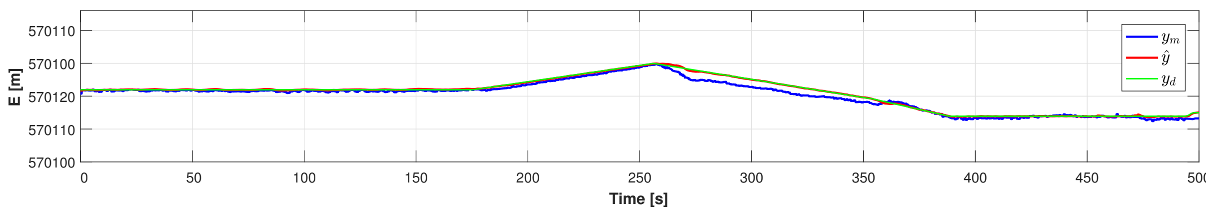


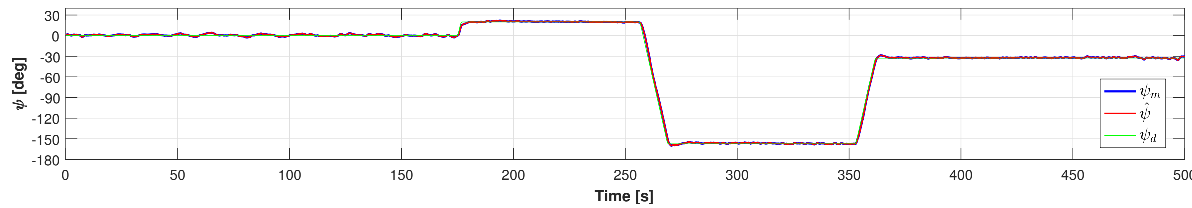
Figure E.5: *Estimated, measured and desired ROV position in NE plane*



(a) *Estimated, measured and desired position in north direction over time*

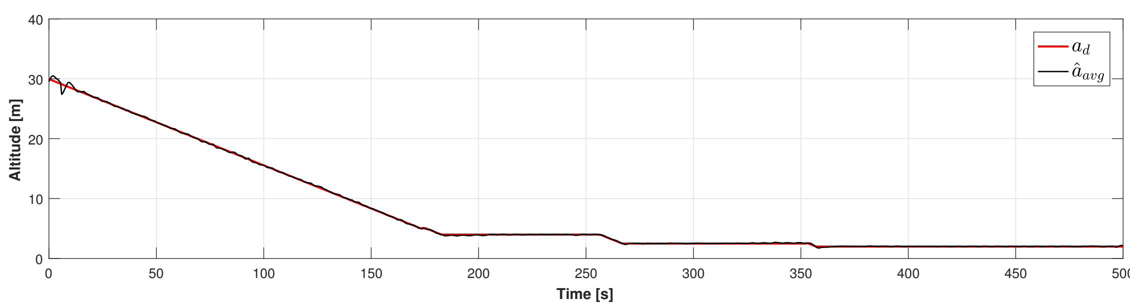


(b) *Estimated, measured and desired position in east direction over time*

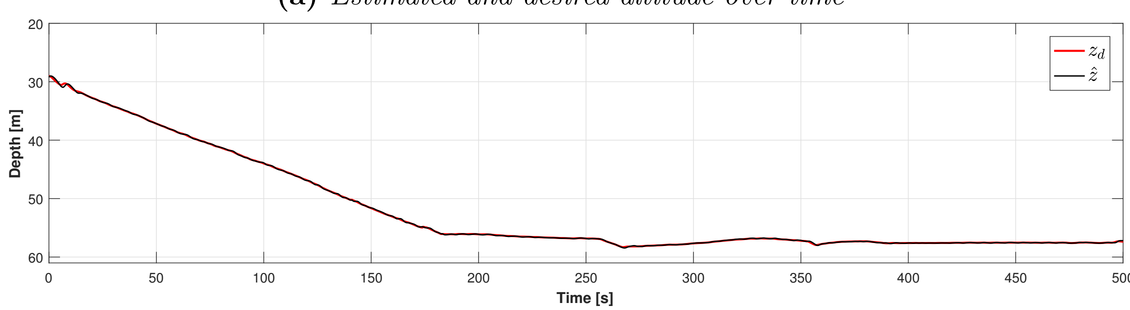


(c) *Estimated, measured and desired heading over time*

Figure E.6: *Estimated, measured and desired ROV position and heading over time*



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure E.7: *Estimated and desired ROV altitude and depth over time*

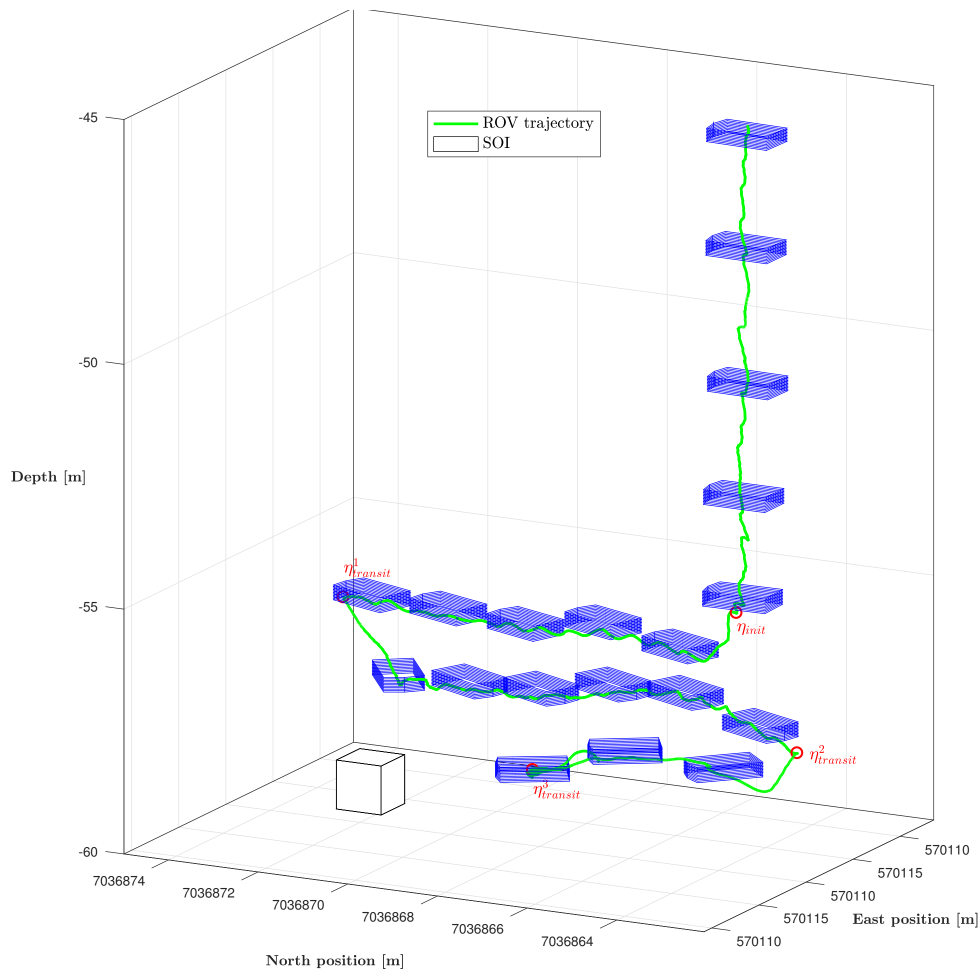


Figure E.8: *3D illustration of the vehicle during mission control*

E.3 Autonomous Mission with OA

The mission was interrupted by the operator before completing the autonomous mission, which is why the plots are not showing the entire mission. Notice that during this trial, the transponder tower described in Section 4.4 is used as an obstacle

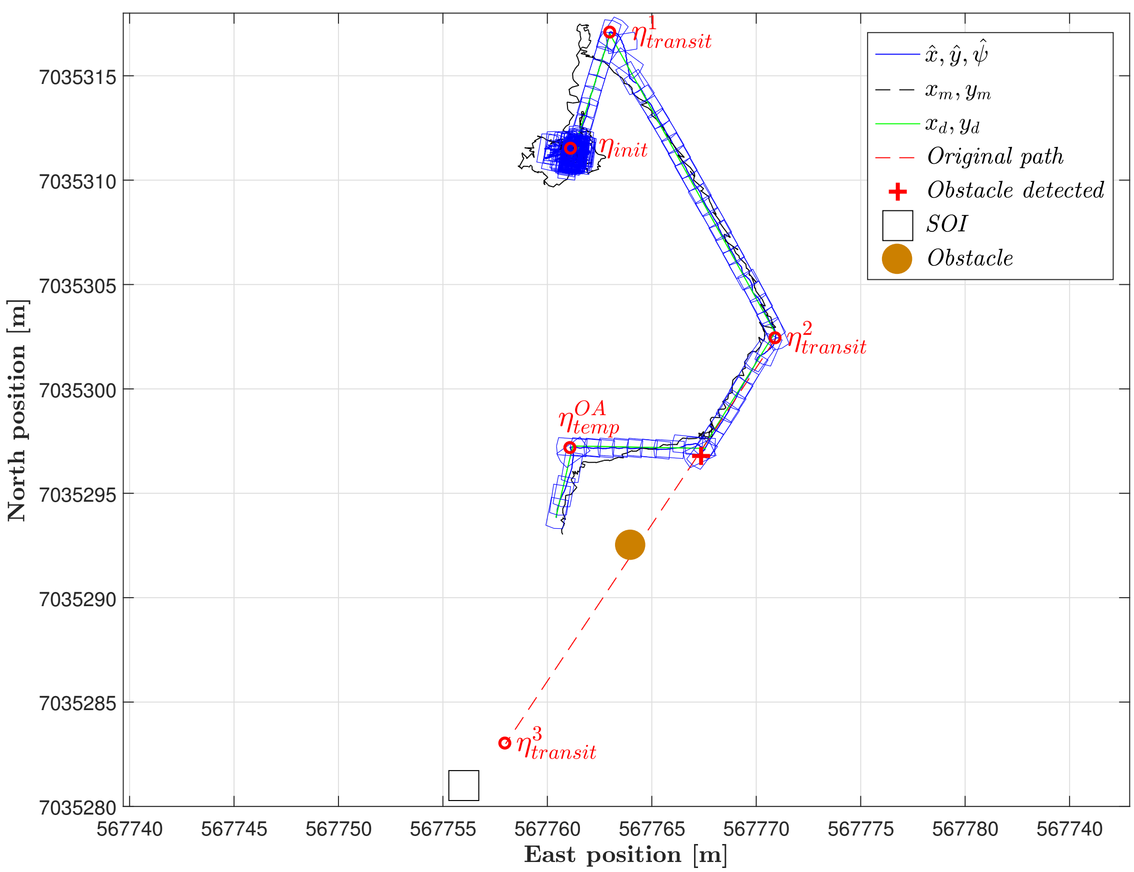
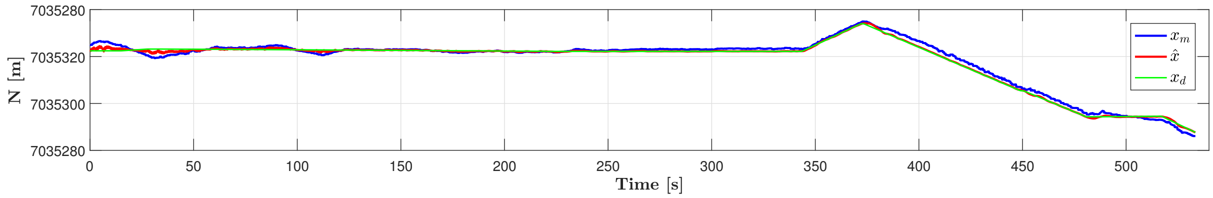
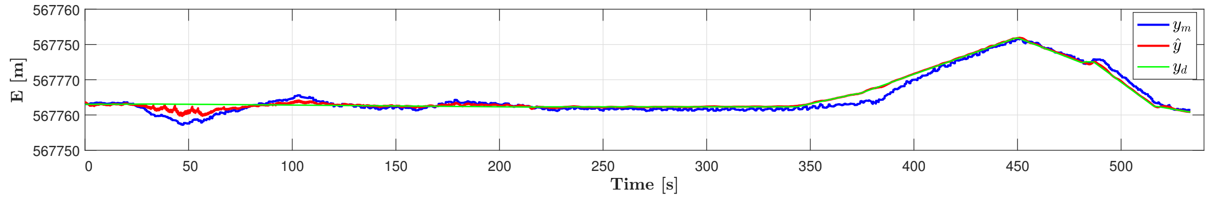


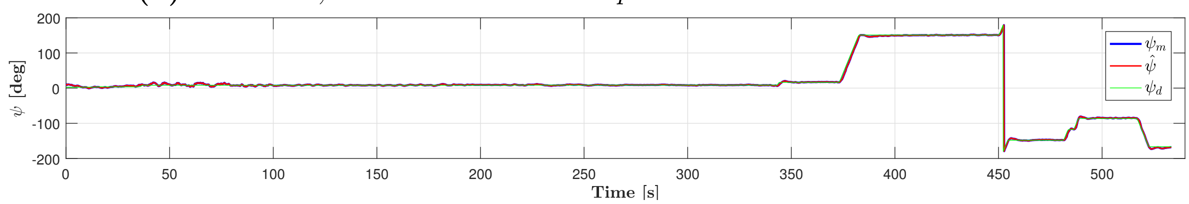
Figure E.9: *Estimated, measured and desired ROV position in NE plane*



(a) *Estimated, measured and desired position in north direction over time*

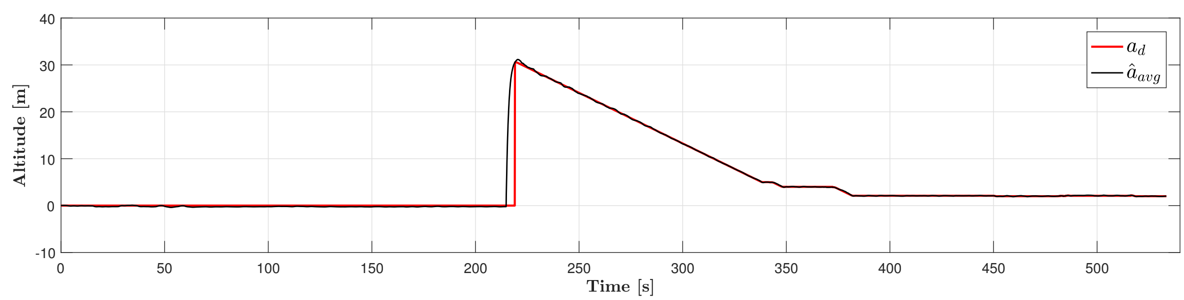


(b) *Estimated, measured and desired position in east direction over time*

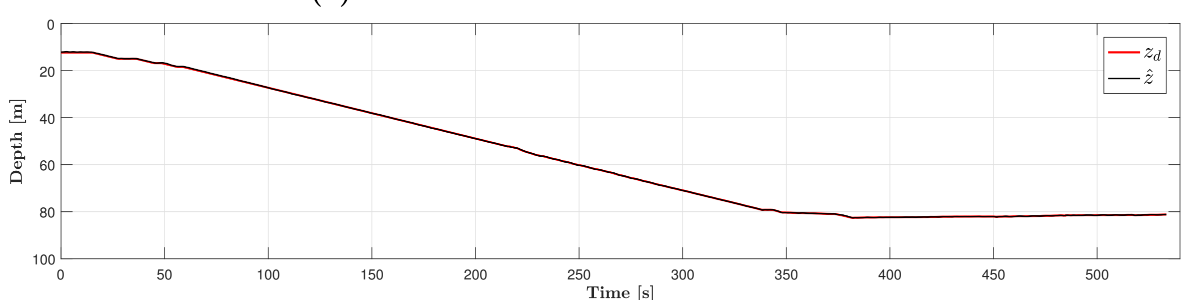


(c) *Estimated, measured and desired heading over time*

Figure E.10: *Estimated, measured and desired ROV position and heading over time*



(a) *Estimated and desired altitude over time*



(b) *Estimated and desired depth over time*

Figure E.11: *Estimated and desired ROV altitude and depth over time*

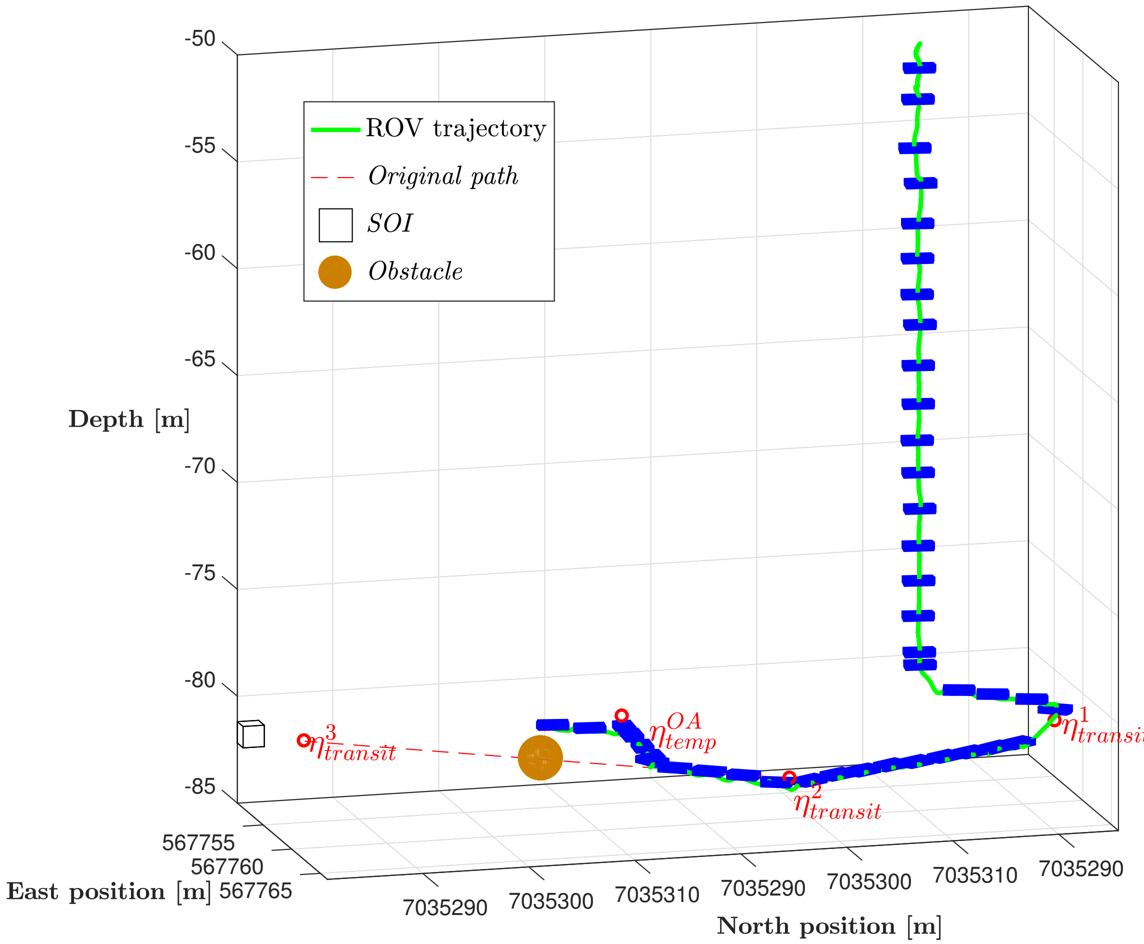


Figure E.12: *3D illustration of the vehicle during mission control*