NTNU
Norwegian University of
Science and Technology

Motion Control Systems for ROVs

Underwater Path-Following for a Videoray
Pro 4 ROV

Bent Oddvar Arnesen

Master of Science in Engineering and ICT
Submission date: June 2016
Supervisor: Ingrid Schjølberg, 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. Bent Oddvar Arnesen

Motion Control Systems for ROVs

Underwater Path-Following for a Videoray Pro 4 ROV

Work Description

The control of Remotely Operated Vehicles (ROVs) is, as the name suggests, performed remotely by a human operator, normally with the use of a controller with joystick(s). To operate an ROV, maneuvering skills are a necessity, and the costs required for training and having an operator in the field are often high. Not only is it expensive, but the error source brought by introducing humans into the robotic system loop may be significant. By implementing various automatic control techniques for the ROV, operators can more easily steer the ROV as desired, and the ROV can even carry out entire missions single-handedly.

This master's thesis describes the development of a guidance and control system for a Videoray Pro 4 ROV. More specifically, what will be created is a system ensuring that a path can be followed automatically. This includes all necessary functionality in order to make the ROV accomplish path-following. In addition, the ROV's lateral thruster will be studied, as it is not possible to control it in a satisfactory manner. In order to receive position data, a method for connecting to the Qualisys Motion Tracking (QMT) system will be developed in Matlab. This functionality is currently only available in C/C++ through Robot Operating System (ROS), which is a collection frameworks for robot software development.

Using Unmanned Underwater Vehicles (UUVs), a diversity of tasks can be solved efficiently and safely and without endangering human lives. By developing a motion control system for such vessels, these tasks can be solved without the need for a human intervention. The functionalities must incorporate safety, and should be optimized in terms of efficiency, fuel consumption, etc. For an ROV, among these features we find dynamic positioning (DP) systems to automatically maintaining a position or orientation, automatic path-following and tracking systems, and so on. More functionality means that a greater specter of tasks can be assigned to ROVs. In turn, this increases possibilities for using ROVs for missions that are too dangerous for humans.

Firstly, we will explore the literature for systems that deal with guidance, navigation and control. This literature review creates the foundation upon which the development of a motion control system used for waypoint guidance/path-following for the Videoray Pro 4 ROV will be built. Lastly, it will study mathematical modeling for ROVs, which will enable us to develop a model of the Videoray ROV to carry out simulation tests. All code will be written in Matlab/Simulink and C/C++, followed by implementation and testing of the solution in the Marine Cybernetics lab (MC-lab) at NTNU. This also includes setting up the QMT system for position measurements in Matlab, which is currently only available for systems written in C/C++ and systems using Robot Operating System (ROS).

Scope of work

1. Experimental work related to the lateral thruster of the Videoray Pro 4 ROV and examination of how to control this thruster. If the lateral thruster can be used with the ROV, the craft is fully actuated (neutrally stabilizing in roll and pitch), and it is possible to control the vehicle in four degrees of freedom (4 DOFs). If not, the craft is underactuated, but can still achieve great path-following results.

2. A literature study of specific topics relevant to the thesis work will be included. Focus is centered around marine control systems for ROVs, with a greater emphasis on

    a. Motion control systems for ROVs: Guidance, navigation and control

    b. Methodologies used to develop a suitable path-following and/or tracking system for ROVs

    c. Mathematical modeling of ROVs that incorporate properties similar to those manifested by the Videoray Pro 4 ROV

3. Set up the Qualisys Motion Tracking (QMT) system and a method to access position and orientation measurements in Matlab. The QMT system already exists in C/C++ that uses Robot Operating System (ROS), but is currently unavailable for applications that run on Matlab.

4. Create a simulation model of the Videoray Pro 4 ROV based on mathematical modeling principles. Design the system in a nice, structured and intuitive fashion, making implementation and verification of new functionality easier

5. Experimental work with the Videoray Pro 4 ROV in a test facility and simulations related to the simulation model. More specifically, the objectives are concluded by;

    a. Development, implementation and testing of control techniques for controlling depth and heading for both the simulation model and the Videoray Pro 4 ROV

    b. Development, implementation and testing of a path-following methodology for both the simulation model and the Videoray Pro 4 ROV

    c. A report on the simulation and experimental testing, including discussions and comparisons of the results. Conclusions will be presented along with recommendations for further work.

The master's thesis will be written in English, structured and edited as a research report. To begin with, preface, abstract and a summary will be presented, followed by a table of contents and lists of figures, tables and abbreviations. The summary will be written in both English and Norwegian, as requested by the departmental guidelines for writing master's theses at NTNU. Next, the report will present an introduction accounting for the motivation behind the project and relevant background material, together with the scope of work, contributions and thesis outline. The main body of the thesis contains a literature review, description of the ROV's marine control system, results from simulations and experiments, discussions and conclusions with proposals for further work. Ultimately, a list embodying references and (optional) appendices will be attached. Additional appendices containing source code, etc. may be provided on a memory stick, CD or the like. It is understood that the Department of Marine Technology, NTNU, can use the results freely in its research work, unless otherwise agreed upon, by crediting this report.

Supervisor: Ingrid Schjølberg

PREFACE

This report comprises the Master's Thesis in Engineering and ICT - ICT & Marine Technology. The underlying research and development work has been carried out at the Norwegian University of Science and Technology during the spring semester of 2016. The main motivation for the work has been to implement automatic path-following functionality for an actual system, and to verify the performance by experiments. Proposed background for the reader is an adequate competence in marine technology, cybernetics, or a general interest in ROVs and guidance or control systems.

Acknowledgements

I would like to thank my counsellor and supervisor, Ingrid Schjølberg, for all the motivation, encouragement and support. Not to mention, for making this assignment possible and for enhancing my interest in marine technology and cybernetics. I would also like to thank the Centre for Autonomous Marine Operations and Systems (AMOS) for letting me use the Videoray ROV. Another thank you goes to Torgeir Wahl at Marintek for the help related to the Qualisys underwater camera system.

Ove, thank you so much for your feedback, and for taking your time to review and suggest improvements to the grammatical aspect of this work.

To my family, friends and closest ones, my sincerest thanks to you for all the support and motivation you have given me throughout my years of study.

Bent Oddvar Arnesen

Trondheim, June 2016

SUMMARY

Remotely operated vehicles (ROVs) are used for an increasing amount of underwater applications, ranging from being hobby-related to having military and scientific interests. ROVs may incorporate automatic control strategies, making them capable of carrying out complex missions single-handedly, in a safe and efficient manner. This master's thesis describes the development and implementation of several systems and functionalities for a Videoray Pro 4 ROV. In the thesis, "Videoray Pro 4 ROV", "Videoray ROV", "the ROV" and the like may all be used interchangeable, and it should be clear from the text what it refers to.

The most significant scientific contribution made by this master's thesis is the presentation of experimental testing of successful underwater path-following for a Videoray Pro 4 ROV. The methodology is based on a set of predefined waypoints in 3-D space, and generates paths between each consecutive waypoint location. The ROV uses a technique called lookahead-based line-of-sight steering to follow the path segments and traverse the waypoints. When the distance to the current waypoint is within some *radius of acceptance*, a switching controller will command the ROV to follow the path leading to the next waypoint location.

The force produced by the propellers can easily be controlled with a *Dualshock 3* hand-held controller, i.e., the *user interaction unit*. Occasionally, there are problems with the hand-held controller freezing for about three seconds, due to a problem with the software managing this system. The issue seems to occur when the buffer storing the joystick input commands is filled up faster than it is being read. If the buffer is full, it will initialize a process that forces it to empty completely. The controller will then stop taking input, and will freeze at the final command received. This means that the thrusters will freeze at the last command received for about three seconds as well. When the thrusters freeze, if the previous command given was "0", the ROV will simply stand still. However, if the command was, say, "give 30% force in surge", the thrusters will give 30% thrust for about three seconds. This may have hazardous consequences to the ROV and nearby equipment. Attempts have been made to change the buffer size, but without luck.

It is difficult to make the ROV move with a very low velocity, due to physical limitations in the thruster system. In the later stages of the experimental work, the cause of the restrictions on the thrusters was identified. One of the cartridge seals has less oil than what is needed to properly lubricate the propeller shaft, meaning that the thrusters cannot produce forces below certain limits. The physical explanation to this is that the propeller is unable to overcome the static friction force in the shaft. Much oil is still left in the port-thruster cartridge seal, but this is not the case for the starboard thruster. An air-bubble indicates how much oil is left, and when the air bubble is 3/4 full, the cartridge seal should be replaced. Not only does this capsule ensure proper lubrication of the propeller shaft, but it also keeps water out of the pressure hull and prevents oxidation. The starboard cartridge seal has an air bubble that stretches throughout the whole capsule, meaning that it should have been replaced a while ago. Therefore, the Videoray ROV experiences difficulties when controlling the heading, and is bound to have a velocity greater than desired in order to move forward. This forces the ROV to move slightly beyond the desired location. The magnitude of this overshoot depends on the radius of acceptance and the ROV's velocity at the waypoint. Using a radius of acceptance equal to 10 cm, the ROV will move past the waypoint with about 5-10 cm. Replacing the existing cartridge seal with a new one will allow the ROV to move at lower speeds, implying better path-following performance.

The Qualisys Motion Tracking system has been used to receive position and orientation data for the ROV. This system captures a set of markers using *optical tracking technology*. Markers are attached to a rig, which is fastened to ROV's bottom side. A lot of time and experimental work was dedicated to make rig, markers and Qualisys work together to receive measurements consistently. Difficulties were experienced with receiving position data for an area large enough to perform path-following. Eventually, it was concluded that improvements had to be made for the rig. Extending arms were created and attached to the rig. In addition to adjusting the camera settings and turning off ceiling light to reduce light reflection, acceptable response from the Qualisys camera system was achieved. Position and orientation data could now be extracted at a reliable rate, but random data drop-outs for tenths of a second still occurred. The data loss that lasts for shorter durations than a second are handled by a simple algorithm, which uses the previously measured position data directly.

In the beginning of the thesis work, a lateral thruster was mounted on the ROV, and code was developed to make this thruster work. Unfortunately, it communicated in a different manner than the other thrusters, with a method called *round-robin* or 1/2 duplex. Lacking documentation and difficulties communicating with the manufacturers of this thruster prolonged simultaneous control of all thrusters. Contact was made with Andrew Goldstein, Director of Software Engineering, Videoray LCC, in the beginning of January. In the later stages of the thesis work, Andrew found the time to help out with the documentation, and all the thrusters can now be controlled in a satisfactory manner. Therefore, future work with the Videoray ROV can benefit from a fully actuated underwater vehicle. Along with the successful implementation of waypoint guidance and path-following, the ROV may in the future be used to carry out a great variety of underwater control objectives.

SAMMENDRAG

I dag benyttes fjernstyrte undervannsroboter (ROV, eng.: Remotely operated vehicle) til stadig fler applikasjoner, alt fra å være hobbyrelaterte til å være av militær og vitenskapelig art. Kontrollstrategier kan implementeres for at ROVen skal utføre en rekke operasjoner helt av seg selv og uten menneskelig innblanding, på måter som både er effektive og trygge. I denne masteroppgaven har flere systemer og funksjonaliteter blitt utviklet og implementert for en undervannsrobot ved navn Videoray Pro 4 ROV. Navnene "Videoray Pro 4 ROV", "Videoray ROV", "ROVen" og lignende vil alle bli brukt gjennom masteroppgaven, og det vil klart gå frem av teksten hva som menes med de ulike navnene.

Masteroppgavens hovedbidrag er et vitenskapelig bidrag, og består av vellykket eksperimentell testing av banefølging under vann for en Videoray Pro 4 ROV. Algoritmen baserer seg på at vi har et sett med punkter i bassenget som vi ønsker å besøke, og at baner genereres mellom disse punktene. ROVen bruker algoritmen til å kjøre gjennom punktene ved å følge de genererte banene. Når ROVens avstand til den ønskede posisjonen er innenfor en radius fra punktet, her kalt *akseptert radius*, vil algoritmen velge det neste punktet som ønsket posisjon. Årsaken til at vi ønsker at ROVen er innenfor en radius fra punktet er fordi ROVen ikke vil klare å treffe punktet helt nøyaktig. Det er i tillegg vanskelig å kontrollere ROVen til å kjøre på en veldig lav hastighet grunnet fysiske begrensninger i thrustersystemet. I slutten av arbeidet med masteroppgaven har årsaken til de fysiske begrensningene blitt oppdaget, og er grunnet mangel på olje i en cartridge seal (tetningspatron, eng.: cartridge seal) til å smøre propellskaftet . Dermed trenger undervannsroboten en større hastighet enn den vi ønsker for å kunne nå en bestemt posisjon. I det ROVen er innenfor den aksepterte radiusen vil farkosten kjøre noe forbi punktet, avhengig av størrelsesordenen til akseptert radius og hastigheten som holdes når akseptert radius nås. Ved å erstatte nåværende cartridge seal med en ny vil Videoray ROVen klare å følge banen enda bedre, da lavere hastigheter kan holdes inn mot punktet. I tillegg vil dette forbedre ytelsen til systemet når det forsøker å regulere kursen sin mot en ønsket kurs, da denne styres direkte av momentene som skapes av styrbord og babord propellthruster.

Kraften som gis av propellene kan enkelt styres med en Dualshock 3 kontroll. Enkelte ganger er det problemer med at kontroller fryser i omtrent tre sekunder. Dette gjør det risikabelt å kjøre ROVen med kontrollen. På den annen side, når ROVen kun kjører på kontrollsystemene som er blitt implementert oppfører den seg helt fint. Mer spesifikt virker det som at bufferet som lagrer kommandoene sendt fra kontrollen bygger seg opp og fylles raskt. Dersom bufferet ikke kan leses (og dermed tømmes) raskere enn det klarer å bygge seg opp, vil det til slutt bli fullt. Et fullt buffer vil sette igang en tvungen tømmeoperasjon. Kontrollen og thrusterene fryser da i en gitt tidsperiode, estimert til å ta ca. tre sekunder, ved siste input som ble mottatt. Dermed vil ROVen være ukontrollerbar i denne tidsperioden, noe som kan føre til at ROVen kræsjer i gulvet, veggen eller annet utstyr. Det er blitt forsøkt å endre størrelsen til bufferet og lese av bufferet oftere, og selv om det er observert forbedringer er ikke problemet helt løst. Av den grunn foreslås det at det nåværende, manuelle styringssystemet undersøkes grundigere og at det eventuelt erstattes med et annet system.

Qualisys sitt kamerasystem ble benyttet for å oppnå posisjonsmålinger i MC-lab. Systemet finnes både over og under vann i labben, og felles for de to er at de baserer seg å ta lynraske bilder av et sett med markører. Et objekt defineres ut fra et sett med markører, der posisjon og orientering estimeres ved å bruke hva Qualisys selv kaller *optical tracking technology*. For å gjenkjenne Videoray ROVen som et objekt måtte markører først plasseres på en rigg, og riggen ble deretter skrudd fast på undervannsrobotens underside. Mye tid og eksperimentelt arbeid gikk til å få rigg, markører og Qualisys til å fungere som ønsket. Det var store vanskeligheter med å få inn målinger så ofte som mulig, og målinger så ut til å falle ut helt vilkårlig over flere sekunder. Evaluering ble gjort underveis, og det ble konkludert med at forbedringer måtte utarbeides med riggen. Forlengere ble dermed laget til riggen, som gjorde at markørene ble plassert lengre ut fra ROVen. Etter å i tillegg ha justert kamerainnstillingene begynte Qualisys nå å gi ut målinger oftere. Enkelte målinger faller ut for tideler av et sekund, men dette håndteres av en enkel algoritme som baserer seg på å bruke forrige måling om den ble gjort for under ett sekund siden.

I starten av arbeidet ble den lateral thrusteren undersøkt, og kode ble utviklet for å kontrollere denne. Dessverre kommuniserer den på en annen måte enn både de to horisontale og den vertikale thrusteren. En spesiell kommunikasjonsmetode krevdes også for å bytte mellom å snakke med hver av propellsystemene. Grunnet begrenset dokumentasjon og vanskeligheter med å få hjelp fra utviklerene av Videoray-produktene, ble fokuset på å få thrusterene til å samkjøre nedjustert. Kontakt med Videoray-utvikler Andrew Goldstein har vært opprettholdt siden starten av oppgaven. Da slutten av perioden med masteroppgaven nærmet seg fikk endelig Andrew tid til å hjelpe med dokumentasjonen, og det ble da orden på thruster-systemet. Systemet kommuniserer helt perfekt og nøyaktig som ønsket. Fremtidig arbeid kan dermed benytte seg av en fullt aktuert Videoray undervannsrobot. Sammen med vellykket banefølging kan ROVen brukes til å utføre et utvidet mangfold av kontrolloppgaver.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AMOS	Centre for Autonomous Marine Operations and Systems
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
CHSI	Cubic Hermite Spline Interpolation
DOF	Degree of Freedom
DP	Dynamic Positioning
FF	Feed-Forward
FPP	Fixed-Pitch Propeller
FS	Fermat's Spiral
GNC	Guidance, Navigation and Control
GUI	Graphical User-Interface
IMU	Internal Measurement Unit
LOS	Line-Of-Sight
LQR	Linear Quadratic Regulator
MC-lab	Marine Cybernetics laboratory
MCS	Motion Control System
MSS	Marine Systems Simulator
NED	North-East-Down

NTNU	Norwegian University of Science and Technology (*Norwegian*: Norges Teknisk-Naturvitenskapelige Universitet)
NWU	North-West-Up
PID	Proportional-Integral-Derivative
QMC	Qualisys Motion Capture
QMT	Qualisys Motion Tracking
ROS	Robot Operating System
ROV	Remotely Operated Vehicle
SLAM	Simultaneous Localization and mapping
UAV	Unmanned Aerial Vehicle
UUV	Unmanned Underwater Vehicle
WADAM	Wave Analysis by Diffraction and Morison Theory
WP	Waypoint

CHAPTER 1 INTRODUCTION

1.1 Background and Motivation

The control of Remotely Operated Vehicles (ROVs) is, as the name suggests, performed remotely by a human operator, normally with the use of a controller with joysticks. For operational tasks with ROVs like picking up an item on the seabed or adjusting a valve, several human operators are required in the field. They are needed to manually control the ROV, to steer the manipulator arm(s), for maintenance, and so on. To perform these operations, a particular set of skills are required. For simply remotely controlling an ROV, the costs required for training and having an operator in the field are often high, but by developing a control system for the ROV, the operator can more easily operate the craft as desired. Basic and simple automatic control system features can be used to control for instance the depth and heading. Further on, more complex control techniques like tracking or path-following algorithms can be developed for the craft. These control strategies may be used in conjunction to perform a variety of tasks, without the need for human intervention.

Generally speaking, there should be a nice and easy way to manually steer the ROV. We wish to develop simple, automatic control strategies, where the overall goal is to increase the ROV's capabilities, while at the same time reducing the need for human intervention. In the end, a fully developed system will be *autonomous*, meaning that it is able to decide how to satisfy any feasible control objective in an optimal way. To begin with, the system should be able to move to specific locations and to reach and maintain predefined orientations, while still incorporating efficiency, safety and sustainability. Other reasons for developing automatic systems that exclude human intervention is to reduce the cost of having operators in the field. In addition, humans are often part of the system when errors are found. Computerized systems can be made "perfect" with proper development techniques, implementation and verification by testing. Creating such systems is what motivates the topics of this master's thesis, and especially for underwater robotic systems like ROVs. They are capable of performing an ever growing variety of tasks. Typical tasks assigned to ROVs today are related to advanced control techniques like dynamic positioning (DP), automatic tracking or path-following, exploration, underwater item localization and operation, and much more.

Today, there is a huge focus on unmanned underwater vehicles (UUVs), automatic robotic systems developed for underwater operations. These systems have a varying degree of automatic functionality installed. The world today sees a growing interest in developing autonomy, i.e., the robotic system's ability to use available information to make its own, optimal decisions for how to perform an operation. The ocean is vast, and the development of unmanned, automatic and autonomous marine vehicles makes it possible to carry out complex missions. In turn, this allows us to retrieve more information about the world that we live in, increasing and expanding our technological possibilities and limitations.

Several operations and applications can make great use of UUVs. Among those we find, for instance, exploration, fishing, cleaning and inspection of fishing nets and ships, cleaning oil spills, subsea mining, pipeline laying, inspection and maintenance of pipeline systems, and much more. To perform these tasks is often difficult and expensive, and may not even be done in a safe manner. This gives motivation for developing implementing and testing control system features on a real ROV.

Due to the general interest in the above-mentioned topics, this master's thesis has been dedicated to the development and implementation of a guidance, navigation and control system for a Videoray Pro 4 ROV. The goal is to implement a system that makes the ROV perform path-following and waypoint guidance in a local lab-facility, namely the MC-lab at NTNU. There is a big step from theory and simulations to experimental trials. Therefore, the aim is not to make a system that is overly complicated, but instead, to focus on making an actual system that can carry out the tasks in a satisfactory manner. Simulations are, however, a nice tool for displaying the expected behavior that can be used for comparison and verification of the experimental testing results.

In the following, an informal description of the Videoray Pro 4 will be introduced for the reader to understand the ROV's capabilities. The objectives, scope and limitations will be introduced next, before presenting the contributions made by this thesis. Finally, the thesis outline will be given in a clear and concise manner.

1.2 The Videoray Pro 4 ROV

The Videoray Pro 4 ROV is a small observation class ROV, incorporating the design and technology that makes it one of the most advanced, capable and versatile small ROVs on the market today. It is equipped with three thrusters, allowing movement in surge, heave and yaw, in addition to a lateral thruster for controlling sway motion. The craft weighs about 6.1 kg, but with the lateral thruster, the weight goes up to about 6.9 kg. Additional accessories can be mounted on the ROV as well, such as cameras, sonars or manipulator arms. Worth to mention is that the weight can be manually adjusted in the bottom of the ROV, in order to change the weight and buoyancy properties when equipping the ROV with extra equipment. The ROV is purchased from the Videoray company for research purposes, and all code except for the standard communications protocols, sample codes and interfaces are developed from scratch. Figure 1.1 depicts the Videoray ROV equipped with a lateral tunnel thruster, along with the controller that is used to manually operate the ROV.

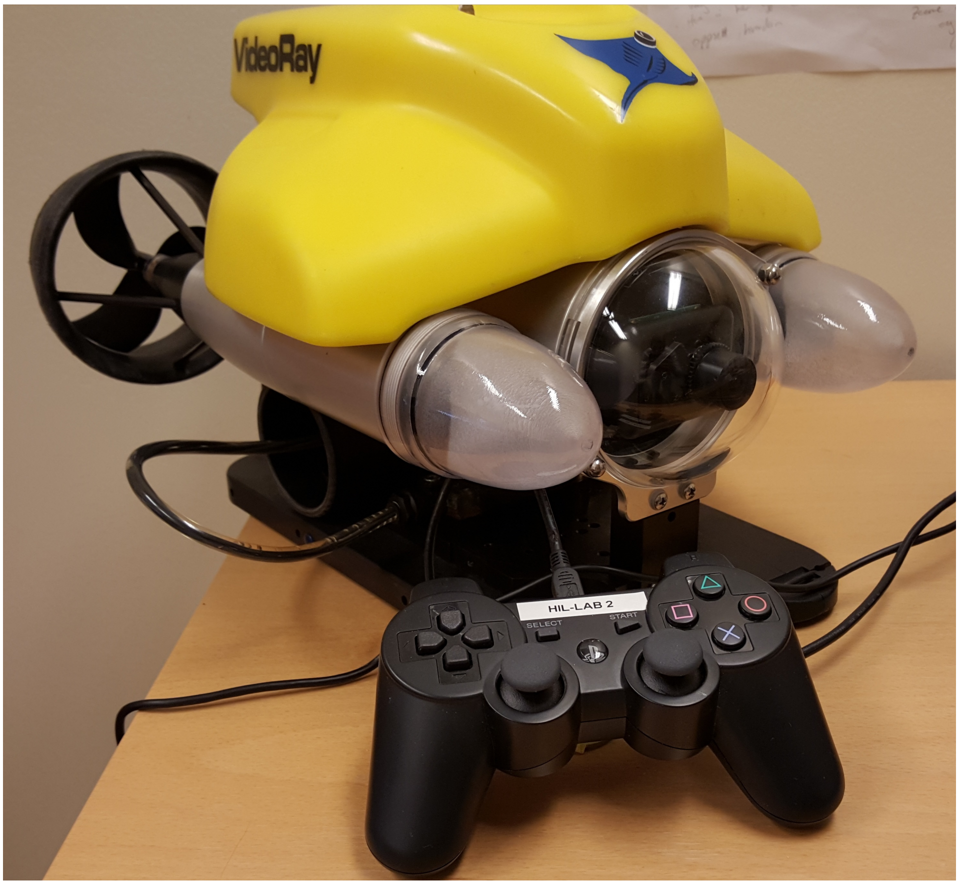

Figure 1.1: Videoray Pro 4 ROV and the Dualshock 3 (Sixaxis) controller

Description of the ROV and equipment

Table 1.1 below shows the specifications for the Videoray Pro 4 ROV, and summarizes some of the main hardware properties of the ROV. Not all the data about the ROV is available, for instance, information about sensor and thruster technology is well hidden (for more information, the reader is advised to visit the Videoray homepage). In addition, it should be mentioned that the lab computer used is a Dell latitude e7440.

Table 1.1: Videoray Pro 4 ROV - Table of specifications

Size (LxWxH)	37.5 x 28.9 x 22.3 cm
Weight	6.1 kg
Buoyancy properties	Positively buoyant in heave Neutrally stabilizable in roll and pitch
Max depth	305 m
Power input	100-240 VAC
Thrusters	Horizontal 2 x 12 VDC / 200W / 17A Vertical 1 x 12 VDC / 200W / 17A Lateral 1 x 12 VDC / 200W / 17A
Speed	Horizontal: 4.0 knots
Main camera	570 lines, 0.004 lux
Lights	2 x Optimized LED Arrays 3,600 lumens
Integrated sensors	3D-Tilt Compensated Compass Leak Indicator System Voltage Accelerometer Water Temperature Depth Sensor MEMS Gyro Internal Temperature

1.2.1 Software and Programming Environment

The code files that operate the joystick and the Videoray ROV are written in C/C++, and have been augmented with an interface to grant Matlab access and use. This interface is called a *mex-function*. Mex compiles source files and links them into a shared library called a mex-file using a C/C++ compiler. These mex-files are built for standalone Matlab engine applications, and act as Matlab object files. Communication flow is depicted in the connection scheme in Figure 1.2. It shows how the operator communicates with the computer system, and how the computer sends and receives data to/from the ROV. The computer system contains an initialization block, which is run when the software is initialized. Matlab commands are sent to the Videoray Pro 4 ROV block, which operates the ROV thrusters and handles sensor data, sent back to Matlab for further processing. All code files can be found in Appendix C.

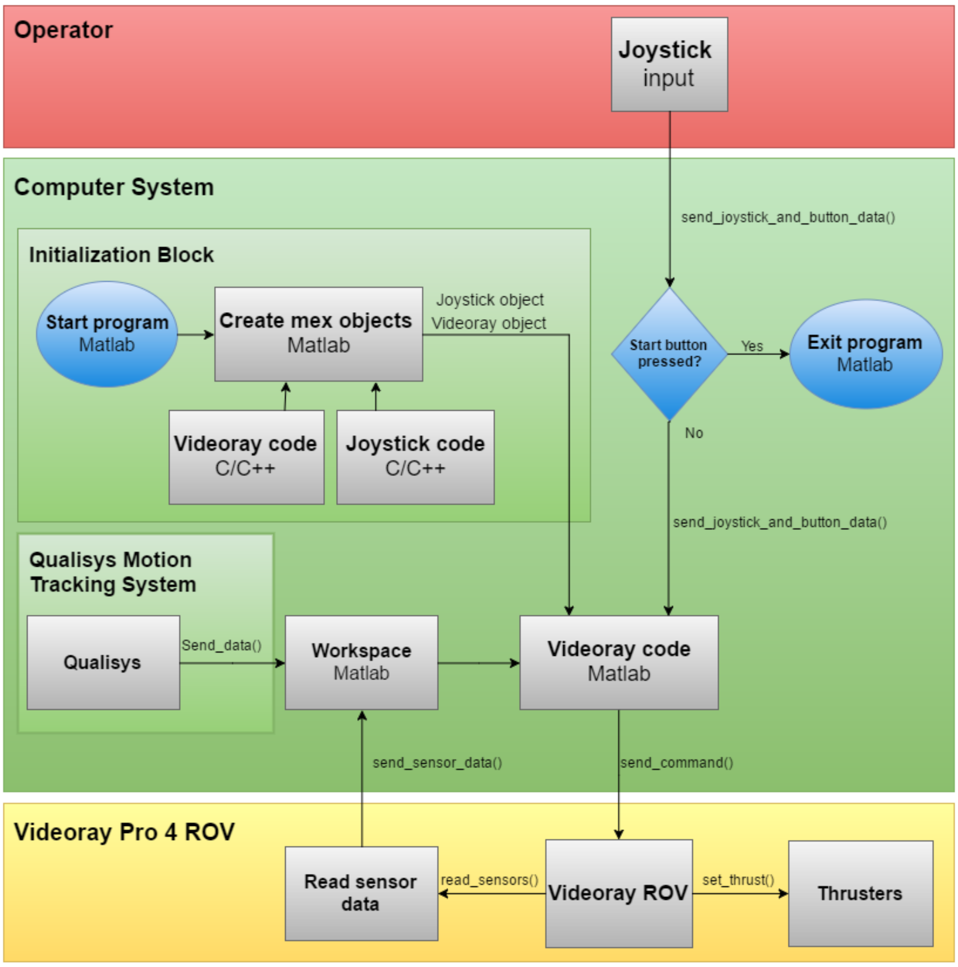

Figure 1.2: Flowchart: System connections and data flow

1.2.2 Controlling the ROV

The user interaction unit is a hand-held controller, and is used to steer and give orders to the Videoray Pro 4 ROV. It is called the SONY *Dualshock 3* or *sixaxis*, also known as the controller used for the Playstation 3 console. For the Videoray Pro 4 ROV, the controller buttons and joysticks have been mapped as in Figure 1.3 below

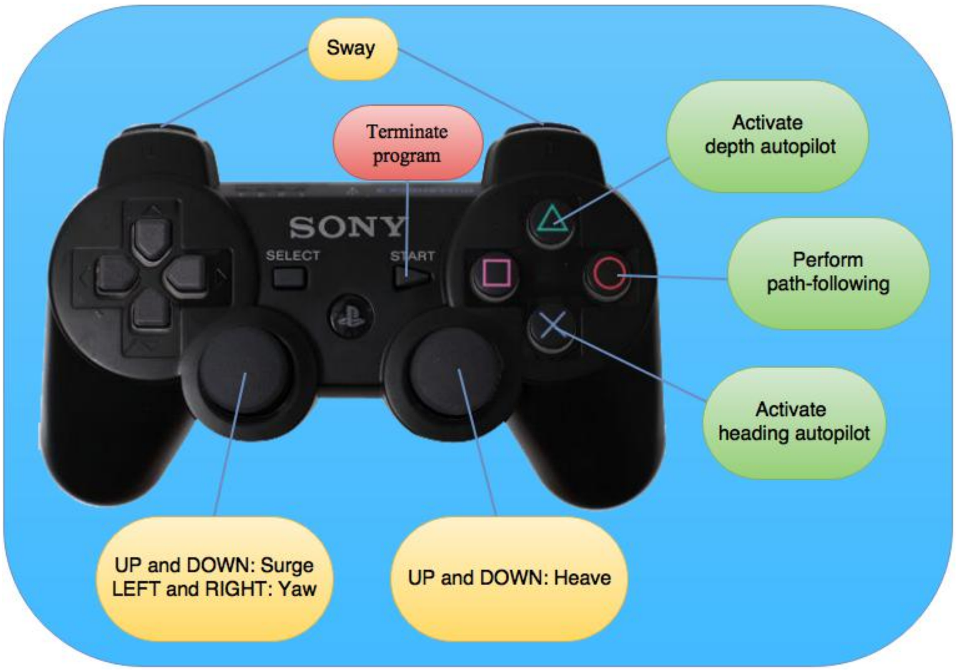

Figure 1.3: The SONY Dualshock 3 (sixaxis) hand-held controller

Automatic control features have been created for the ROV using the hand-held controller, seen in Figure 1.3. The ROV can be moved in surge, sway, heave and in yaw by controlling the joysticks directly. In addition, an arbitrary, desired depth can be kept by pressing the triangle-button. Now, users can manually steer the ROV while the depth is automatically maintained. The same can be done for heading as well, by pressing the cross-button. In this way, the ROV will keep a steady heading, which may be highly advantageous when, for instance, inspecting fishing nets or during investigation within or along offshore pipeline systems. Pressing the circle-button will activate path-following.

The control inputs given to the thrusters are represented by values between -100 and +100 (-1000 and +1000 for the lateral thruster). In order to control both heading and surge motions, the following mathematical relationship has proven to work very well

$$\text{Port thruster input} = \text{Surge joystick input} + \text{Yaw joystick input} \tag{1.1}$$
$$\text{Starboard thruster input} = \text{Surge joystick input} - \text{Yaw joystick input} \tag{1.2}$$

1.2.3 Propulsion System

The Videoray Pro 4 ROV has four thrusters in total. Two of the these let the ROV move in surge and yaw, while the third thruster moves the vehicle in heave. Additionally, a lateral tunnel thruster is mounted on the ROV's bottom side, enabling motions in sway. All the thruster motors are brushless and have counter propeller rotation. They are of the variable-speed fixed-pitch type propellers (FPP), and are placed so that the vehicle turning radius is close. This gives the ROV extremely agile maneuverability properties, and serves as a stable platform for both the camera and sensors. All the thrusters react quickly and respond accurately to the given input, but suffer from one particular flaw. The thruster force produced is zero when the commanded thrust is within certain limits. This is referred to as the thruster *dead-band*, and is considered a non-linear effect that may prevent the ROV from reaching a desired position or orientation. It seems that the dead-band value for each thruster changes over time. This suggests that some parts are subject to wear and tear. Increasing operational time seems to be related to the changing dead-band factor. The effect is discussed in greater detailed in Chapter 6.

Thruster Configuration

For the Videoray ROV, the conceptual description of the thruster configuration is depicted in Figure 1.4. The thrusters are represented by the numbers 1 to 4, where the encircled star refers to the heave thruster.

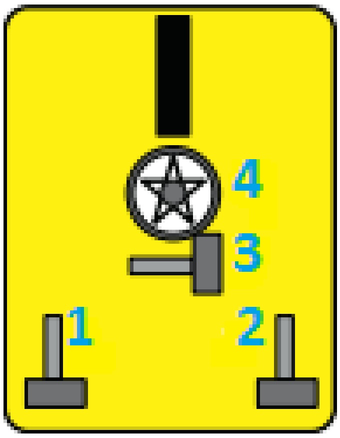

Figure 1.4: Videoray PRO 4 thruster configuration (Christ and Wernli, Dec 2013)

Moreover, because of how mass is distributed, thrusters are placed and how the ROV is designed in terms of geometrical properties, the Videoray can turn on the spot. Turning curvatures are therefore completely avoided. This is a typical problem of larger-sized vessels, as they often need to move in a big radius in order to turn. These properties contribute to making the Videoray Pro 4 an agile, versatile and highly capable ROV.

Main thrusters

The thrusters "1" and "2" in Figure 1.4 are mounted in the aft of the hull, seen in Figure 1.5, and produce surge force and yaw momentum. The propellers receive an integer value between -100 (max thrust reverse) and 100 (max thrust forward), where a combination of two different thruster values in "1" and "2" is used to control the heading. The thrusters have an in-built "ramp-up" function towards the input force. This means that, e.g., when a commanded thruster input of 30% of maximum power is given, the thrusters will ramp up steadily over a small time period towards this value. In more theoretical terms, this can be seen as a low-pass filter on the input step response, and serves as a method for reducing wear and tear on the thrusters.

If we desire to mount the ROV with additional equipment on the ROV, the main propellers may no longer be stationed in the craft's center of gravity. This will result in the vehicle pitching when producing thrust in surge. The higher the surge force, the more the vehicle will pitch. Maximum thrust generated by the main propellers is estimated to be 4.8 kg force for each thruster. According to the producers, these thrusters have a very high thrust-to-drag ratio (2:1) and thrust-to-vehicle weight (10 kg force thrust on a 6 kg vehicle), but these ratios do become a little weaker with the lateral thruster configuration (10 kg force on a 6 kg vehicle, and some additional damping in surge).

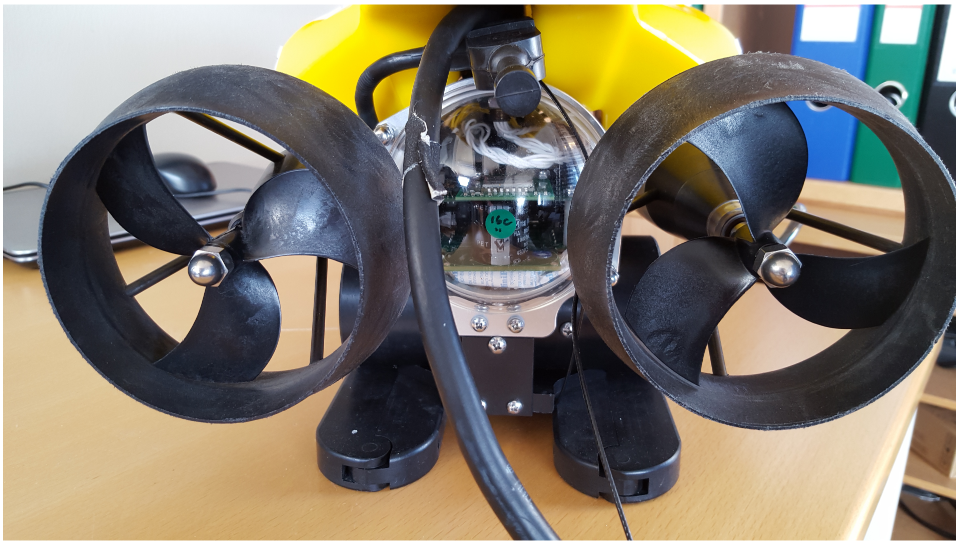

Figure 1.5: Videoray PRO 4 thruster: Main thrusters

Lateral thruster

The lateral thruster refers to thruster "3" in Figure 1.4, and can be seen in Figure 1.6. Different from the other thrusters, it takes a value between -1000 and +1000. It is a transverse tunnel thruster that goes "through the hull" of the craft, mounted inside a tube and producing a sideways force. The distance between the ROV and its center of gravity is small, meaning that the roll momentum generated by this thruster will be very small. In addition, the momentum generated in yaw by the lateral thruster is small, and should not be high enough to alter the vehicle's heading. However, by adding this thruster to the ROV, the weight of the vehicle is increased by approximately 800 grams and the center of gravity is lowered.

This thruster was not part of the standard package when the ROV was purchased. It is an additional equipment to the standard set, and code that allows for communication with this thruster had to be developed. The main issue with the thruster is that it communicates differently from the other thrusters. It communicates in a *round-robin fashion*, also referred to as 1/2 duplex, which may be compared to how communication takes place with walkie-talkies. This means that in order to give commands to the lateral thruster, only one command can exist on the line (tether) at a time. Before a new command can be sent, an appropriate response must be received. In the beginning of the thesis work, communication with each of the thrusters had been achieved. However, simultaneous communication did not work. The documentation on this thruster is scarce, and work with the lateral thruster had to be prolonged. Contact with the Videoray developers was made early on, but proper help was not received until the end of the thesis work. This made it possible to finish the work with the thruster and to receive appropriate control. Due to the delayed response from the developers, the lateral thruster has not been included in experiments or simulations.

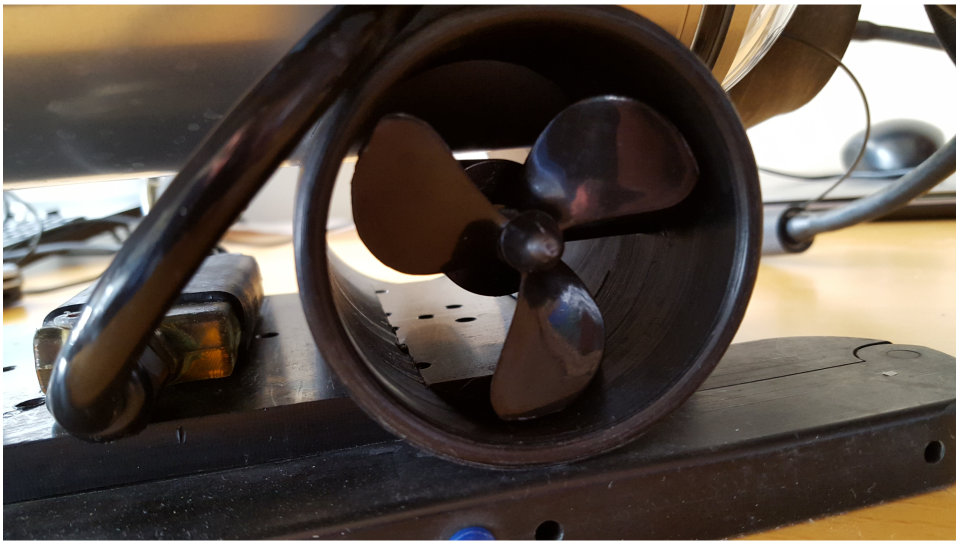

Figure 1.6: Videoray PRO 4 thruster: Lateral thruster

Vertical thruster

The vertical thruster in Figure 1.7 refers to thruster "4" (the encircled star) in Figure 1.4, and takes a value between -100 and 100. The Videoray Pro 4 ROV's vertical thruster lets the ROV ascent and descend quickly, but may also produce a small yawing momentum. For high vertical thrust values, running a heading controller for the main propellers in cascade with a depth controller will ensure steady heading during depth control. One reason why this issue may be considered is because the cable is likely to start twisting, which can result in an ROV entangled in its own umbilical. Fortunately, when the distance to be covered by the heave propeller is small, this will not be an issue.

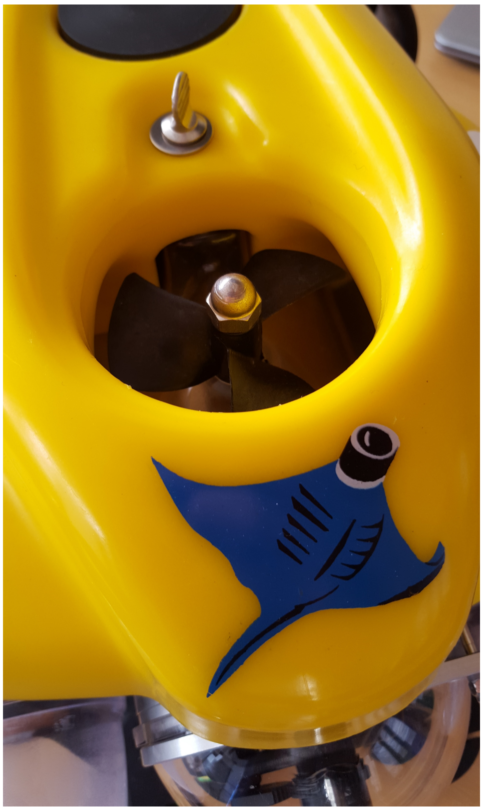

Figure 1.7: Videoray PRO 4 thruster: Vertical thruster

1.3 Objectives, Scope and Limitations

The main objectives and contributions of this master's thesis are summarized below

1. Study the ROV's lateral thruster and develop code that achieves control of this thruster

2. Set up the Qualisys Motion Tracking (QMT) system to receive information related to the ROV's position and orientation in Matlab

3. Review the literature for motion control systems, typical tracking and path-following systems and mathematical modeling of ROVs. The mathematical model of the ROV aims to mimic the properties of the Videoray Pro 4 ROV, to be used in simulations and to show how the craft is likely to behave when attempting to satisfy a set of control objectives. The simulation model should incorporate a neat and intuitive design, and may serve as a platform where further development and testing of functionality for the Videoray ROV can take place.

4. The simulation model of the Videoray Pro 4 ROV should be created using Matlab and Simulink

5. Develop, implement and perform experimental testing of autopilots for heading and depth for both the simulation model and the Videoray ROV.

6. Develop, implement and perform experimental testing of a path-following system for both the simulation model and the Videoray ROV.

7. All experimental testing will be conducted in a local test facility, i.e., the Marine Cybernetics laboratory (MC-lab) at NTNU, Trondheim.

1.4 Contributions

The following contributions are made in this thesis

The scientific contribution, and the main contribution of this thesis, is the development and implementation of a motion control system and successful path-following for a Videoray Pro 4 ROV. A path is created based on a set of predefined waypoints, both on and below the surface, by the guidance system. The navigation system retrieves position data by the Qualisys Motion Tracking system in a test facility, i.e., the MC-Lab located at NTNU, Trondheim. Lookahead-based line-of-sight steering using heading, depth and surge controllers as the control system are responsible for following paths in a nice manner. The underwater path-following has been successfully implemented and verified by experimental testing.

Other contributions are concluded by further development of functionality for the Videoray Pro 4 ROV, listed below

- A simulation model of the Videoray Pro 4 ROV has been created, along with implemented control features to automatically control depth, heading and follow predefined paths. The model has been created in Matlab and Simulink, and is well-suited for further development, testing and verification of functionality.
- Files have been created in Matlab for receiving Qualisys position data. These files are made simple and in a modular fashion, where changes can be made for optimization based on application.
- Code for controlling a new thruster, i.e., the lateral thruster, has been developed. This makes the Videoray Pro 4 ROV fully actuated. Both ROV and hand-held controller code is written C/C++. An interface built on mex-function has been further extended, allowing the complete system to be run by Matlab
- Automatic controllers for maintaining depth and heading of the ROV have been developed. These features are available on the hand-held interaction unit by simply pressing a button. The automatic controllers put ease on the operator of the ROV, and can easily be used in conjunction with more advanced control strategies due to the modular design

1.5 Thesis Outline

The rest of this thesis is structured as follows:

Chapter 2 is dedicated to a literature review. Light will be shed upon the main research, methods and advances in the field of motion control systems for ROVs, path-following and tracking systems. Mathematical modeling of ROVs similar to the Videoray Pro 4 will be covered, as relevant to the development of a simulation model.

Chapter 3 aims to show how the methodologies from the literature review are implemented. It will describe the guidance, navigation and control (GNC) system that has been developed for the Videoray Pro 4 ROV. The thrust allocation system will be reviewed briefly.

Chapter 4 presents results from testing of the simulation model. The testing procedure is divided into several test cases. Some reasoning behind choosing appropriate parameters for the simulation model of the Videoray Pro 4 ROV will be given. Concluding remarks from the simulation results will be presented.

Chapter 5 studies test cases similar to those for the simulations, and presents experimental testing with the Videoray Pro 4 ROV. Results from underwater waypoint guidance, including automatically controlling depth and heading, are emphasized. The response achieved when using the Qualisys camera system for underwater positioning will be presented as well.

Chapter 6 discusses the results from testing the Qualisys underwater camera system, simulation model and experiments with the Videoray ROV in the MC-lab. A comparison between simulation and experimental results will be given.

Chapter 7 draws conclusions and proposes further work with the Videoray Pro 4 ROV.

CHAPTER 2 MARINE CONTROL SYSTEMS

This part of the thesis is dedicated to review some of the literature and recent research in the field of motion control systems, tracking, path-following and mathematical modeling of unmanned underwater vehicles (UUV). More specifically, this will be related to ROVs similar to the Videoray Pro 4 ROV. Focus will not be on studying the complete marine control system, which consists of power systems and electronics, thruster control, and more. Instead, the spotlight is put on the parts of the system that more directly deal with controlling the marine craft and with satisfying specific control objectives.

The topics highlighted in the literature review are the motion control system, with a closer look at the guidance system, how to track a reference and how to perform path-following for ROVs. Mathematical modeling covers the theory and recent research behind the model for developing a model for the Videoray Pro 4 ROV. Furthermore, when discussing marine control systems in today's context, it is impossible to omit autonomy. Therefore, some emphazis is put on autonomy and autonomous systems.

2.1 Motion Control Systems for Marine Crafts

More than a hundred years has passed since the first automatic control system for ship steering was developed in 1911, better known as a proportional-integral (PI) controller, by Elmer Sperry (Bennett, 1996). The interest in marine control systems has grown even greater in recent times, where marine vehicles can perform a larger specter of tasks. Among others, these tasks involve path-following, dynamical positioning and automatic execution of small missions, e.g., seabed mapping, state estimation, etc. One of the subsystems that contributes to a big part of the marine control system is the *motion control system* (MCS). Marine vehicles' motion control systems may all vary when it comes to accuracy and complexity, depending on dynamical properties and intended operations workspace.

2.1.1 Guidance, Navigation and Control

According to Fossen (2011), the MCS of a marine craft is responsible for dealing with all of the craft's motion based properties. This includes calculating the forces that act upon the vehicle and involves processing of all the information required to satisfy specific *control objectives*. These objectives are for instance to maintain or move to a desired position with a predefined orientation. In order to develop such a system, it is highly convenient to divide the motion control system into three separate parts.

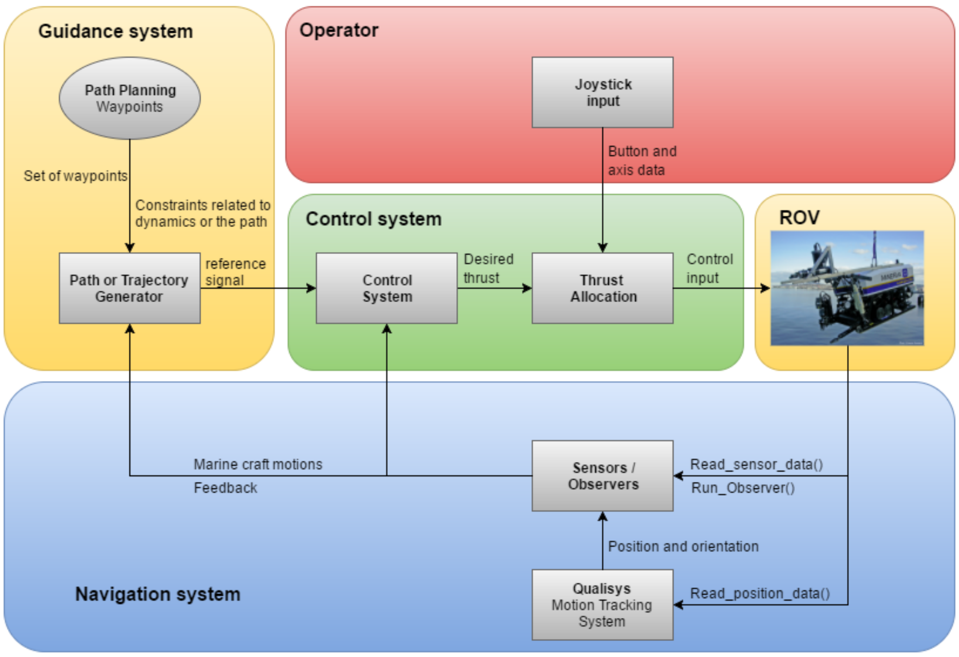

Figure 2.1: Closed-loop guidance system

The three subsystems that make up the motion control system are known as the *guidance*, *navigation* and *control* systems, and they are often referred to as the marine craft's *GNC* system. The guidance, navigation and control modules communicate through data transmission as illustrated in Figure 2.1. It depicts a typical data transmission flow for an ROV with trajectory tracking functionality. Please note that for a craft with a different workspace, the GNC system blocks may be coupled and represented differently than what is shown in Figure 2.1. In general, a tighter coupling often gives less modularity, while a more coupled GNC system yields higher performance (Fossen, 2011).

Lekkas (2014) explains that it is advantageous to adopt a modular approach and look at the GNC system as three modules making up a marine craft's motion control system. A *path planning* module should be part of this system, because all the modules interact with each other and none of them are completely independent. Of course, this may be difficult to avoid as the robotic systems grow in complexity and are more tightly coupled with the increase in advanced functionality. One may see this more easily today, as there is an ever growing focus on autonomy in robotics. The path planning or the *motion planning* module (Aghili (2012); From et al. (2010)) is responsible for planning a path for and/or with the guidance system (see Lekkas (2014), Section 1.1). Lekkas (2014) continues with describing how this module often includes both designing a suitable path and actions to be taken for the robot to accomplish the mission. With an increase in the level of autonomy in robotic systems, it is intuitive that path planning should be done autonomously and online by the craft for an optimized path planning module. Giving the system a predefined path will rarely include the ever updating information about the robot's surroundings. This is still a big challenge, as implementing autonomy and intelligent decision-making in robotic systems is a predominant difficulty (see for instance Blouin et al. (2015) and Vassev and Hinchey (2013), among others).

In the following, a description of the guidance, navigation and control systems will be given, based on the work by Fossen (2011), particularly in Chapter 9. Next, the focus will be on path planning, where a substantial amount of information is drawn from the work of Lekkas (2014).

Navigation System: The navigation system is comprised of sensor devices for measuring a system's state, i.e., distance traveled, position, velocity, and so on. Where sensor measurements are not available, for instance due to high cost or because no sensor exist for measuring the specific state, *observers* are used for *state estimation*. Observers are computerized systems that use a model of the craft (high/low fidelity control plant) in order to estimate the model's state (Sørensen, 2012). Often, observers and sensors are used in conjunction, yielding more accurate and stable results. Navigation systems may, for example, let us know our position within a specific reference frame, which is highly valuable information.

Guidance System: The guidance system creates a reference signal for the control system by taking information from the path planning module and feedback from the navigation system. Guidance systems should aim on creating optimized paths, avoid obstacles along the paths and could also include other specified requirements. Such requirements may be, e.g., minimize fuel consumption or time usage, re-plan the path in case of bad weather or coming storms, etc.). Next, the path and any constrains specified are sent to the control system as a trajectory signal.

In addition, the guidance systems may be classified as either a *closed-loop* or an *open-loop* guidance system (Fossen, 2011; Balchen et al., 2003). If sensor data and observer estimates are available through a feedback to the guidance system, the guidance system is a closed-loop guidance system. These systems usually perform better than their counterpart, as the feedback information can be used to achieve the desired response faster. If the system does not have a feedback loop back to the guidance system, it is classified as an open-loop guidance system. This means that already known/estimated information about the environment may be given as a feed-forward (FF) with techniques for automatic and dynamic compensation. Figure 2.1 shows a closed-loop guidance system with feedback, while Figure 2.2 displays an open-loop guidance system where known information about the wind, waves and current are used as feed-forward. Please note that sensor data and observer estimates may still be used as feedback to the control system (and not the guidance system) for the system to be an open-loop guidance system. This is called an internal closed-loop in the control system, and the guidance system will still be open-loop.

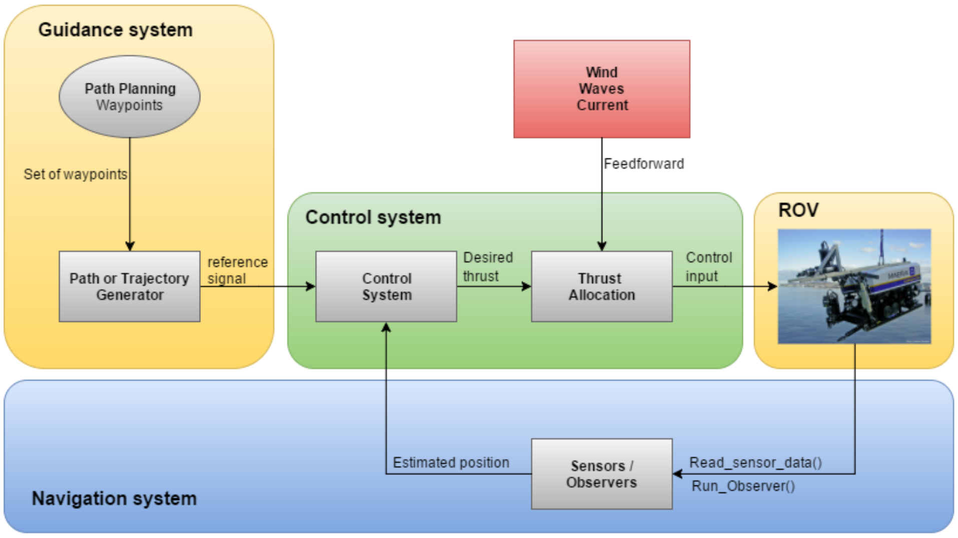

Figure 2.2: Open-loop guidance and closed-loop control system with feedforward

Control System A control system refers to a computer controlled system, which often manages or regulates one or several devices to achieve some desired behavior. Just like guidance systems, control systems are commonly divided into two classes: open-loop and closed-loop control systems. Both systems generate output depending on the input, but closed-loop systems feed back the response in an inner loop for further computation and correction, while open-loop systems do not. This means that closed-loop control systems require that an output of the system can be measured (or estimated). Most systems have sensors and/or observers, meaning that several states can be either measured or estimated with little error. This encourages the use of closed-loop systems. Conventionally, a control system may both contain feed-forward and feedback techniques, hence being classified as a closed-loop system. A system containing reference feed-forward only is classified as an open-loop system.

2.1.2 Path Planning

Path planning is a research field that has been studied broadly, and a considerable number of contributions have been dedicated to this topic. As the name suggests, path planning is concerned with planning a path that can be designated to and followed by a vehicle. It is of great importance that the paths satisfy properties related to the environment where the vehicle will travel, also considering the vehicle's dynamical limitations. In addition, obstacle avoidance can be incorporated in the path planning system, but not all applications may require this. According to Lekkas (2014), the path planning process includes two main steps. Firstly, the path planning system must be able to determine a set of points on a map, i.e. the *waypoints*, and secondly, to generate a path based on these waypoints. Other terms are used for path planning as well, for instance, "motion planning" in robotics. A conceptual description of how a simple path-planning system behaves is depicted in Figure 2.3. The path is generated based on data from weather forecasts and information about difficult terrain.

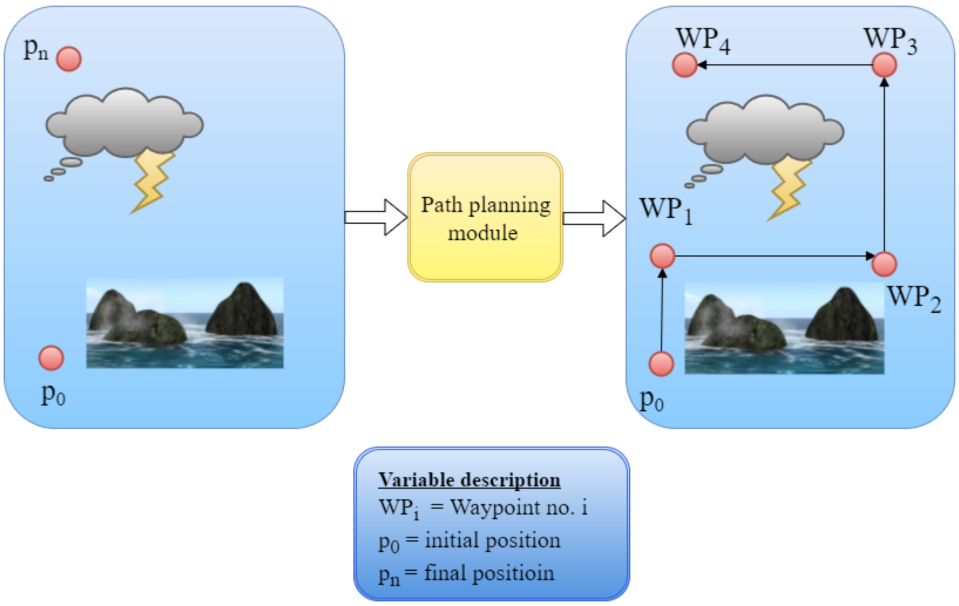

Figure 2.3: Generating path based on information about weather and difficult terrain

Often we desire that the paths satisfy a specific set of criteria. For autonomous systems, the robot should be able to evaluate these criteria online, but predefined paths based on a set of waypoint locations can be evaluated on beforehand. It is convenient that certain path specifications are met in order to solve the control objective efficiently. The path evaluation criteria can be summarized (Dahl, 2013) as:

- **Path parametric smoothness:** Describes the speed of the curve that will be traced. It requires that the parametric derivatives exist and are continuous of all orders, but in most practical applications, only a certain degree of smoothness is required (Dahl, 2013). This means that the path must be connected, implying that no spaces can be calculated as part of a path. In guidance terms, the path consists of several subpaths, and it is required that all of the subpaths are connected. Additionally, other continuity requirements can be formulated, like velocity and acceleration continuity. Mathematically, this is equivalent to a continuous derivative of the associated order. What that means is that in order to formulate velocity constraints, it is required that the derivative of the function describing the path is continuous. For acceleration, this function must be continuous after derivating twice, and so on.

- **Path geometric smoothness:** Defines the geometric behavior of the path. The speed at which the curve is traced may not be relevant as long as the path is geometrically smooth, and the geometric smoothness is a more applicable criterion to the path. Just as for the parametric smoothness criterion, the geometric smoothness criterion will be dependent on the application. A nice feature of the geometric smoothness criterion is that geometric continuity of the path is ensured if the path is parametrically continuous.

  These two criteria are important for marine underwater vehicles. The course of the craft is controlled continuously, and the path must also be continuous. If the path is discontinuous, the craft will fail to control its course towards the desired course. Geometric continuity is therefore important. Furthermore, if desired velocity or acceleration properties along the path are set, the velocity and acceleration continuity become equally important. Please note that even though fixed point rotation may be possible, it is still impossible for the vehicle to change its course in zero time, which is why the course angle cannot be discontinuous in time.

- **Path length:** The distance that the vessel will travel if the path is followed perfectly. It is known that between two points, the shortest path is a straight line joining the points. However, for some marine vessels, just following this shortest path is not possible due to the vessel's dynamics. For vehicles that are not this dynamically restricted, let's say for a vehicle that can change its heading while keeping its position, following the straight lines that connect the waypoints is a fully manageable task. One interesting thing to mention here is that when the vehicle is affected by a force field, i.e. under influence by external forces like wind and currents, the geometrically shortest path is not necessarily the shortest path in terms of distance traveled (Zermelo, 1931).

- **Path allowance:** Departure from the piecewise linear path, or how much the marine craft is allowed to deviate from the path or the desired waypoint (Lekkas, 2014). Some allowance may be needed to adjust for the dynamics of the craft, and the allowance must be known if the mission is to perform path following or path tracking when obstacles are present.

- **Tractability:** A measurement of how the path can be locally changed, or how easily the path can be managed. This is especially important for autonomous systems. Lekkas (2014) states that the tractability property mainly takes on two issues. Firstly, the *practicality* of the shape of the path. Between two waypoints, having a proper geometry is important. The path may have unnecessary zig-zagging or wiggling, and may still satisfy the geometric or parametric smoothness or continuity constraint, but will be difficult for the craft to follow. Secondly, it is a measurement of the effect of adding, removing or changing waypoints. It impacts all of the above criteria listed, and may affect the path either locally, globally or partially locally and globally. Furthermore, Lekkas (2014) proposes that using the correct interpolation method may be crucial and provides some suggested guidelines for the three different cases. The Cubic Hermite Spline Interpolation (CHSI) technique has shown to perform well for local waypoint control. For partial control, the monotone CHSI method is well-suited, while natural cubic splines are appropriate for global interpolation.

- **Algorithm complexity:** The time it takes to perform the path planning procedure is dependent on both hardware and software properties. Run time is governed by necessary inputs, number of operations and how fast these operations can be executed. For a path being created offline, i.e., the path is entirely constructed before the information is given to the marine craft, the computational time is not critical. This is also true for online simulations for large vessels, where shortage of computational power often is rare. However for smaller crafts, computational power may be limited, and hence, the algorithm complexity is crucial. Software solutions must be both effective and intuitive, as maintenance will be required and more than one person is likely to read the code. Therefore, developed and implemented code and software should be properly documented, so that other developers easily can understand and modify as required.

How to generate a path out of a set of waypoints and what smoothing effects to apply to the path will depend on the application, vehicle dynamics and the environment in which the vehicle will travel. The formulation of the path planning problem is addressed by Tsourdos et al. (2010), however for unmanned aerial vehicles (UAVs), but with several similarities related to marine applications. One way to generate and optimize a smooth path from a set of waypoints in 3-D space is performed by C. L. Bottasso and Savini (2008), which was done by combining the work of Anderson et al. (2005) and Frazzolini and Dahleh (2002). They state that there are three general approaches to planning the motion, namely, the *cell decomposition*, *roadmap* and *artificial potential field* methods.

A system's *configuration space* is defined as the space of possible positions and orientations (or configurations) that may be realized by the system (Fossen, 2011). The paper by C. L. Bottasso and Savini (2008) shortly explains that many motion planning algorithms rely on using the robot's configuration space, before it presents a concise description of the three approaches. Cell decomposition methods starts with partitioning the configuration space into a finite number of regions and to find collision-free paths for each region. Next, the problem is to find a sequence of neighboring cells that include the initial and final condition. For roadmap methods, the idea is to construct a network of collision-free connecting paths that span the configuration space. Then, the problem is reduced to finding the paths that connect the initial and final configuration to the roadmap. After that, an appropriate sequence of paths must be selected so that the previous problem is solved. Building this roadmap can be performed by, for instance, visibility graphs and voronoi diagrams, to mention only a few methods (Dahl, 2013). When using artificial potential field methods, obstacles are modeled as potential fields by using gradients, and optimization techniques are used to find the fastest/shortest path through this field. The method relies on solving an optimization problem, and additional requirements and criteria can be added to the system, e.g., currents and winds over specific regions, speed limitations, and so on. While modeling the potential field is difficult enough (Dahl, 2013), solving the optimization problem offers several challenges to even the bravest solver. One frequently visited issue is the existence of a local minimum. If the solver finds this as the optimal solution, the robot may find itself trapped before reaching the path endpoint.

Some methods are based on using the Dubin's paths, clothoids and Fermat's spiral (FS) path planning strategy, where the latter also considers obstacle avoidance (M. Candeloro and Fossen, 2013). Lekkas (2014) states that FS has no initial curvature, meaning that it is very well suited for combining with both straight lines and other FS arcs. This implies that curvature-continuous paths can be generated and implemented in path-following and tracking applications. Another method that has shown great simulation and experimental results are presented in Skjetne (2005). The technique parametrizes line and arc segments before they are concatenated to a full path. However, as mentioned in Chapter 10.4 by Fossen (2011), the method suffers from the drawback that the path must be known, calculated and parametrized in advance. This may not always be very practical, or even possible. Instead, simpler paths consisting of waypoints and straight lines often have to be used.

In Chapter 4, Lekkas (2014) discusses a path generation technique using the CHSI spline interpolation method. It preserves the *monoticity* property of the path, which means that it is entirely increasing (or decreasing) until it stops, while at the same time giving realistic solutions. Further on, it is desired that the path is *practical*, meaning that all unnecessary wiggling can be avoided. While the monotone CHSI technique has curvature discontinues at the waypoint locations, it gives very practical paths that are almost impossible to get with conventional splines (Lekkas, 2014). The methodology developed by Fritsch and Carlson (1980) is proposed as a highly efficient spline-based methodology for generating paths. It demands low computational power and creates very convenient shapes. The drawback of the technique is the curvature discontinuity that induces a cross-track error on the locations of the waypoints. However, as Lekkas (2014) explains and proves in Chapter 9, it does not prevent the method from being used successfully with guidance laws like the *Line-Of-Sight* (LOS) guidance law.

2.2 Guidance System Design

According to Fossen (2011), there are three different scenarios related to motion control. Ordered from the simplest to the most difficult control objective to satisfy, the three objectives can be classified as *setpoint regulation*, *path following* and *trajectory tracking*. In setpoint regulation, the desired position and attitude are constant, and no actual path is created. Path-following involves following a predefined path, without any restrictions on following the path or reaching the desired location(s) within a specific time limit. Finally, in trajectory tracking the goal is to follow a reference signal that is time-varying, meaning that there is a spatial and a temporal constraint on the path. What these systems often have in common, is that the desired location or set of locations can be defined using *waypoints*. All systems that deal with waypoint guidance benefit from a waypoint-generator. That is because waypoints are stored in a waypoint database that is designed for path or trajectory generation. Additionally, weather routing, obstacle avoidance and mission planning can be implemented when designing waypoint guidance systems. In 3-D space, the set of waypoints may be represented in Cartesian coordinates as

$$WP.pos = \{(x_0, y_0, z_0), (x_1, y_1, z_1), ..., (x_n, y_n, z_n)\} \tag{2.1}$$

Moreover, speed, heading and radius of acceptance for the different waypoints may be specified in a similar manner. While there are two popular methods when it comes to LOS guidance, namely *lookahead*-based and *enclosure*-based steering, only lookahead-based steering will be considered. This is because the enclosure based method is more computationally inefficient, in addition to not being valid for all types of cross-track errors. For the lookahead-based LOS methodology, the first step is to calculate the course angle, $\chi_d(e)$, which can be written as

$$\chi_d(e) = \alpha_k + \chi_r(e) \tag{2.2}$$

where $\chi_r(e)$ is given by

$$\chi_r(e) = arctan(\frac{-e}{\Delta}) \tag{2.3}$$

Here, e is the cross-track error (normal to the path), Δ is the lookahead distance and α_k can be written as

$$\alpha_k = atan2(y_{k+1} - y_k, y_{k+1} - x_k) \tag{2.4}$$

The steering law given by $\chi_r(e) = arctan(\frac{-e}{\Delta})$ can also be interpreted as a saturating control law, i.e.

$$\chi_r(e) = arctan(\frac{-e}{\Delta}) \iff \chi_r(e) = arctan(-K_p e) \tag{2.5}$$

where the gain K_p is given as $K_p = 1/\Delta > 0)$.
With the desired heading angle given in a current-free environment as

$$\psi_d = \chi_d \tag{2.6}$$

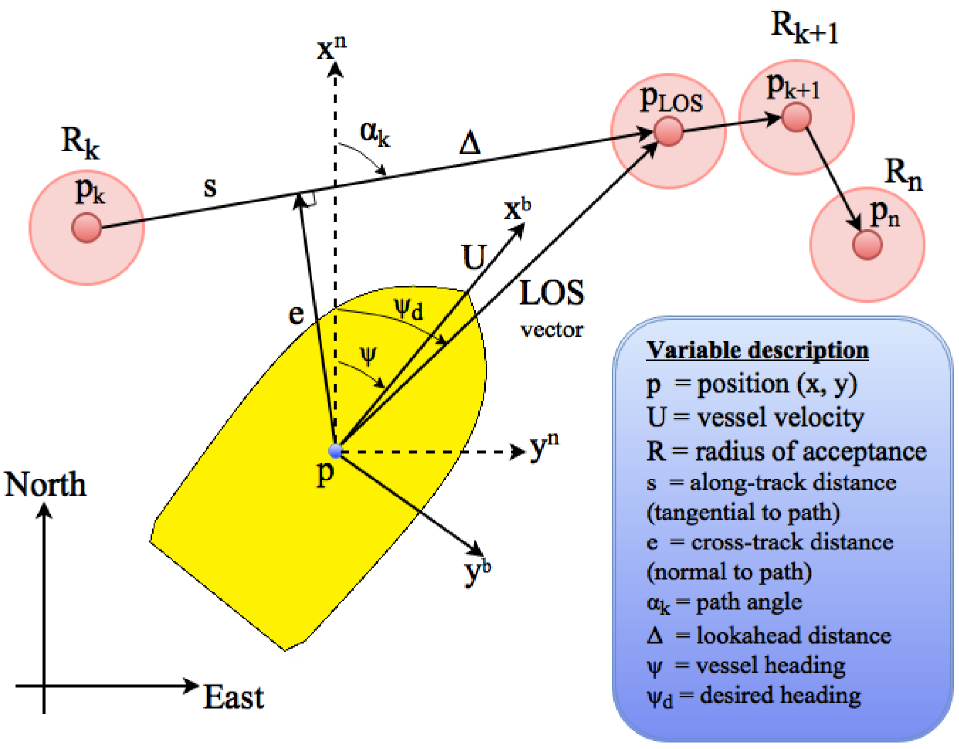

Figure 2.4: Lookahead-based LOS guidance

Finally, the desired heading angle is found as

$$\psi_d = atan2(y_{k+1} - y_k, y_{k+1} - x_k) + arctan(-K_p e) \tag{2.7}$$

It should be noted that for the lookahead distance Δ, a small value increases K_p and gives a more aggressive controller. If the lookahead distance is large, however, the desired heading angle is reduced to only depending on α_k. In addition, the desired state above can be augmented to include integral action. This has proven to be very useful for underactuated crafts that can only steer by attitude information and that have a small lookahead distance (Fossen, 2011). Care must however be taken to avoid overshoots and anti-windup effects.

Fossen (2011) states that LOS-guidance is a frequently used method for path-following, which easily adopts and benefits from the waypoint database. To ensure path-following, guidance laws may compose methods for controlling the vehicle's speed and the LOS steering laws. Speed measurements are not always available, and instead, suggesting that thrusters output is controlled in a different manner. As an example, the thrust can be constant for a given interval, and when the distance to the desired locations starts closing in, thrust may be reduced. The overall result is that the guidance laws ensure that the path is tracked in finite time, which is the main goal.

Several contributions to guidance systems have been made, especially by the missile guidance community, where the mission is to guide a missile to hit and destroy a specific object. In a marine cybernetics context, the analogy is to move or converge to some target asymptotically. In the following, two of the control methodologies will be presented in more detail, i.e., the trajectory tracking and path following control technique. Finally, a method for solving both the geometric and a dynamic assignment in one task is visited, namely the *maneuvering problem*.

2.2.1 Trajectory Tracking

In mathematical terms, the objective in trajectory tracking is to force the system's output $\mathbf{y}(t) \in \mathbb{R}^m$ to track a desired, time-varying output $\mathbf{y}_d(t) \in \mathbb{R}^m$. The trajectory may, for instance, be generated by low-pass filters, optimization techniques or simulating the model of the marine craft, and can be designed for tracking specific targets as well as just paths. If the system achieves this, it is said to solve a trajectory tracking problem (Fossen, 2011). Further on, it is highly convenient to classify the trajectory tracking control laws while regarding the system's configuration space, i.e., according to the available actuators. More generally, with the time-varying reference trajectory for a 3 DOF marine craft defined as

$$\eta_d(t) = [N_d(t), E_d(t), \psi_d(t)] \tag{2.8}$$

tracking of $\eta_d(t)$ is achieved by minimizing the tracking error, defined as

$$e := \eta(t) - \eta_d(t) \tag{2.9}$$

For an ROV that can move in surge and yaw, there are three possible control scenarios. When sway motions cannot be controlled, the ROV is said to be *underactuated*, meaning that it has less control inputs than generalized coordinates (Fossen, 2011). Common for these three scenarios is that the vehicle is attempting to solve an underactuated control problem. The problem can be solved, but may have limited use, and it cannot be solved using linear theory. For a marine vehicle that has two controls for moving in surge and yaw, it is more common to perform setpoint regulation or path-following control. The procedure will be to define a 2D workspace with a cross-track error e (perpendicular to the path) and to minimize e by using an LOS path-following controller. A conceptual description of the LOS guidance law can be seen in Figure 2.4. Δ is the lookahead-distance, a tunable parameter that decides how aggressive the controller should be. For small lookahead distances, augmenting the controller with integral action can be convenient. This is especially true for underactuated crafts that only steer using attitude information and that are affected by current or wind forces (Fossen, 2011). While another method for LOS guidance exist, namely, the enclosure-based steering, only the lookahead-based steering method is considered here, as it is less computationally expensive in addition to being valid for all cross-track errors. Vehicles that can move sideways (perpendicular to the path) by, for instance, a lateral thruster will have less need for a saturated lookahead distance parameter.

2.2.2 Path-Following

According to Dubins (1957), the shortest path between two configurations (x, y, ψ) for a craft moving with a constant speed consists of straight lines and circular arc segments. Therefore, path following is divided into two categories, i.e., path following for straight-line paths and for curved paths. In this way, the paths will be both short, practical and connected.

Just as for trajectory tracking, lookahead-based LOS steering laws have proven to achieve good results when the control objective is path following (Fossen, 2011). While it may be intuitive, LOS guidance also works in 3-D for underwater vehicles. When the vehicle is within a sphere of acceptance defined by some radius, i.e., when the relative error to the waypoint is within some bounded region, a switching mechanism tells the vehicle to move to the next waypoint.

Fossen (2011) states that when attempting to perform path-following for curved paths, the path should be parametrized. This can be done by using interpolation methods such as splines or polynomial interpolation techniques. A solution often used for the path-following involves solving the problem as the geometric task of a maneuvering problem.

2.2.3 The Maneuvering Problem

Skjetne et al. (2004) describes a method on how parametrized paths and a desired dynamic behavior can be used to define a problem statement, called the *Maneuvering Problem*. It focuses on converging to and following a continuously parametrized path, and on how to achieve the desired dynamic behavior along this path. This problem formulation lets us define a general path following problem in terms of waypoints and a dynamic assignment. The latter may involve reaching some desired position, velocity or acceleration condition along and between different waypoints. This implies that the Maneuvering Problem statement is applicable for a great number of applications, and is why the parametrization of paths may be very useful. It does, however, suffer from the fact that the path must be known and parametrized in advance (Fossen, 2011). Skjetne (2005) presents the Maneuvering problem as a technique for formulating the two tasks and solving them both at once:

1. Geometric task: Force the position of the vehicle $\mathbf{p}^n(t)$ to converge to some desired path $\mathbf{p}_d^n(\bar{\omega}(t))$, where $\bar{\omega}$ is a continuous path variable that parametrizes the path:

$$\lim_{t\to\infty} \left[\mathbf{p}^n(t) - \mathbf{p}_d^n(\bar{\omega}(t))\right] = 0 \tag{2.10}$$

2. Dynamic task: Force the speed $\dot{\bar{\omega}}$ to converge to some desired speed U_d:

$$\lim_{t\to\infty} \left[\dot{\bar{\omega}} - \frac{U_d(\bar{\omega})(t)}{\sqrt{(x_d')^2+(y_d')^2}}\right] = 0 \tag{2.11}$$

where the dynamic task follows from

$$U_d(t) = \sqrt{\dot{u}_d^2(t) + \dot{v}_d^2(t)} = \sqrt{(x'_d(\bar{\omega}))^2 + (y'_d(\bar{\omega}))^2} \cdot \dot{\bar{\omega}}(t) \tag{2.12}$$

Worth to mention is that a special case of the maneuvering problem arises when

$$\dot{\bar{\omega}}(t) = 1, \qquad \bar{\omega}(0) = 0 \tag{2.13}$$

which can be recognized as the *tracking problem*. More generally, Skjetne (2005) states that path following is the same as only solving the geometric task, while tracking is a method for both solving the geometric and dynamical objective in the same assignment. This means that tracking is a special case of maneuvering, while maneuvering is a special case of path following.

Some paths are created on beforehand, for instance, for a container ship that is set to track some route in order to deliver goods on the other side of the planet. For real-time operations, e.g., for autonomous operations, path generation and path parametrization need to happen on-the-fly. The paths may be discrete, continuous or a combination of those (hybrid), and path-planning algorithms must be able to generate complete paths that can be followed. In addition, the algorithms need to incorporate techniques for calculating the desired dynamic behavior (expression differentiation, combining path segments, etc.).

2.3 Mathematical Modeling

This section aims to study the mathematical model of a typical remotely operated underwater vehicle, and then to develop the mathematical model for the Videoray Pro 4 ROV. Mathematical models are used to represent a system to study the effects from forces and moments, and to make predictions about the behavior when affected by these forces and moments. It is required to have a model of the system in a model-based control design, when developing observers for state estimation and for simulations. We wish to develop a model that is close to the actual model of the craft that, unfortunately, is restricted by the great laws of the universe. The closer our mathematical model of our craft is to the actual "perfect" ROV, the better we can control and predict the craft's behavior. This model will be of great value when performing simulations with the simulation model, and might be of importance in the further development of functionality for the Videoray Pro 4 ROV. The mathematical model will be derived by studying the dynamics of an underwater vehicle that is moving in four degrees of freedom (DOF).

Fossen (2011) explains that when developing a mathematical model for a robotic system, we may for convenience divide the system into two parts: *Kinematics* and *Kinetics*. Kinematics deals with the geometrical aspects of motion, and does not consider mass and forces. Normally, there is no uncertainty associated with the kinematics, as the knowledge about transformations and reference frames is well established. On the other hand, the kinetics focuses on the forces and moments that act on the marine craft and how they create motions. Mathematical models of these forces and moments have uncertainties connected to them, as the forces are non-linear phenomenons that are difficult to model accurately. In order to properly model the forces and moments, assumptions and simplifications are usually made about the systems to reduce overall complexity. The disadvantage with this is that it may restrict the mathematical model of the system to be valid only when the assumptions are valid.

2.3.1 Kinematics

Notation

The notation that will be used throughout the thesis is the vectorial representation for 6 DOF vessels by SNAME (1950).

Table 2.1: SNAME (1950) notation for marine vessels

DOF		Forces and moments	Linear and angular velocities	Positions and Euler angles
1	motions in x direction (surge)	X	u	x
2	motions in y direction (sway)	Y	v	y
3	motions in z direction (heave)	Z	w	z
4	rotation about the x axis (roll)	K	p	ϕ
5	rotation about the y axis (pitch)	M	q	θ
6	rotation about the z axis (yaw)	N	r	ψ

Reference Frames

A reference system is necessary when we wish to describe and represent the motions of an ROV. Several sets of geographic reference frames exist, but in the cases of local positioning systems, using only two local reference frames suffices to properly describe the geometrical aspect of motions. These reference frames are the known as the *North-East-Down* (NED) or the *North-West-UP* (NWU) and the *body-fixed* (BODY) reference frames, referred to by the {n} and {b} notation, respectively. The geographical reference frames are defined in rotating Earth-centered reference frames. However, the vehicle will be moving at a relative low speed and within a local area where the longitude and latitude values are approximately constant. We may then neglect the Earth's angular rate of rotation, and Newton's laws of motion will be valid (Fossen, 2011).

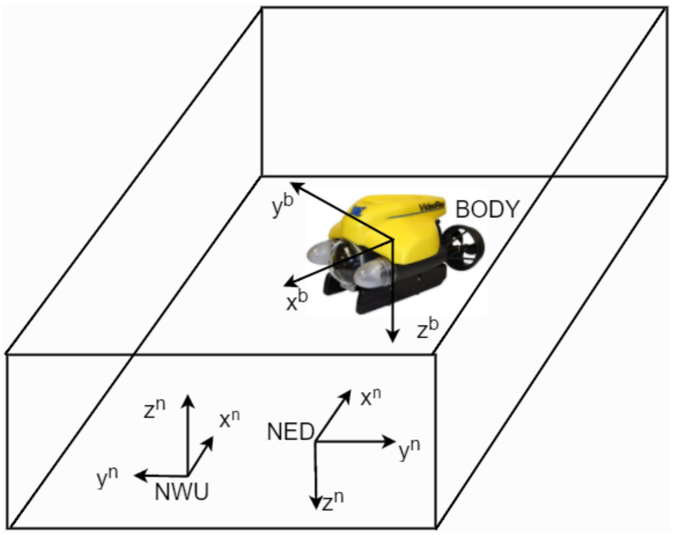

Figure 2.5: BODY frame (x^b, y^b, z^b) inside NED / NWU frame (x^n, y^n, z^n)

For the BODY frame, axes are defined locally, and the design is related to how the vehicle moves and how the actuators generate motion. Both the NED and NWU reference frames define the x-axis as pointing towards true North. The y- and z-axis will point East and downwards for NED, while pointing West and upwards for NWU, respectively. The choice between NED and NWU will depend on the application and practicability (Jekeli, 2001). Both frames are represented here, as the Qualisys Motion Tracking system defines both objects and the local geographical reference frames as NWU frames. All of the above-mentioned reference frames can be seen in conjunction in Figure 2.5.

In the following, both NED and NWU frames will be considered, with focus on NED frames for simplicity. All properties that are dealt with by the NED frame in this thesis frame are inherited by the NWU frame. The main difference is related to how the frames are transformed/rotated and represented.

Transformation

A transformation function is necessary to represent the BODY reference frame in the NED frame. This function is often denoted the *rotation matrix* $\boldsymbol{J}$, and may be decomposed into the rotation matrices $\boldsymbol{R}$ and $\boldsymbol{T}$ in order to transform position and orientation, respectively. Roberts (2006) proposes to write this as

$$\boldsymbol{J}(\boldsymbol{\eta}) = \begin{bmatrix} \boldsymbol{R}(\boldsymbol{\Theta}) & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \boldsymbol{T}(\boldsymbol{\Theta}) \end{bmatrix} \tag{2.14}$$

where $\boldsymbol{\Theta}$ contains the vehicle's orientation. Roberts (2006) continues with explaining that for a 4 DOF system that can move freely in surge, sway, heave and yaw, where roll and pitch are neutrally stabilizing, the rotation matrix can be simplified to

$$\boldsymbol{J}(\boldsymbol{\eta}) = \boldsymbol{J}(\psi) = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 & 0 \\ sin(\psi) & cos(\psi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.15}$$

2.3.2 Kinetics

The mathematical model of a 4 DOF underwater vehicle that moves freely in surge, sway, heave and yaw, can be derived and represented in compact form (Fossen, 2011) as

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{J}(\psi)\boldsymbol{\nu} \\ \mathbf{M}\dot{\boldsymbol{\nu}} &= \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{g}(\boldsymbol{\eta}) \end{aligned} \tag{2.16}$$

where the bold symbols represent that the parameters are of higher order than 1 (not scalars). Here, $\mathbf{M} \in \mathbb{R}^{4\times4}$ is the rigid body and added mass inertial matrix, and $\mathbf{C} \in \mathbb{R}^{4\times4}$ is the rigid body and added mass terms due to Coriolis and centrifugal forces. $\mathbf{D} \in \mathbb{R}^{4\times4}$ contains dissipative (damping) force terms, $\boldsymbol{\tau} \in \mathbb{R}^{4\times1}$ is the propulsion forces and $\boldsymbol{g}(\boldsymbol{\eta})$ comprises the restoring forces. $\boldsymbol{\eta}$ = [x, y, z, ψ]^T and $\boldsymbol{\nu}$ = [u, v, w, r]^T represent the state vector for position and heading, and for velocity and yaw rate, respectively.

The model above can be written in several ways, and external excitation forces are omitted for simplicity. The choice of writing is mainly based on how the different terms will be described and how they are used. In the following, an explanation of the terms in equation (2.16) will be given. As the Videoray Pro 4 ROV can only move in four DOFs, it is only necessary to define and derive the 4 DOF mathematical model.

Rigid Body Kinetics

In order to calculate the forces that act on a system, the most basic form of Newton's second law tells us that the total force equals mass times acceleration. Hence, we start off by finding the rigid body system inertia matrix. For a marine craft with center of mass equal to zero is x- and y-direction, and where the density is evenly distributed, we can simplify the rigid body mass matrix. This is similar to having a starboard–port and fore-aft symmetrical object. The main assumption is that the off-diagonal terms are small compared to the diagonal terms (Fossen, 2011), which simplifies the system to a diagonal matrix. In addition, the roll and pitch states are neutrally stabilizing towards the system's equilibrium for these states, which are zero in both cases. This results in a reduced mathematical model of the rigid body mass matrix to a 4 DOF system taking the following form:

$$\mathbf{M}_{RB} = \mathbf{M}_{RB}^T = \begin{bmatrix} m & 0 & 0 & 0 \\ 0 & m & 0 & 0 \\ 0 & 0 & m & 0 \\ 0 & 0 & 0 & I_z \end{bmatrix} \tag{2.17}$$

Other inertial forces that act on the system are the Coriolis and centrepetal forces. These forces are put on a system or an object that is moving relative to a rotating reference frame. In our case, the object is an ROV and the moving reference frame is the Earth. The intertial force contribution due to the Coriolis and centrepetal effect is often referred to as the Coriolis matrix. Section 3.3 in Fossen (2011) and mass distribution for the ROV allows for simplification of the Coriolis matrix

$$\mathbf{C}_{RB}(\boldsymbol{\nu}) = -\mathbf{C}_{RB}^T(\boldsymbol{\nu}) = \begin{bmatrix} 0 & 0 & 0 & -mv \\ 0 & 0 & 0 & mu \\ 0 & 0 & 0 & 0 \\ mv & -mu & 0 & 0 \end{bmatrix} \tag{2.18}$$

Radiation Induced Forces and Moments

Fossen (2011) defines added mass as a virtual mass that is added to a system due to acceleration or deceleration of a body when the body displaces a fluid. The fluid is forced to move around the object to take the object's former place, as the object and the fluid cannot occupy the same physical space at the same time. In short, the added mass is the extra inertia mass that is added to a system when the body is accelerated. Worth to mention is that some of these hydrodynamical effects are *not yet fully understood* (Skjetne et al., 2004). Further on, because the off-diagonal terms are small and may be neglected when velocities are small, Fossen (2011) proposes that the 4 DOF added mass matrix can be written as

$$\mathbf{M}_{A} = \mathbf{M}_{A}^T = \begin{bmatrix} X_{\dot{u}} & 0 & 0 & 0 \\ 0 & Y_{\dot{v}} & 0 & 0 \\ 0 & 0 & Z_{\dot{w}} & 0 \\ 0 & 0 & 0 & N_{\dot{r}} \end{bmatrix} \tag{2.19}$$

and that the Coriolis matrix due to the added mass effect can be represented as

$$\mathbf{C}_{A}(\nu) = -\mathbf{C}_{A}^T(\nu) = \begin{bmatrix} 0 & 0 & 0 & Y_{\dot{v}}v \\ 0 & 0 & 0 & -X_{\dot{u}}u \\ 0 & 0 & 0 & 0 \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 & 0 \end{bmatrix} \tag{2.20}$$

Hydrodynamic Damping

The dissipative forces that act on the ROV are related to the ROV's velocity. Total damping force on the ROV can be decomposed into two main categories, a linear term and a set of higher order non-linear terms. The dissipative forces acting on the craft can be written as

$$\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_L + \boldsymbol{D}_{NL}(\boldsymbol{\nu}) \tag{2.21}$$

The damping effect is a highly non-linear phenomenon, and it is difficult to estimate the total damping force. The linear part of the damping is dominated by linear skin friction when water is in contact with the ROV. Higher order damping forces are due to turbulent skin friction, wave drift damping, lift forces and drag created by vortex shedding. For a more complete list of potential damping effects, please see Section 6.4 in Fossen (2011). By restricting the speed that the ROV will reach (less than $0.5\frac{m}{s}$), it follows that non-linear damping of higher order than quadratic damping will have a negligible impact. Hence, we obtain

$$\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_{lin} + \boldsymbol{D}_{quad}(\boldsymbol{\nu}) \tag{2.22}$$

Off-diagonal terms may be small (Fossen, 2011), meaning that the two damping matrices can be presented as

$$\mathbf{D}_{lin}(\nu) = \begin{bmatrix} X_u & 0 & 0 & 0 \\ 0 & Y_v & 0 & 0 \\ 0 & 0 & Z_w & 0 \\ 0 & 0 & 0 & N_r \end{bmatrix} \tag{2.23}$$

and

$$\mathbf{D}_{quad}(\nu) = \begin{bmatrix} X_{|u|u}|u| & 0 & 0 & 0 \\ 0 & Y_{|v|v}|v| & 0 & 0 \\ 0 & 0 & Z_{|w|w}|w| & 0 \\ 0 & 0 & 0 & N_{|r|r}|r| \end{bmatrix} \tag{2.24}$$

Restoration Forces

The restoration forces that act on an underwater vehicle are the forces that attempt to drive the system to its natural equilibrium. Fossen (2011) shows that the restoration forces that act on an underwater vehicle where the roll and pitch states are neutrally stabilizing can be written as

$$\boldsymbol{g}(\boldsymbol{\eta}) = \begin{bmatrix} 0 \\ 0 \\ -(W-B) \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.25}$$

where W is the force due to gravitational pull on the system and B is the buoyancy force.

CHAPTER 3

THE VIDEORAY PRO 4 MARINE CONTROL SYSTEM

This chapter is dedicated to the description of the marine control system of the Videoray Pro 4 ROV, based on presented information about the ROV in Chapter 1. Emphasis has been put on the systems dealing with guidance, navigation and control.

The Videoray Pro 4 ROV's guidance system will adopt some of the methodologies elaborated in Chapter 2, while the navigation system shows how to acquire position and orientation data in the MC-lab at NTNU. Finally, the control system presents methods implemented for controlling the ROV, so that the desired control objectives can be satisfied. A thrust allocation module to be developed will be studied briefly, to shed light upon how heading can be controlled and how a proper perfect thrust allocation may be designed.

The low level control is implemented in C/C++. Higher level control layers that directly impact the way of generating paths, calculating control forces and steering the ROV are written directly in Matlab or as mex-functions. This allows for quicker implementation and verification of code. The code commits to a modular approach for the low level control layers, while being slightly more coupled in the higher layers. This ensures that the code is easy to read and understand, while focusing on system utility and making room for further development.

It should be mentioned that the lateral thruster was made to work in the later stages of the thesis work. Therefore, this chapter will deal with the marine control system as if the lateral thruster is non-operable.

3.1 Guidance System

The Videoray Pro 4 guidance system consists of a path-following and a path planning module. While the latter may be regarded as a separate system, it is considered as part of the guidance system due to its simple form. The ROV's guidance system is classified as a closed-loop guidance system, using position and heading measurements as feedback in cascade with a switching controller for deciding the next waypoint. In order to reach the waypoints, the ROV performs path-following built on lookahead-based LOS guidance, as described in Section 2.2. A heading controller ensures that the vehicle is on the path, while a surge thrust controller provides convergence towards the desired waypoint. The Videoray Pro 4 ROV's guidance system block is depicted in Figure 3.1.

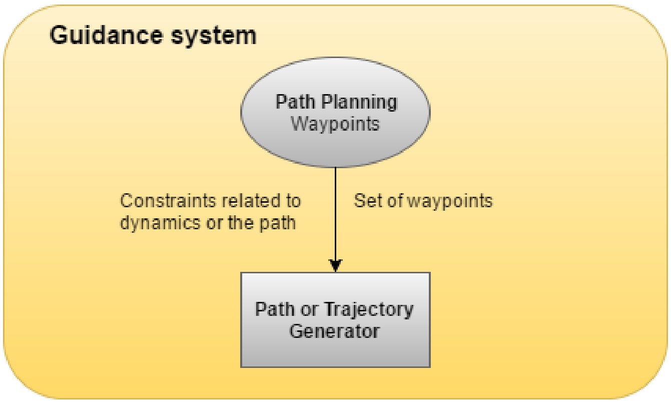
Guidance system
Path Planning Waypoints
Constraints related to dynamics or the path
Set of waypoints
Path or Trajectory Generator

Figure 3.1: Videoray Pro 4 ROV Guidance block

3.1.1 Path Planning

In the path planning module, a predefined set of waypoints is sent to the ROV upon program initialization. In the first phase of developing a waypoint guidance system, focus is put on showing that a set of waypoints can be generated and given as input to the guidance system. Path-evaluation is not necessary for the path-planning being performed here, but may become very useful for further development of path planning systems. Also, the ROV will move within a bounded area, defined by where position data can be retrieved and, hence, making a restriction on reachable waypoints.

Waypoints are defined in 3-D space by a matrix. Radius of acceptance and desired velocities between waypoints can be defined, including other user-defined specifications, but the matrix will need to be augmented for this kind of set-up. The simplest way of doing this is by creating an object, sometimes referred to as a *struct*. It should be noted that a struct is equivalent to the *type* object, and may contain a large set of properties. In many cases, the struct is called WP, and it is given properties such as $WP.Position$, $WP.R_{acc}$ and $WP.Speed$, or similar. The system is already designed for this approach, but has not been extended to fully develop these additional requirements.

3.1.2 Path-Following: LOS Guidance

The waypoint guidance system mimics the methods described in Section 2.2. No velocity measurements exist, but it is possible to use a Kalman filter to estimate velocity based on position data, in order to control thrust with a speed controller. The Kalman filter can be developed by letting the system be defined by the first part of equation (2.16). However, the thrust-to-force relationship is unknown, and with dead-band in the thrusters, this approach was not taken.

The system aims to follow the path generated by the LOS guidance system with a heading controller, and by producing constant thrust while on the path. This ensures that the craft converges to the desired location. If the vehicle reaches the waypoint within a radius of acceptance, the next desired waypoint will be chosen. By exploiting that a speed different from 0 can be kept by the craft when closing in on the desired waypoint, the radius of acceptance can be chosen so that the ROV's trajectory is smoother.

References generated by the guidance system consist of desired values for both depth and heading, before being sent to the control system. In order to determine the reference for the current state of the ROV, receiving data from the navigation system will be required. Two constraints are put on the path-following system. Firstly, the heading angle must be reached with an error less than 5 degrees before moving along the path. This criterion should ensure that the path is followed successfully without deviating too much. Next, the desired depth should not have an error larger than 5 cm. When these control objectives are satisfied, the guidance system will give signal for the control system to start following the path by producing thrust in surge. As mentioned previously, the guidance system implemented is based on LOS guidance, but only in the plane (2-D) and in cascade with a depth controller. That is because a 3-D LOS guidance system would require that the vehicle is within a radius of acceptance of the waypoint defined as a sphere. If the waypoint is reached in the plane before the desired depth, the ROV will attempt performing DP by creating forces and moments in surge and yaw. As the craft has no proper method for performing DP without a fully operable lateral thruster, the ROV will begin spinning and get entangled in its own tether. Including lateral thruster control will allow for proper 3-D LOS guidance, as described in Chapter 10 by Fossen (2011).

In order to move towards the waypoint, the desired heading must be calculated. Both the heading and the course angle were previously defined in equations (2.2) - (2.7). By neglecting the lookahead distance Δ, i.e., assuming an infinite lookahead distance, the expression for the desired course angle can be reduced to

$$\chi_d = atan2(y_{k+1} - y_k, x_{k+1} - x_k) \tag{3.1}$$

The environment is current-free, and the desired heading being fed to the control system is given as

$$\psi_d = atan2(y_{k+1} - y_k, x_{k+1} - x_k) \tag{3.2}$$

3.2 Navigation System

The navigation system, depicted in Figure 3.2, uses position data from the Qualisys motion tracking (QMT) system, and incorporates systems responsible for communicating with and receiving measurements from the QMT system. For both textual and figurative description of the communication tree, the reader is advised to visit Appendix B.

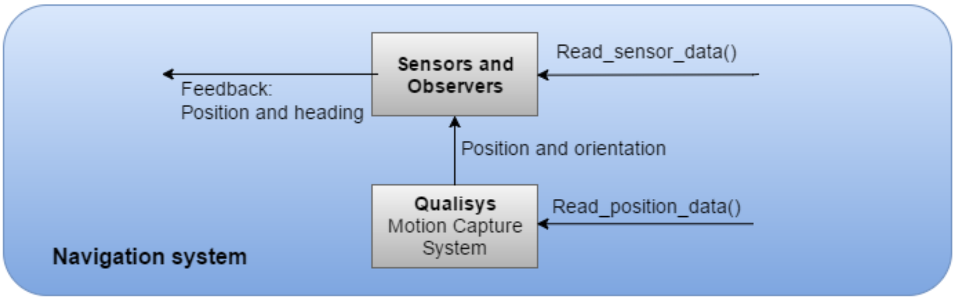

Figure 3.2: Videoray Pro 4 ROV Navigation block

3.2.1 Qualisys Motion Tracking System

Qualisys, also referred to as the Qualisys Motion Tracking (QMT) or Qualisys Motion Capture (QMC) system, is responsible for measuring both the position and orientation of the ROV. In general, this data is estimated using cameras incorporating an optical motion capture technology, estimating the position of an object with extremely high accuracy. The cameras detect a set of four markers or more as a complete object, uniquely defined by the marker positioning. From this, the cameras estimate position and orientation of some weighted average of the designated marker configuration. In the MC-lab, the system is defined as a local North-West-Up (NWU) frame, with the positive z-axis pointing upwards and being 0 on the pool floor. By default, all identified objects are defined as NWU frames.

Capture rate is updated at a very high rate (100-150Hz), and can be read by Matlab in approximately 0.02-3 seconds. The ROV's movement should be confined to the area where Qualisys cameras are effective. This effective area mainly depends on the camera configurations, water quality, lighting effects and marker placement. Connecting to the underwater QMT system and receiving this information in Matlab has been further developed to work in Matlab, and can be seen in Appendix B. This is a general method for communicating with the QMT system in Matlab, but requires Linux OS. For Windows and Mac, this technology already exists as mex-functions.

For the Videoray Pro 4, a custom made rig for the ROV is mounted on the ROV's bottom side. Markers are placed on the arms extending out from the rig, with focus on having a good marker spread. Rig and marker set-up can be seen in Figure 3.3. The camera system can more easily distinguish the markers and identify this configuration as an object, even if some of the markers are not successfully captured. Due to the ROV's characteristic shape, especially in terms of shadowing the markers, difficulties may be encountered when attempting to achieve position measurements using the Qualisys underwater positioning system. Not only is the marker configuration important for finding the object, but the loss of markers may affect how Qualisys estimates the object's orientation.

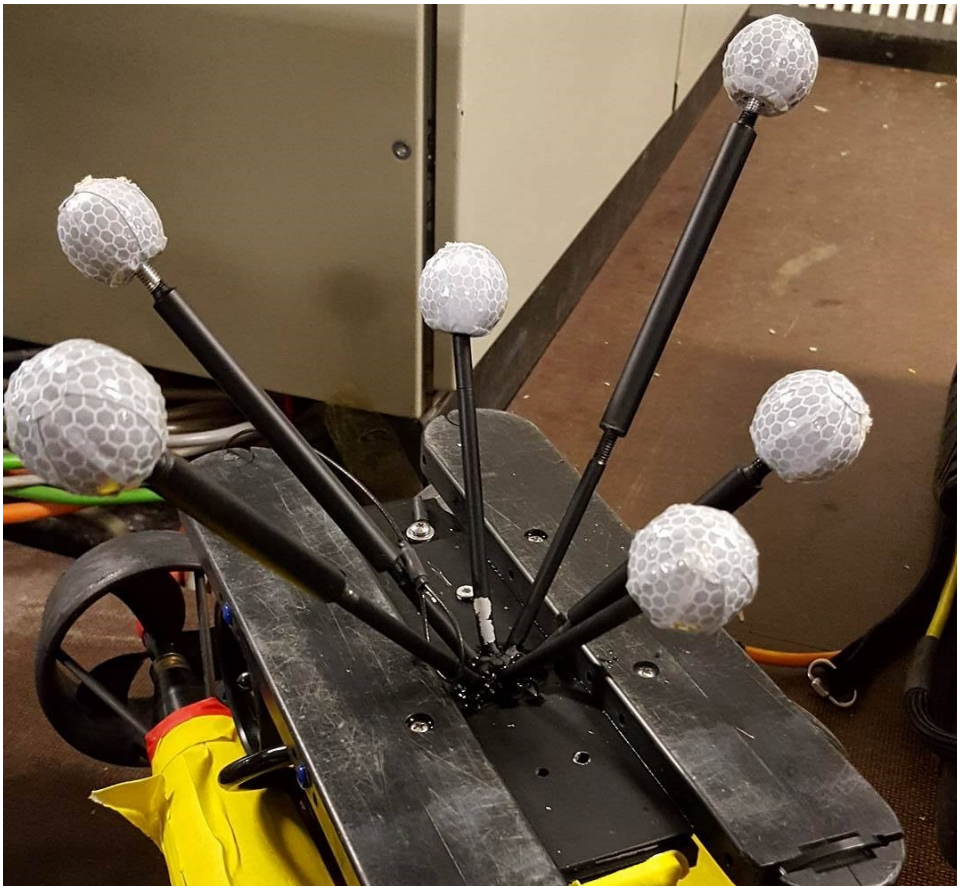

Figure 3.3: Videoray Pro 4 ROV with rig and markers

To handle occasional data drop-outs, one simple technique involves using the previously measured position and orientation if it is a short duration since they were updated. Supposedly, this time frame may be a second or less, but can be adjusted manually. It should be noted that system is already designed for adopting observers for position and orientation estimation.

There are some challenges to overcome when attempting to receive position data. Light is reflected and absorbed differently underwater, and the water may be dirty and full of small particles, hence reducing marker visibility. Properly calibration of the cameras, configuring the underwater camera settings and turning off lights in the lab are the main

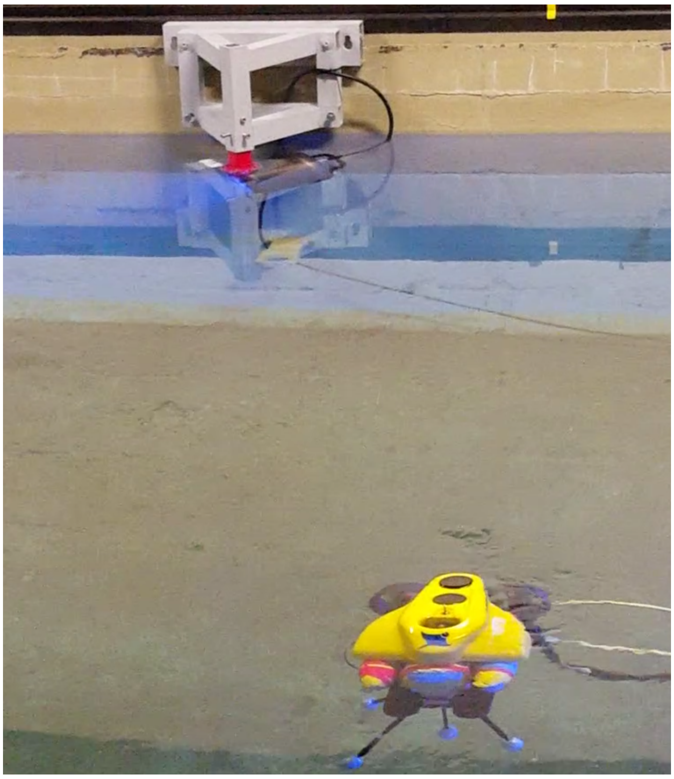

Figure 3.4: Videoray Pro 4 ROV in the MC-lab

suggestions for having an improved experience with receiving position measurement. For the Videoray Pro 4 ROV, it is suggested that the camera settings are adjusted somewhat similar to those presented in table 3.1 below. Capture rate and exposure time should be high, and the marker threshold low. These values are good for 1-2 cm diameter-sized markers, and filters out a lot of the unwanted noise. Focus and aperture will depend a lot on the area of operation, but should probably remain between 2-5 and 2-4 for use in the MC-lab, respectively.

Table 3.1: Suggested underwater Qualisys camera settings

Marker	**Value**	**Unit**
Capture rate	100	Hz
Exposure and flash time	1000-1500	μs
Marker threshold	40	%
Focus	3	m
Aperture	2.8	f

3.2.2 Sensors

As the vehicle is positively buoyant and Qualisys data drop-outs occur, using the pressure sensor for depth estimation addresses several advantages. Having measurements is an absolute necessity for a positively buoyant underwater vehicle performing depth control. A proper way to ensure this is by using the built-in pressure sensors and perform a pressure-to-depth conversion. When Qualisys measurements exist, however, the Qualisys data may be used in feedback. This may be beneficial, as Qualisys measurement noise is in the order of millimeters, while the pressure sensor commits to noise of centimeter order.

Pressure measurements are already filtered, but measurement noise on the sensor encourages for applying further filtering to the signals. Additional filtering has not been done to the full extent, but some testing with varying degrees of discrete lowpass filters resulted in unsatisfactory and slow measurement update rates. To make comments about the other sensors, the in-built compasses and IMUs that measure the heading are useless, as the drift is highly inconsistent. The reason may be due to all the metals and the magnetic fields they create in the MC-lab, disturbing the actual reference to the true North that the sensors try to measure.

The pressure sensor is used to estimate the ROV's depth and to account for varying atmospheric pressures. Depending on whether we are in the ocean or in a test facility above (or below) sea level, the atmospheric pressure is calculated when the ROV software is initialized (when it is on the surface and only the atmospheric pressure is read). This allows for using the barometer for estimating the pressure with great certainty that the measurements don't deviate much from the actual pressure. Of course, other factors will limit the correctness of using the pressure sensor for depth estimation, e.g., varying water column density and gravitational pull. According to Pettersen (2007), the connection between heave and pressure can be expressed as

$$P_m + b = P_{atm} - \rho g z(t) \implies z = -\frac{\Delta P_m - P_{atm}}{\rho g} \tag{3.3}$$

where P_m is the measured pressure, b is the sensor bias, ρ is the density of the water, g is the gravitational constant and where z is defined positive upwards. Any change in the pressure will come from a proportional transition in depth. The linear connection implies that the ROV can operate solely on pressure as control input.

3.3 Control System

The control system calculates the forces to be produced by the ROV so that it can satisfy some control objective. A modular approach as depicted in Figure 3.5 below has been taken, which only depends on feedback from the position and heading measurements.

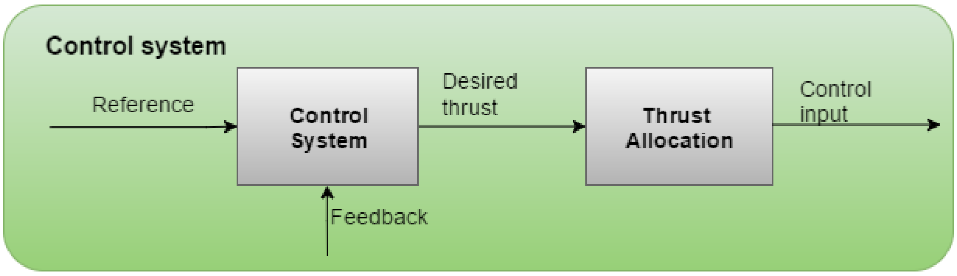

Figure 3.5: Videoray Pro 4 ROV Control block with position and heading feedback

3.3.1 Controllers

All the controllers that have been developed and implemented are proportional-integral-derivative (PID) controllers. The conceptual description of the implemented PID controllers can be seen in Figure 3.6.

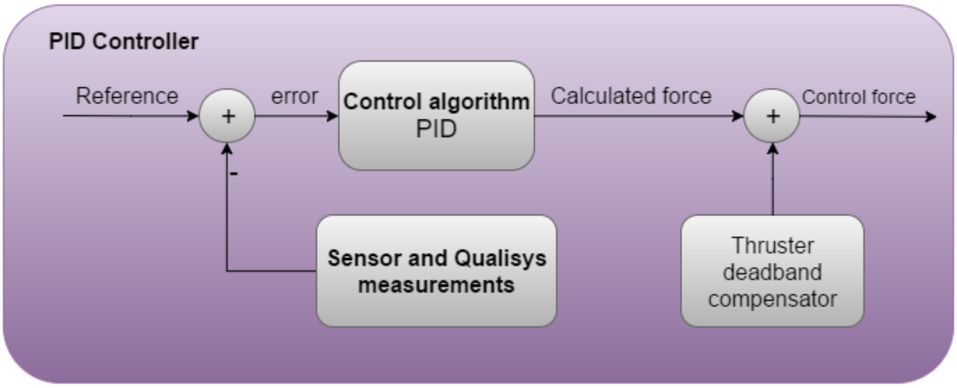

Figure 3.6: PID control law

The PID control law minimizes the error e between desired and measured state in order to satisfy the control objective. A thruster dead-band compensator block is added in Figure 3.6, but is only present in the PID controller that manages the depth. This term helps overcoming the slow integral build up, minimizes overshoot towards the reference and, more importantly, helps overcoming the dead-band when attempting to keep the reference due to the craft's positive buoyancy. Mathematically, the PID control law can be represented as (Fossen, 2011)

$$\tau = K_p \cdot e + K_i \int_0^t e \,\mathrm{d}\tau + K_d \cdot \dot{e}. \tag{3.4}$$

where τ is the control force, $\dot{e}$ is the derivative of the error w.r.t. to time and K_p, K_i and K_d are the proportional-, integral- and derivative gains, respectively. The system is a *discrete-time* system, and the integral is calculated as the sum of errors over time. A number of discrete integration techniques exist, each with pros and cons in terms of computational time, complexity and accuracy. For most practical applications with the Videoray ROV, it is argued that the forward-Euler integration technique is sufficient.

Remarks on the Depth Controller

The depth controller relies on feedback from the pressure sensor measurements and a simple conversion from pressure to depth, which yields a good depth estimation. Qualisys depth data is not used, mainly due to the importance of having depth measurements when submerged. A method for alternating between Qualisys and sensor data, that prioritizes Qualisys data when available, can be developed. This is not seen as a necessity, and is therefore not implemented. Additionally, the depth controller has been augmented with a constant thrust force to overcome a problem related to a small heave thruster dead-band and a slightly positive buoyancy force.

Remarks on the Heading Controller

For controlling heading, the PID controller is used with Qualisys measurements as feedback. If Qualisys data is unavailable, either due to the ROV moving outside the area where cameras can see the markers or if communication is lost for other reasons, satisfying the control objective will be difficult. However, if measurements are lost for a short duration of time (this duration can be specified manually), the system will use the previously estimated position and orientation. When data are absent for longer time intervals, the system returns to manual control if only the heading controller is active, but will continue with performing LOS-based path-following and heading control if the path-following objective has been activated.

Saturating elements on the output force and integrator anti-windup effect has been included. This prevents the controller from producing thrust above or below physical limits, and restricts the integrator from building up and growing out of proportions. It should be noted that since the force-to-thrust is unknown, and no estimated relationship between these was made, controllers calculate the system input directly (thrust in percent). Comparing controller gains for simulation model and the actual system is, therefore, non-trivial until the thrust-force relation has been found. The final remark on this controller is concluded by how the heading is calculated. A *rad2pipi* function (MSS, 2010) transforms an angle to be within $(-\pi,\ \pi]$ in radians. This feature lets the system understand how to make the smallest turns, increasing overall system performance.

3.4 Thrust Allocation

The system responsible for allocating thrust is based on how the thrusters are placed. Figure 3.7 below shows how the propellers are aligned, and how this configuration is able to produce thrust in surge, sway, heave and yaw.

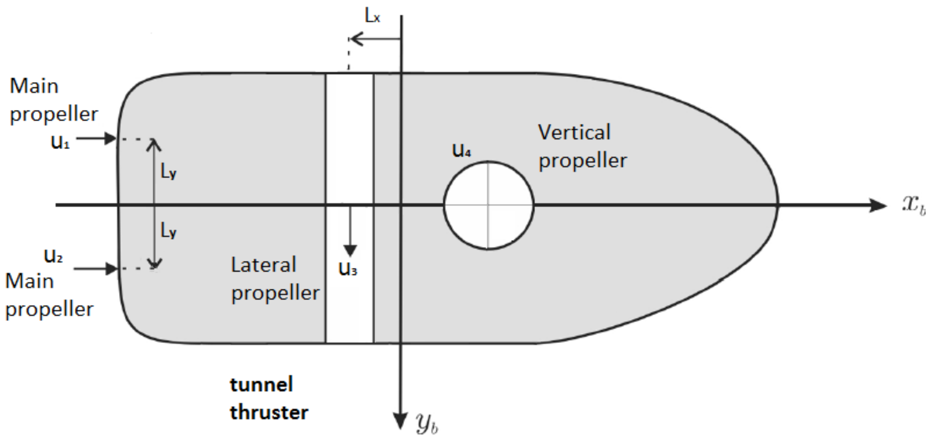

Figure 3.7: Videoray Pro 4 ROV thruster configuration

By looking at Figure 3.7, the thrust allocation algorithm presented in Chapter 12 by Fossen (2011) for the Videoray ROV can be written as

$$\tau = F\,K\,u \tag{3.5}$$

$$\tau = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ L_y & L_y & 0 & 0 \end{bmatrix} \begin{bmatrix} K_1 & 0 & 0 & 0 \\ 0 & K_2 & 0 & 0 \\ 0 & 0 & K_3 & 0 \\ 0 & 0 & 0 & K_4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \tag{3.6}$$

where K is a function to be determined by thruster force tests, and where $K_1 = K_2$. While no proper values for the K matrix exist, presenting equations (3.5)-(3.6) gives insight in how to allocate the forces once the force[N]-to-thrust[%] relationship is found. In order to find these values, one method is to fit a rig to fasten the ROV to a force measurement device, command thrust and measure the force. For different input commands, a force value will be produced. This gives a set of points, where second or third order curve fits can be applied to find the value for K as a function of control input u. The only challenge with this is to make a rig that fits the Videoray ROV, as its geometrical properties are a bit clunky for placing or connecting equipment.

Thrust allocation for the controller used when manually controlling the ROV is given in Section 1.2.

CHAPTER 4 SIMULATION RESULTS

In this chapter, we will investigate a simulation model of the Videoray Pro 4 ROV. Testing is necessary in order to verify that the model is able to satisfy a set of control objectives. The goal is to show that automatic controllers for depth and heading, in addition to waypoint guidance in 3-D space, can be performed. Moreover, the simulation model has been developed using mathematical modeling from Section 2.3. How the parameter values were chosen is the first topic to be addressed in this chapter.

Results from testing the controllers and lookahead-based LOS waypoint guidance system are presented next. For controlling depth and heading, PID controllers have been developed, including a linear quadratic regulator (LQR) for depth, while a PID speed controller has been created for thrust generation in surge. The simulation model aims to verify that desired control of our system can be achieved, and represents how the experimental results should look like if the path is followed successfully.

Finally, a small conclusion to the simulation results will be drawn. Arguments can be made about the model's current state having limited use due to errors in numerical values for the hydrodynamical parameters. Successful implementation of the previously described control techniques proves that the model is suited for further research, and a change in parameter values should not impact the system's general behavior. Everything is modeled using Matlab and Simulink, where structure and simplicity have been emphasized.

4.1 Mathematical Model and Hydrodynamic Parameters

A mathematical model of the Videoray Pro 4 ROV uses the hydrodynamical parameters found in Eidsvik (2015). There are uncertainties linked to these numerical values, mostly due to bad sensor calibration and fitting of the equipment used for the parameter estimation. Some parameters are not even estimated, and assumptions have been made to the missing or badly estimated model parameters. In addition, the validity of comparison with the real system may be further diminished due to the mounting of a lateral thruster and a rig with markers. The thruster adds about 0.8 kg mass to the system, while the rig adds another 0.3 kg. Geometry of the vehicle is slightly changed compared to its previous geometrical state. This may suggest investigation of the impact caused by increased weight on the system, and more specifically, how damping effects are affected. The parameter values and system properties are listed in Appendix A. In the following, some comments and arguments on the mass and damping properties, including how the parameters were chosen for the simulation model, will be presented.

Rigid body and added mass parameters

The rigid body mass of the system is found by measuring the ROV's weight, while the mass moment of inertia is found through a CAD analysis made by Eidsvik (2015). Further on, added mass was provided by Eidsvik (2015), using the *WADAM* hydrodynamic analysis program. All of these values are assumed to have been measured with minimal error, but the mounted lateral thruster and rig were not included in these estimates. The effect imposed by the rig and lateral thruster may not be too large for added mass, but will add another 1.1 kilos to the total system mass and a slight change in geometry. For a refreshed set of numerical values, the CAD model can be reworked to also include the lateral thruster and rig.

Coriolis matrices are calculated based on the rigid body mass and added mass matrices, and will have the same error magnitude as these matrices.

Hydrodynamic damping parameters

Linear damping

Linear damping coefficients are taken from the empirical results. This is because the CFD analysis most likely underestimated the linear damping by assuming smooth surfaces, and because the experimental testing was prone to unknown errors. Some additional damping in surge has been added to compensate for the mounting of the lateral thruster.

Quadratic damping

Quadratic damping coefficients in surge and heave were taken from the empirical values, or as the mean of the CFD and the experimental results. For sway, the empirical and CFD values are almost identical, and the empirical value was chosen. In roll, pitch and yaw, the experiments gave no realistic quadratic damping coefficients, but as the values for empirical and CFD are close, the mean of these values were chosen.

4.2 Case 1: Depth Controller

The objective is to control the vehicle to a set of different depths and to study the ROV's behavior. Both a PID controller and a linear quadratic regulator (LQR) controller was developed, derived using the work of Chen (1984) with reference feedforward. The LQR controller can be found inside the control system block in Appendix C, which has been depicted in Figure A.2. Results from testing the PID and LQR controllers are almost identical. The plot of the LQR controller performance is shown in Figure 4.1 to demonstrate that simple model-based control can achieve satisfactory behavior. It can be seen that the controller moves the ROV to a depth quickly and without difficulties. When the error is less than 5 cm and this error has stayed within 5 cm accuracy for about 10 seconds, the next desired depth is set. The ROV is assumed to be neutrally buoyant, which will be the case for the actual ROV as well.

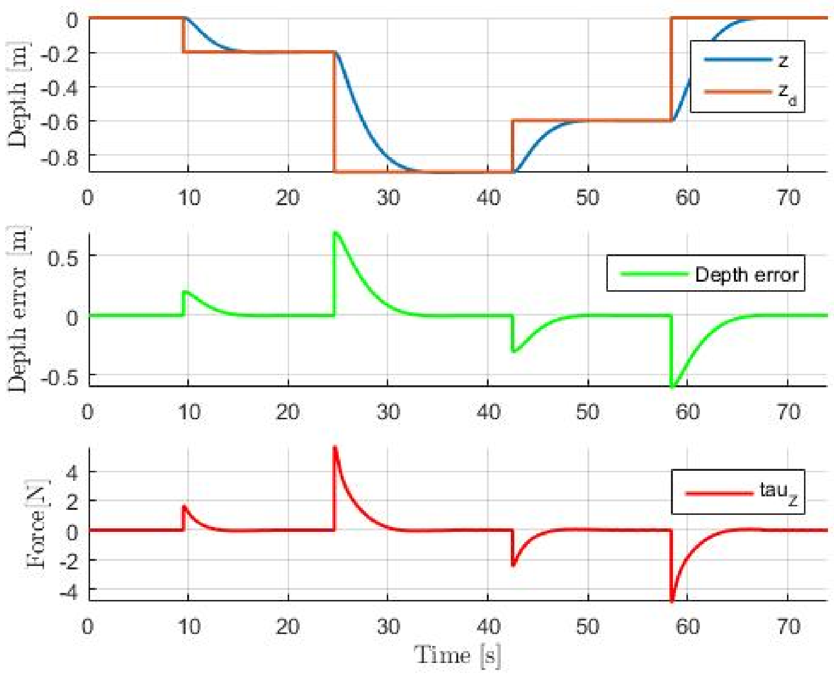

Figure 4.1: LQR depth controller response

4.3 Case 2: Heading Controller

This simulation case shows that the desired heading can easily be reached and maintained, seen in Figure 4.2. It can be observed that the momentum required to change the heading is very small, due to the properties of the ROV, but also that the produced momentum oscillates slightly. This oscillation happens when the derivative-term of the PID controller is increased, ensuring that the ROV's heading better converge to the desired value. Even with these small oscillations, the system nicely tracks the desired heading angles. The system will always choose the optimal turning angle when a there is a change in the desired heading. This can be observed after 60 seconds in Figure 4.2, where the desired heading angles are drawn in orange while the actual heading of the model is described by the blue line. The heading angle is given between -180 and 180 in degrees for all consecutive plots depicting the heading.

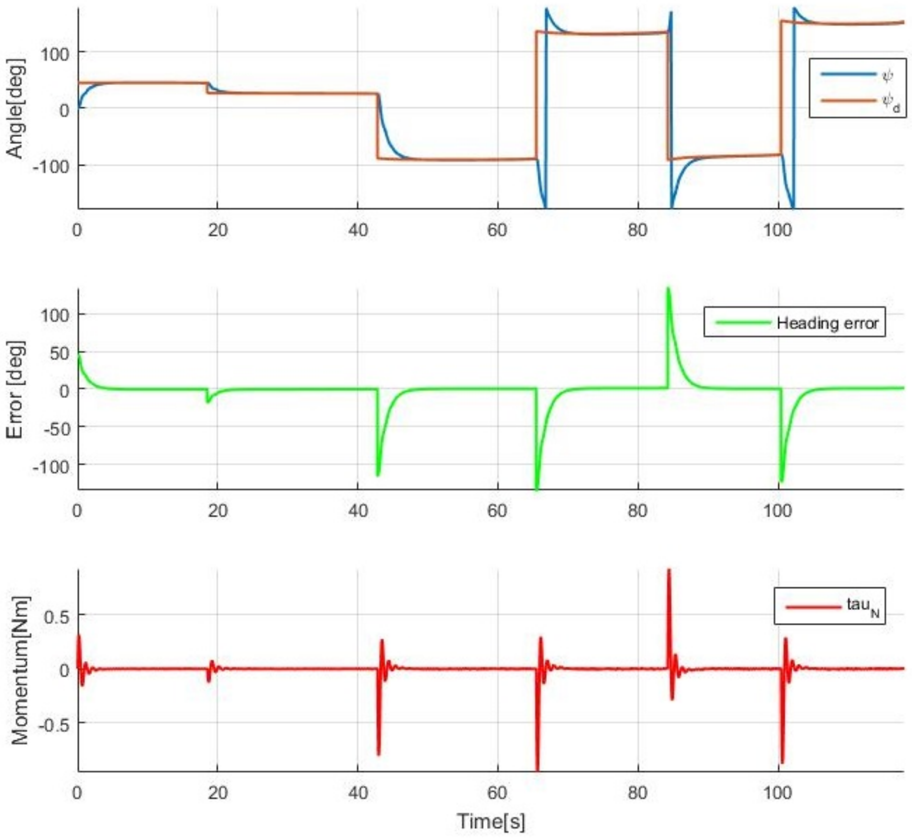

Figure 4.2: PID heading controller response

4.4 Case 3: Underwater Path-Following

When performing path-following, two different simulation cases will be presented, with the intent to carry out similar tests in the experiments with the Videoray ROV. The two cases are path-following for a zig-zag pattern with sharp corners on the surface, and a lawnmower pattern below the surface. As will be further elaborated in Chapter **??**, the XY-plots have been rotated to a North-West-Up frame. This may already be obvious, however, considering that the depth plots present a z-axis positive upwards, and makes comparing with experimental results a great deal easier.

4.4.1 Path-Following: Zig-zag Pattern

In the first path-following simulation, the objective is to follow a path taking a zig-zag shape on the surface. This pattern is recognized as straight lines and sharp edges, i.e., the angles created by two lines intersecting in the connecting waypoint are small. This puts a lot of pressure on the ROV in terms cutting the corner and following the path. The result from the run can be seen in Figure 4.3.

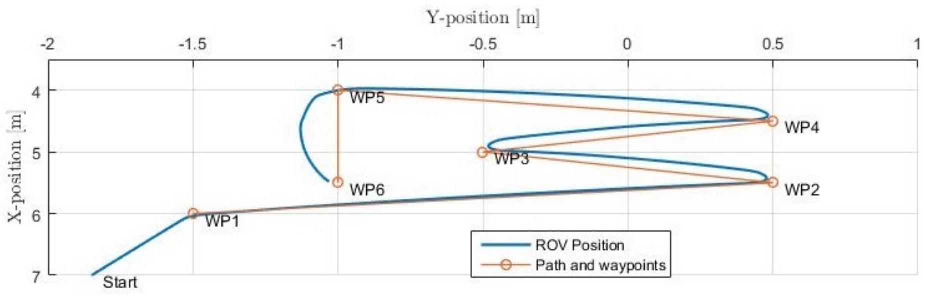

Figure 4.3: Path-following simulation: Zig-zag pattern

The ROV is initialized on the surface at (x, y) = (7, -1.9). The path is difficult to follow nicely, and the craft motions are smooth. The full run takes about 35-40 seconds, and all of the waypoints are reached within an error of a few centimeter. A desired velocity of 0.5 m/s is given to the speed controller, which is produced when error measurements for heading and depth are within 3 degrees and 1 cm, respectively. Radius of acceptance is set to $R_{acc} = 10\ cm$ in order to better cut the corners and follow the next straight-lined path segment. The velocity when reaching waypoint 5 is a little too high, resulting in a small deviation to the path. The path deviations between waypoints 2-5 are below 5 cm.

4.4.2 Path-Following: Underwater Lawnmower Pattern

Figure 4.4 shows the final simulation model test case, where the ROV moves in an underwater lawnmower pattern by adopting the lookahead based LOS guidance law. Similar to the previous path-following case, the desired velocity is chosen as 0.5 m/s. The ROV is commanded to lower the velocity when closing in on the waypoint to reduce deviations from the path. Radius of acceptance is set to $R_{acc} = 10\ cm$ for all waypoints except the first and final waypoint, where $R_{acc} = 5\ cm$ has been chosen. Having a higher radius of acceptance makes the ROV better cut the corners, but may also contribute to deviation from the path. From Figure 4.4, it can be observed that the simulation model follows the path nicely. After desired heading and depth are reached, surge force in produced.

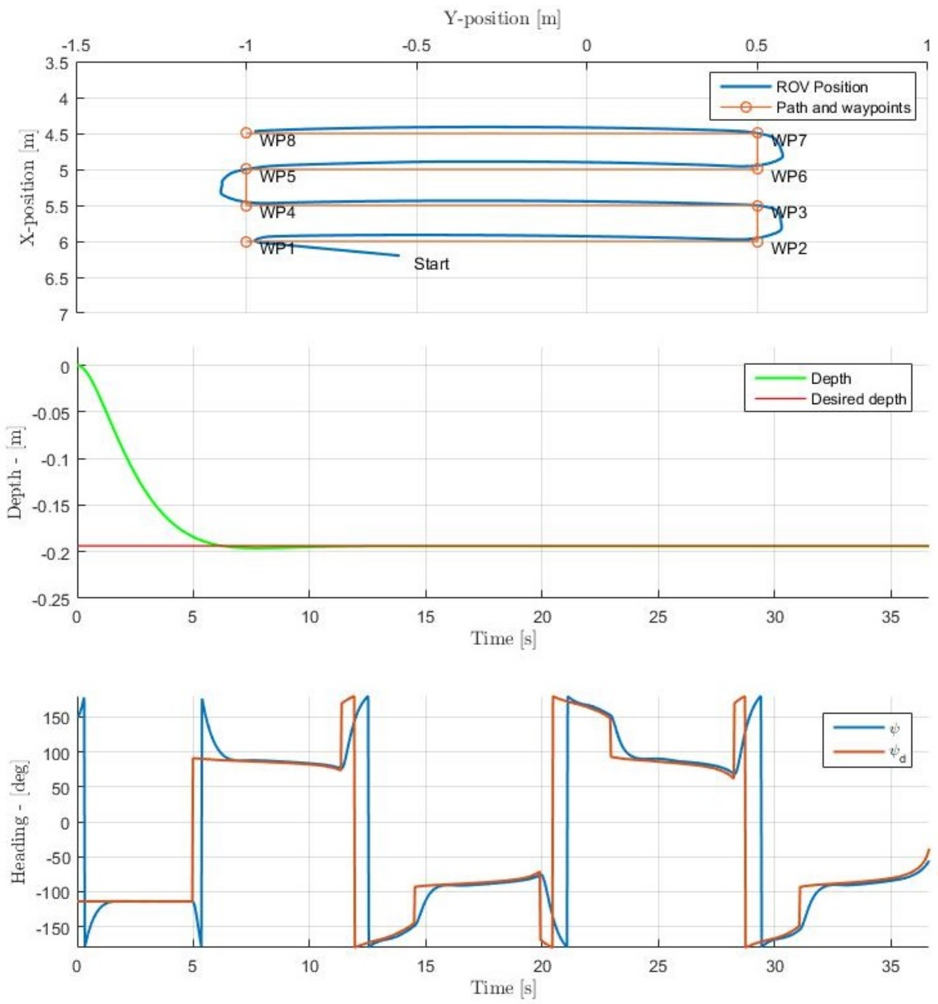

Figure 4.4: Path-following simulation: Underwater lawnmower pattern

4.5 Concluding Remarks

This section concludes the simulation model testing, where a mathematical model of the Videoray Pro 4 ROV has been created and simple control techniques for controlling the depth and heading have been successfully implemented. The system can perform path-following along straight-lined paths, based on a set of waypoints and by using the lookahead-based LOS guidance steering law. As proposed by Figures 4.3 and 4.4, the set of waypoints can be chosen arbitrarily and the path will be followed.

Both the heading and depth controller show quick convergence towards the desired values. A time-varying heading reference would allow the system to perform path-following along any arbitrary, smooth path, as the turning radius is 0. How the system so nicely satisfies both reaching and maintaining desired depth and heading using PID controllers motivates for using the same control laws for the actual system. While the gains used are not commented thoroughly, they are found by trial and error and can be seen in the Simulink model in Appendix C. Different types of noise could be added to the system for more realistic measurement data, which will be the case in real-life applications. This has been omitted for simplicity, and because the emphasis is on verifying implementation of methodologies.

The LOS steering guidance law with the PID speed controller ensures that the model stays on the path and follows the path towards the waypoints. No obstacles are present, information about the system is known in advance and all distances between waypoints have been kept short for simplicity. In addition, by keeping the velocity small when closing in on the waypoint the path can be followed nicely even without a saturated lookahead distance. If the ROV started deviating a lot from the path, using a saturated lookahead distance Δ would have been attempted. Also, because of the small region where path-following will be performed in the MC-lab, introducing a lookahead distance may simply make the system unnecessarily complicated. As Fossen (2011) explains in Chapter 10, the steering law incorporating the lookahead distance is equivalent to a saturating control law. It is useful for underactuated crafts that only steer by attitude, which will be excessive when the Videoray makes use of its lateral thruster. Therefore, the saturation of Δ was not given much attention, and while it has its advantages in terms of, e.g., current compensation, no currents are present. Therefore, the idea is to show that reaching desired waypoints are in fact a feasible task using LOS guidance in a small area. Simulations showed that a nice response was achieved without saturating Δ, and it was concluded that, initially, it will not be implemented in the actual ROV. LOS steering has therefore successfully been implemented to ensure waypoint guidance for a vehicle similar to the Videoray ROV, by only producing forces and moments in surge, heave and yaw.

CHAPTER 5 EXPERIMENTAL RESULTS

Chapter 5 covers experimental testing with the Videoray Pro 4 ROV, based on Chapters 1-3. Conventionally, the testing procedure is divided into different test cases similar to those in Chapter 4. As previously stated, the idea is to keep the test cases simple. Achievements, difficulties, limitations and results are presented in a clear manner, proving that the basic control strategies work as intended. It should be noted that the plots displaying positioning in the XY-plane are drawn in a North-West-Up (NWU) reference frame, as this is the way the position is represented in the QMT software. This means that the heading is calculated as positive when moving counter-clockwise from the North-axis, and the z-axis (depth) is negative downwards. Most of the experimental testing was performed from the bridge in the MC-lab, and the plots are turned 180 degrees for a, presumed, better visual experience.

Furthermore, all the experiments were conducted in the MC-lab at NTNU with the aid of the QMT system for position and orientation measurements. To begin with, emphasis will be put on how the positioning system is able to capture the markers. Then, test case results will be studied, while the next chapter will discuss these findings. Overall, testing shows very satisfying results. The ROV nicely adopts PID controllers for depth and heading, and the LOS guidance methodology to successfully perform automatic waypoint guidance and path-following.

5.1 The Qualisys Underwater Positioning System

The Videoray object is defined in the QMT software by some "optimal structure" of the marker configuration. As a consequence of this, Qualisys defined the heading of the ROV different from the actual heading, and an offset ψ_0 was added to compensate for this error. For the rig and marker configuration used, $\psi_0 = 130$ [deg] has been applied. This value can either be changed manually in the Qualisys software, or in the Matlab code file that receives estimated heading angle. The offset angle may not be 100% accurate, but has shown to be close to the actual offset.

Once in a while, one or more of the markers are not captured by the cameras, and the Qualisys algorithms will attempt to use the previously stored data to estimate the location of the object. Qualisys may lose the exact position and orientation if a number of markers are not found for a set period of time. These drop-outs seem to be happening more frequently in some areas and for heading angles close to 0 and 180, but more testing will be required in order to determine whether this is actually the case.

Most of the time, however, Qualisys measurements are available. Figures 5.1 and 5.2 depict the position of the ROV from typical runs in the MC-lab. The ROV is manually controlled by the Sixaxis controller, where the position of the ROV is represented by the blue line, and it is attempted to stay close to a path, drawn in orange. This aims to show what to expect from the Qualisys system when performing path-following for similar patterns. Position and orientation data are occasionally lost, sometimes only for fractions of a second, but may also be absent for longer periods of time (rarely ever more than a second). Initial positions are approximately (x, y) = (3.3, -1) for the first run in Figure 5.1 and (x, y) = (6.8, -2) for the second run in Figure 5.2.

Not all possible positions and orientations have been attempted to be reached. The area covered here represents where position data is most likely to be received for the Videoray ROV, with the attached rig and marker set-up in Figure 3.3.

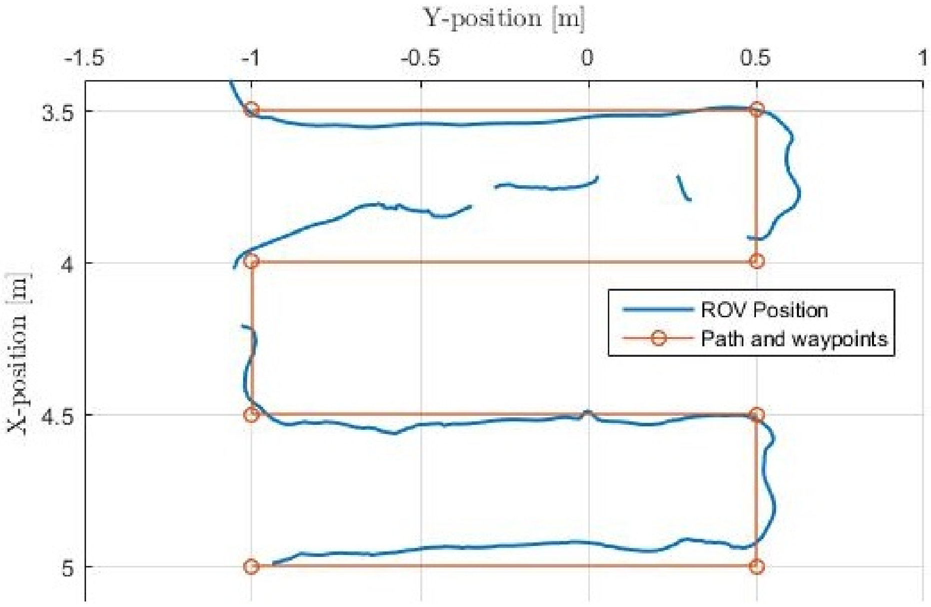

Figure 5.1: Position capture using Qualisys. Area: Near the control room

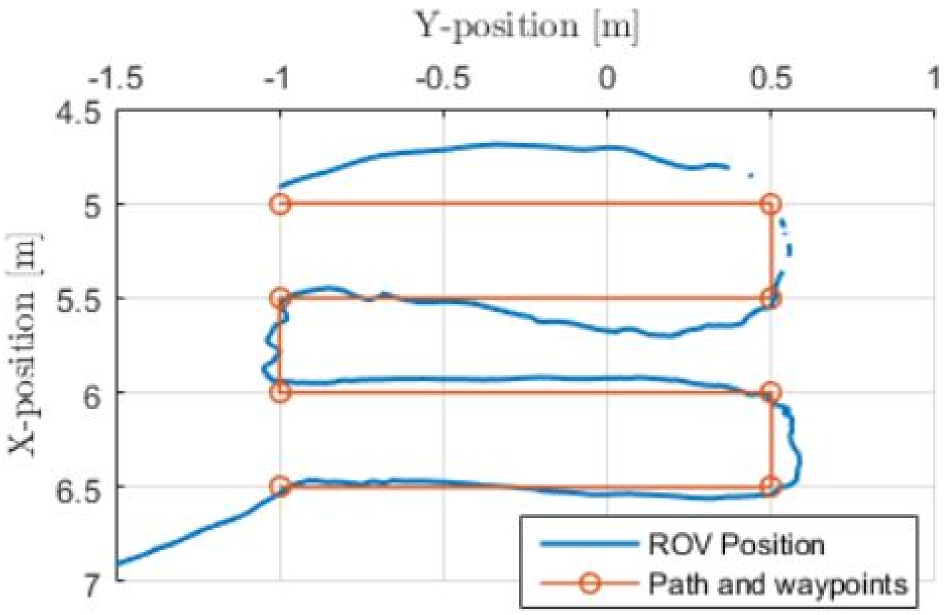

Figure 5.2: Position capture using Qualisys. Area: In the middle of the basin

5.2 Case 1: Depth Controller

Two depth control test cases were examined. The first test aims to show how the ROV manages to maintain the desired depth. In the second test, behavior related to movement between different depths is studied, including how these references are kept when moving in surge. As mentioned in Chapter 3, only the pressure sensor is used for depth measurements. This is because Qualisys data may be lost, while pressure sensor data always are available. In the depth plots to be presented, the red line shows the reference, the green line is the sensor measurement, and the blue line represents the error.

For the first depth test case, the ROV will simply attempt to maintain an arbitrary desired depth. This depth is chosen to be equal to the current depth, in order to study how the reference can be kept when not affected by any other motions. This means that the integrator term will increase and decrease too slowly to influence the controller, and that the ROV maintains the depth by only using a PD-controller. A red line shows that the desired depth is set and that the depth controller is on, while the absence of any red line implies that the depth controller is turned off. The results display an error peaking at ± 5cm, while mostly being either 0 or somewhere between ± 2,5cm. This is especially easy to observe in the last part of Figure 5.3, where the depth controller is active for about 15 seconds.

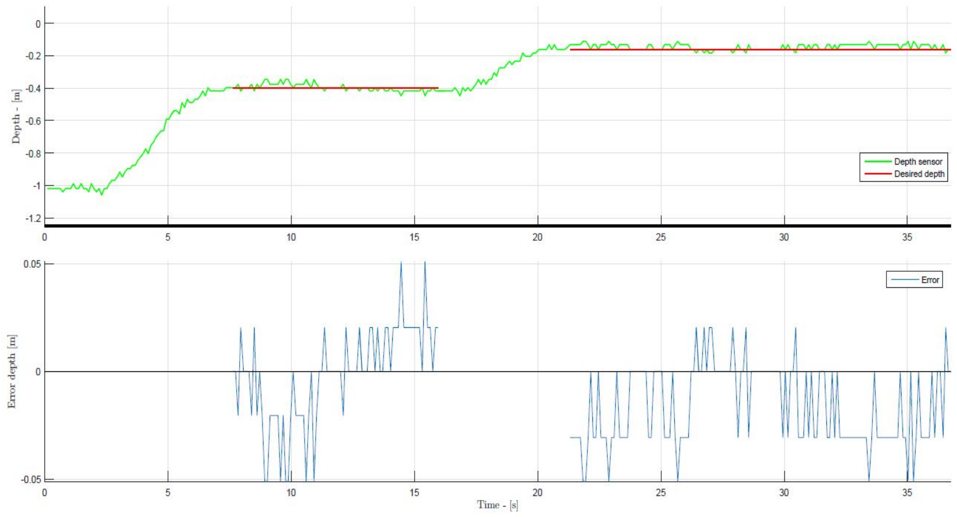

Figure 5.3: Desired depth: Keeping the reference

In the second test case, three desired depths were set to 0.3 m, 0.5 m and to 0 m, respectively. The depth plot is drawn from when the ROV was manually controlled in a lawnmower-like pattern, as can be seen in Figure 5.1 and Figure 5.2, while the depth controller was active. The heave motions are depicted in Figure 5.4, where the system quickly converges to the desired depth without overshooting. Some difficulties can be observed when the ROV attempts maintaining the depth while at the same time moving in surge. This is best seen as jumps occurring at timestamps around 8 and 12 seconds. Here, the measured error peaks at approximately 8 cm. Surge thrust is kept equal to 30% of maximum power throughout the entire experiment, which is the produced thrust force when performing automatic path-following.

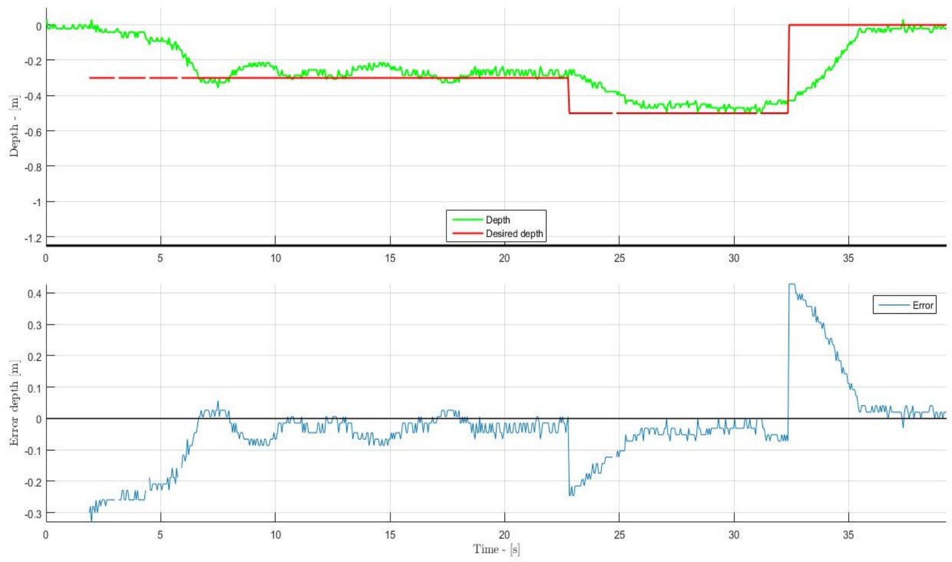

Figure 5.4: Desired depth: Reaching and keeping the reference

5.3 Case 2: Heading Controller

In order to follow a path, the desired heading should be both reached and kept with an error as small as possible. In the following, two interesting test cases will be presented. The first case studies how a slowly varying heading can be tracked and presents the expected deviation from the desired heading angle. In the second scenario, the focus is put on how the desired heading is reached and how the vehicle tracks a more quickly varying reference. In the second case, the ROV produces surge force equal to 30% of maximum power to generate a rapidly varying reference. Difficulties keeping the heading as desired during surge motions both increases and decreases the desired heading angle rate of change, as will be observed in Figure 5.6. This means that the ROV moves in a circular-like pattern. Testing shows that the expected deviation from the desired heading angle is lower when the craft moves at low surge velocities and when the angle rate of change is within certain limits (below ± 50 degrees per second, or $\pm 50\frac{deg}{s}$). More slowly varying heading angle references are, as anticipated, easier to track and maintain. Results from the testing are presented in Figures 5.5-5.7.

Slowly Varying Heading Reference

A heading controller must be able to ensure that a set-point or slowly varying heading reference can be maintained. The reference angle is calculated based on a desired position, which is placed far away. This position moves slightly to generate the slowly varying desired heading. While oscillating behavior is expected during heading control, the ROV occasionally refuse to produce force. This is observable after 30 seconds in Figure 5.5, and represent that thrust is not produced to minimize the error. The result shows that when the error is small, the thrust that is calculated is not successfully produced by the craft. This suggests that there are physical limitations in the thruster system.

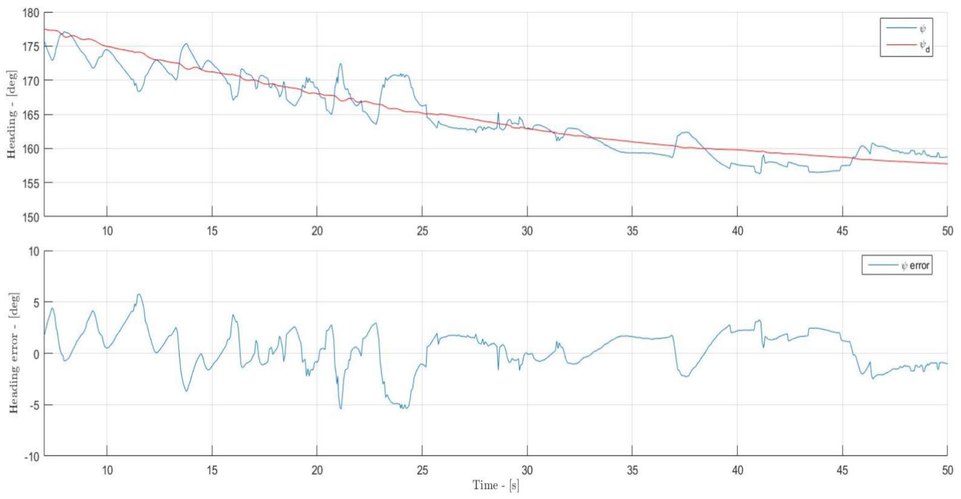

Figure 5.5: Tracking a slowly varying heading reference

Rapidly Varying Heading Reference

It is interesting to see how the system behaves when the reference is varying more rapidly. A desired course angle is calculated by tracking the location (x,y) = (5,-0.5), based on the methodology in Section 2.2. An offset close to $\psi_0 = 90°$ is added to the ROV's heading. By giving a constant surge thrust, the ROV will move in a circular motion, never actually reaching the desired location. How the reference is tracked can be seen in Figure 5.6, along with the position of the ROV in Figure 5.7.

Some overshoot and oscillatory behavior can be observed as the ROV tries to reach the desired heading for the first time. When reaching the reference after approximately t=6s, the desired heading is kept nicely. The angle rate of change in degrees per time unit is estimated to be near $25\frac{deg}{s}$ for $t \in [6s,\ 10s]$. Difficulties with keeping the reference emerges after t=10s, where the measured heading angle change over time is increased, finding itself at a rate closer to $50\frac{deg}{s}$.

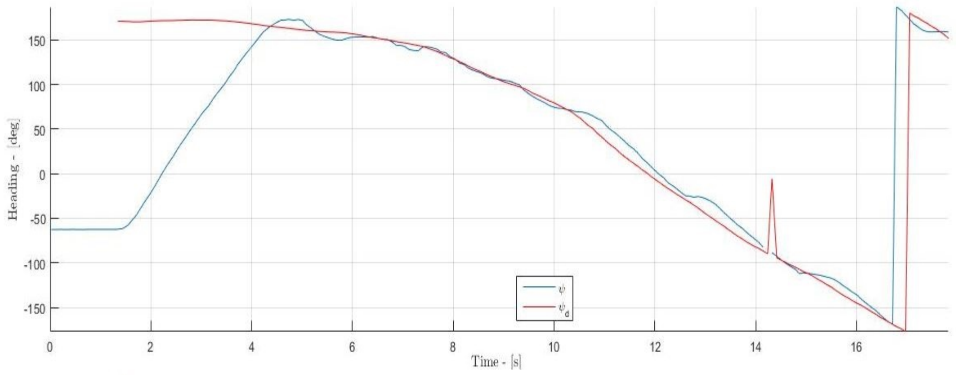

Figure 5.6: Tracking a constantly varying heading reference

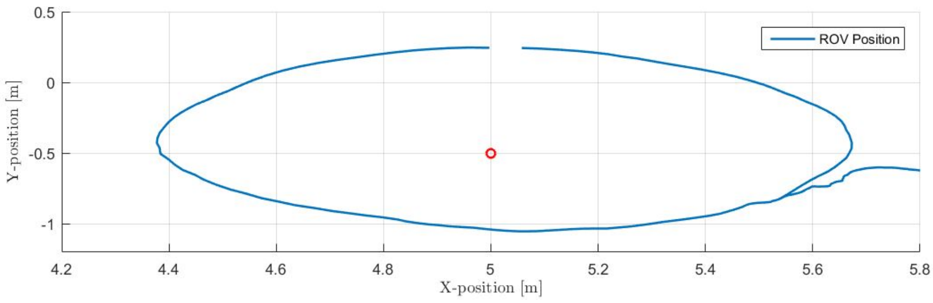

Figure 5.7: Position due to surge motion during heading reference tracking

5.4 Case 3: Underwater Path-Following

In the final case, LOS guidance is used to ensure path-following in 2-D with a depth controller to operate the depth. Two sets of waypoints will be presented, while several other waypoint configurations have been tested with great success using the developed LOS guidance system. The reason for choosing waypoints such as these is mainly related to the confined area where Qualisys underwater cameras are likely to identify the Videoray ROV marker object. In addition, the patterns are equal to those presented in the simulations, using a zig-zag pattern on the surface and an underwater lawnmower pattern.

5.4.1 Path-Following: Zig-zag Pattern

The first path-following objective is to follow a path taking a zig-zag shape in 2-D on the surface. This pattern is recognized as straight lines and sharp edges, i.e., the angles created by two lines intersecting in the connecting waypoint are small. This puts a lot of pressure on the ROV in terms cutting the corners and following the path. A typical response when attempting to follow this pattern is depicted in Figure 5.8.

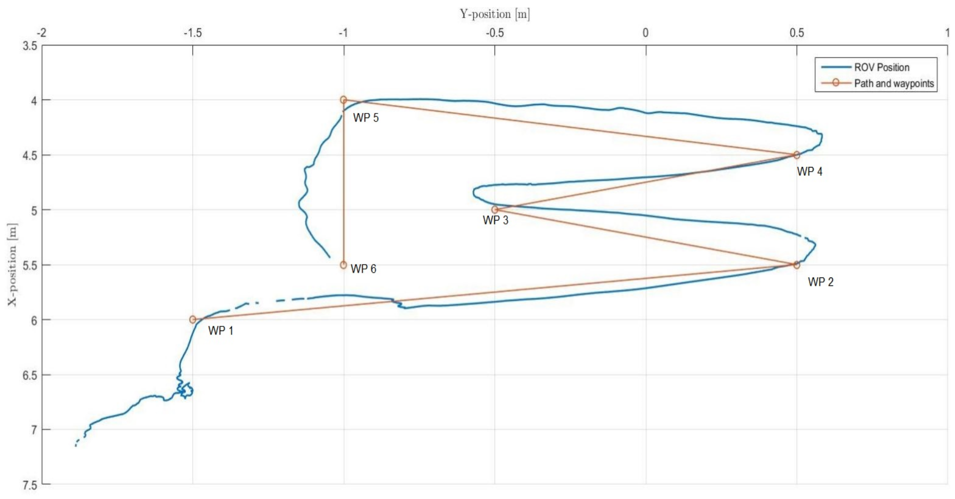

Figure 5.8: Path-following experiment: Zig-zag pattern

The ROV is stationed on the surface near (x, y) = (7, -1.9), and struggles slightly with maintaining the position data in the beginning of the run. However, when the markers are recognized properly, measurements are received and the path-following procedure begins. The path is difficult to follow perfectly, especially right after affected by position data drop-out. Velocity is attempted to be kept constant throughout the mission at 30$\%$ of maximum thrust, with the final waypoint being reached after about 35 seconds. Some wiggling can be observed, especially in the later stages of the run. All the waypoints are reached within a few centimeters, and the radius of acceptance is set to $R_{acc} = 10\ cm$ to better cut the corners and follow the next straight-lined path segment.

5.4.2 Path-Following: Underwater Lawnmower Pattern

In the final path-following case, the ROV follows a path constructed by a set of waypoints forming a lawnmower-like pattern below the surface. The radius of acceptance is chosen as $R_{acc} = 10\ cm$ for all waypoints except for the final waypoint, where it has been set to $R_{acc} = 5\ cm$. Focus is put on following the straight lines and to traverse the waypoints, while at the same time to keep an arbitrarily chosen desired depth set equal to 0.19 m.

Initially, the depth controller moves the ROV to the desired depth while the heading controller ensures that the desired course angle is maintained. The system then starts producing surge thrust. All the waypoints are reached except for the first waypoint, only being used to construct the path, by clocking in on the final waypoint location in approximately 40 seconds. The full response can be seen in Figure 5.9 on the next page.

It can be observed that the ROV motions are smooth. The vehicle moves in straight lines while both reaching the desired heading quickly and maintaining a sufficiently small heading error, i.e., below 5 degrees. This result presents the first successful underwater path-following developed for the Videoray Pro 4 ROV in the MC-lab using the QMT system. Overall, the waypoints are reached and traversed. The long path segments are better followed than the short line pieces. Staying on the short paths, however, seem to be slightly more difficult. Still, the overshoot here is small, never exceeding 10 cm, and emphasis is put on keeping smooth motions throughout the mission. The heading is plotted between -180 and +180 degrees, and the shortest turning angle is always chosen by the ROV, as described in Section 3.3.1.

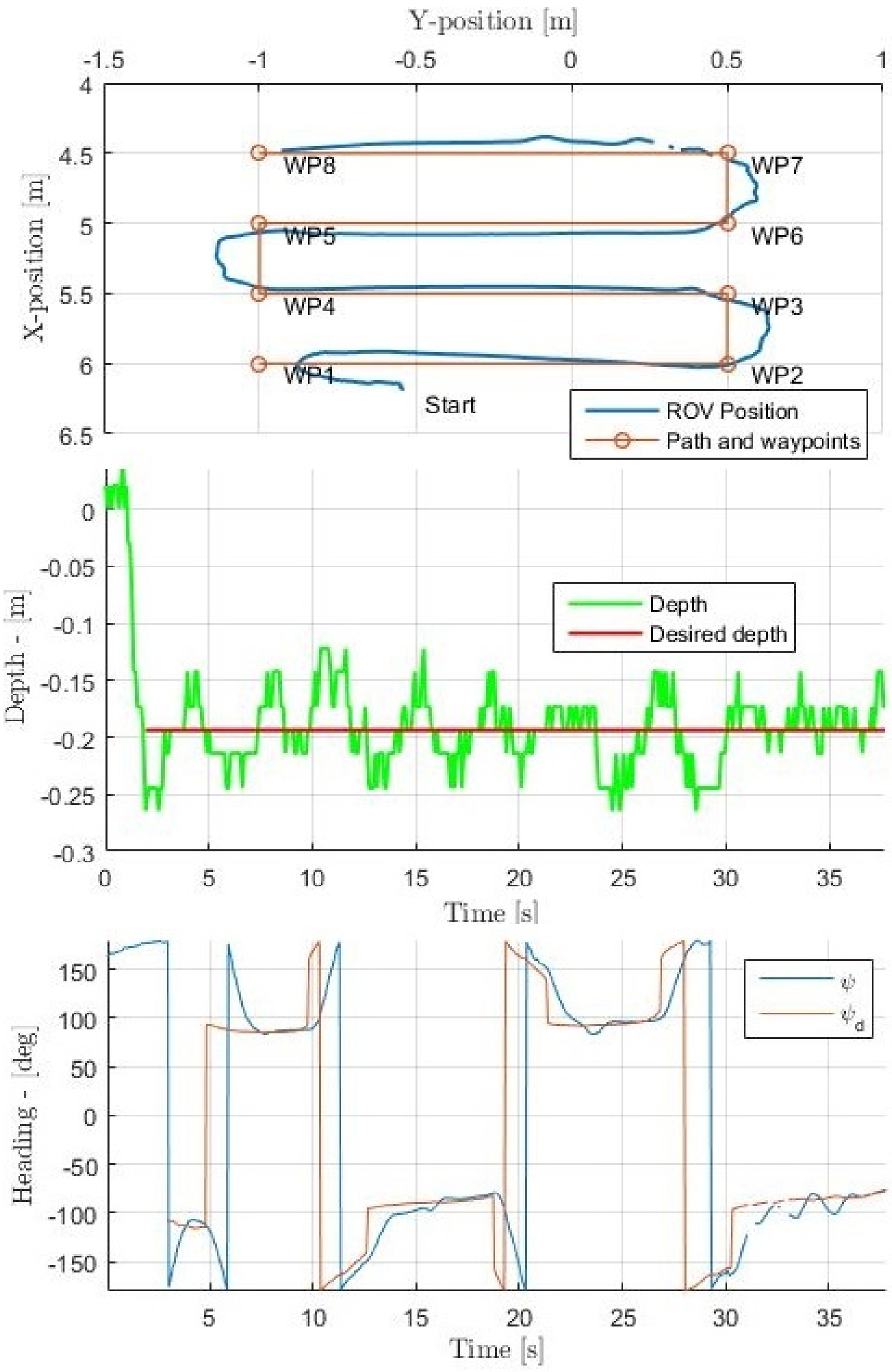

Figure 5.9: Path-following experiment: Underwater lawnmower pattern

CHAPTER 6 DISCUSSION OF RESULTS

In this chapter, the previously presented results and findings will be discussed in a broader context. An issue experienced with manually controlling the ROV will be examined first. This issue emerged in the later stages of the experimental testing, and has not received much attention. The chapter will continue with clarifying what to expect of the Qualisys underwater positioning system. This discussion will be based on the rig and marker set-up used for the Videoray Pro 4 ROV in the experimental testing in Section 5.1. The findings suggest that after a specific marker configuration has been chosen, adjusting camera settings and reducing the amount of light from above should be consulted. If struggling with receiving position data, rig and marker configuration will have to be reconsidered.

Both the simulation model and the Videoray ROV in experimental testing behave very similarly when controlling depth, heading and when performing path-following. By comparing the path-following performance, overshoots at the waypoint locations are observed in all cases. To what degree the two systems are comparable is difficult to measure, as some deviations are to be expected in the hydrodynamical parameter values. The discussion here aims to shed light upon the importance of having a simulation model being similar to the actual system, in order to obtain a meaningful outcome of the simulation testing.

Further on, the test cases will be discussed for the experimental testing, as this is the main contribution of the thesis. The challenges with achieving the desired response are governed by how thrust is limited by a *cartridge seal*, a small module responsible for lubrication of the propeller shaft. Each thruster has a *dead-band* range and a *ramp-up* function towards the commanded thrust. While these are supposed to be almost identical, they have proven not to be, which is linked to the cartridge seals containing different amount of lubricating oil. In physical terms, this means that the static friction force in the propeller shafts is different, but this is only an issue for the starboard thruster. Nonetheless, the path-following procedure is successfully carried out by the Videoray Pro 4 ROV below the surface in a highly satisfactory manner.

6.1 Manually Controlling the ROV

In the later stages of experimental testing, the joystick occasionally started misbehaving. What happens is that the Dualshock 3 controller suddenly freezes for a set amount of time. The issue has been looked into and attempted resolved, but no proper solution was found. Reading the joystick buffer more frequently actually increased the reaction time of the ROV and reduced the amount of freezes, but the problem did not disappear completely. The idea is that with the increasing features of computational power needed, the buffer of the controller builds up faster, as it is not being read fast enough to be kept "empty". When the buffer is filled up, the controller freezes and the software attempts reading all store commands until the buffer is empty again. This freezing process lasts for the same time duration at every occasion, and is estimated to approximately 3-3.5 seconds. Plots containing this phenomenon have not been successfully captured.

The freezing controller problem led to increased caution when manually operating the ROV. It is confirmed that the problem is related to the controller, as the system is able to perform path-following and other control strategies without the thrusters freezing. Possible error sources have not been given a complete attention. The main reason for this faulty behavior is assumed to be related to the software operating the joystick commands, including joystick buffer build-up and reading. Several hand-held controllers have been tested and the behavior was reproduced, making arguments for the problem being related to software and not hardware.

6.2 The Qualisys Underwater Positioning System

In Figures 5.1 and 5.2, the expected measurement outcome can be observed. Testing revealed that capturing the position between x = 3.5 and x = 6.5, y = -1.5 and y = 0.5, measurements are very consistent. Outside this area, the cameras hardly ever find the Videoray marker-defined object. The settings were altered for position data for x positions below x = 3.5, but this resulted in bad measurements. With the current spread on the markers and the marker size used, it is suggested that the area defined by x = 3.5 to 6.5, y = -1.5 to 0.5 is used for position based control methodologies, which gives an effective area for path-following of 6 m^2. While this area is small, it is large enough for performing path-following if waypoints are properly chosen. Considering Figures 5.1 and 5.2, the run was divided to better receive the full area where Qualisys can capture position. In addition, the run was divided for safety reasons, due to the limited length of the tether (20 m) and how the power supply had to be stationed.

It is important to note that the area is defined by the current calibration of the Qualisys system. The origin of the local NWU reference frame is located at the pool floor, placed approximately 5 meters from the end of the pool. It should be noted that the depth is measured from the surface by the pressure sensor, and that all Qualisys depth measurements are ignored. That is because the pressure sensor is good for measuring the depth, and as these measurements are never affected by drop-outs. Alternatively, Qualisys measurements may be used when they are available.

While the underwater Qualisys system performance is outstanding, with noise in the magnitude of a millimeter or less, measurements cannot always be received. Several factors impose communication loss between the markers and the Qualisys cameras. The most prominent difficulty is to capture the object in specific regions in the lab. Both challenges and suggested improvements regarding receiving position data are addressed as general guidelines in Chapter 3. Markers are best observed when keeping a close distance to the surface, as increasing depth will result in the ROV masking more of the markers. The performances from following zig-zag pattern in Figure 5.8 and the lawnmower path in Figure 5.9 have been compared. The two tests were performed in the same area with the same camera and light configurations.

Several marker configurations were tested, and one particular set-up gave the most consistent response. This marker set-up can be seen in Figure 3.3. While Qualisys occasionally struggles with capturing the markers, it does perform really well when enough markers indeed are found. Markers could also be placed on the ROV's top side, making the ROV visible whenever submerged, but the ROV's geometry may prevent the effectiveness of this method. The main contributions to improvements were provided by the extending arms for the rig, positioning the markers differently, turning off ceiling lights and adjusting the camera settings.

6.3 Comparing Simulation and Experimental Results

In both the model and the experimental test cases, measurements are assumed to be perfect. The simulation model does not successfully replicate the actual system, which is an argument against the comparison between simulation and experimental results. Running the ROV in a risk-free environment for verification of functionality and carrying out tests that run quickly will reduce the implementation and development time considerably. It does, of course, require a mathematical model of our system being "close enough" to the actual system. In the following, the most significant differences will be summed up.

Due to parameter value uncertainties, it is hard to make assumptions about badly and non-estimated parameters. The ROV also has a lateral thruster and a rig with markers that are not accounted for in the model. These increase the system mass and damping forces, and some additional damping was therefore manually added to the simulation model to better replicate the actual craft's motions. The ROV's tether is not accounted for in the model, but the impact made by the tether is assumed to be very small. The tether force can be neglected in most cases (available tether is in the water, not producing much strain on the ROV), but a minor unknown disturbance term may have been included in the model.

Not knowing the input-to-force relations for each of the thrusters restricts transferability and reduces comparison validity of the system. It should be mentioned that in-depth comparisons of the simulation model and the actual system may be more interesting when implementing model-based control functionality. Such functionality may include, for instance, backstepping controllers, feedback linearization and sliding mode control.

Several factors limit the experimental testing the simulation results to coincide perfectly, but the responses are very similar. Both systems control the depth and heading in analogous ways, and the paths are followed using the same methodology. Unmodeled tether and damping effects and assumptions about badly estimated parameters are likely to be the main differences between the model and the actual ROV. Additionally, the speed of the simulation model can be controlled in a more satisfactory manner, as compared to the actual ROV. Thrusters dead-band effects have not been modeled, but limit the craft from producing motions when being very small. When considering the results, the overall behavior is almost identical. The similarities in experimental and simulation results validate that the Videoray ROV successfully performs path-following and waypoint guidance. Some of the strange behavior exercised by the ROV suggests that there might be something wrong. The behavior is especially prominent when studying the occasional wiggling in the XY-plots in Figures 5.8 and 5.9. This may happen if the PID heading controller is not tuned perfectly, but most of the time it moves almost without any wiggling at all. This further suggests that tether forces are greater than anticipated, that other effects are present, and may also propose flaws with the thruster system. This issue is discussed further in Section 6.5.

6.4 Depth Controller

While the depth controller performs very well in both reaching and keeping a set reference, the measured error lies between $\pm$ 5 cm. When the ROV moves in surge while performing depth control, higher errors peaking at $\pm$ 10 cm are reached due to the thrusters not being placed in CG, resulting in the ROV pitching. How much the craft will pitch mainly depends on the relation between how much surge force is being generated multiplied with the distance to CG and the pitch restoration forces. The pitching effect is a highly undesirable response when no pitch controller exists, as the ROV has no proper way of controlling the pitch angle. Therefore, a higher surge force yields a faster growing pitch angle, meaning an increased difficulty in satisfying the depth objective. Feed-forward on the surge force may however improve the system slightly for smaller velocities, but this issue has not received much attention.

By considering the depth plots and the spikes in depth measurements, a significant amount of this spiky behavior must be due to noise. That is because a pressure sensor is used to estimate the depth, and pressure sensors are known to be prone to measurement noise. This noise is white noise with a zero mean with peaks hitting at $\pm$ 2.5-5 cm, and the challenge with keeping the reference mainly appears to be influenced by measurement noise. The difficulty with keeping the depth increases when the vehicle pitches during surge motion, and may also be slightly altered by the heave thruster dead-band, see Table 6.1.

Table 6.1: Heave thruster dead-band and ramp-up time

Thruster	**Thrust produced (%)**	**Dead-band area (%)**	**Thrust Ramp-up time**
Heave	Any	$\pm$ 5	Less than 0.5 seconds

Overall, the depth controller delivers as expected. Difficulties keeping the depth during surge motions are expected, but problems related to thruster dead-band and the measurement noise are difficult to overcome. Replacing the PID regulator with, for instance, an LQR or a sliding mode controller may improve the performance in terms of quicker and smoother convergence to the desired depth. However, the PID controller does perform very well here, and replacement should not be necessary. Equipping the ROV with additional weights may help overcoming the thruster dead-band to some degree. It seems that keeping the error below 2.5-5 cm is the best the system can achieve, while being slightly increased when producing surge motions. Filtering with second order low-pass filters was attempted, but this only created an undesired delay. If Qualisys data can be received consistently underwater, using Qualisys depth measurements when available may improve depth keeping performance. At least, using the Qualisys depth measurements will reduce the amount of measurement noise, as the error variance for Qualisys position estimates is in the magnitude of millimeters.

6.5 Heading Controller

The ability to follow a time-varying reference with approximately 5 degrees error proves that the system is suited for performing waypoint guidance. However, a 5 degree error is still large, and after extensive testing with heading controllers for the ROV, the craft was examined for hardware-related faults.

This examination lead to a surprising and unexpected finding. The ROV is equipped with cartridge seals with oil at all thruster locations, used for lubrication of the propeller shaft and for keeping water outside the pressure hull, seen in Figures 6.1 and 6.2.

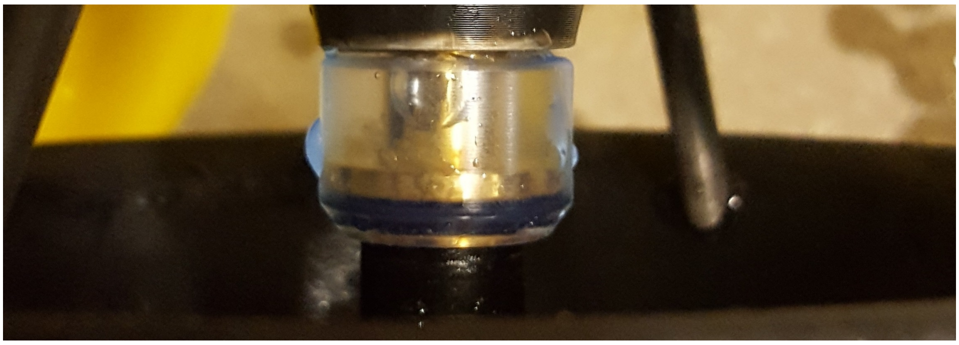

Figure 6.1: Cartridge seal: Air bubble "half full" in the port thruster

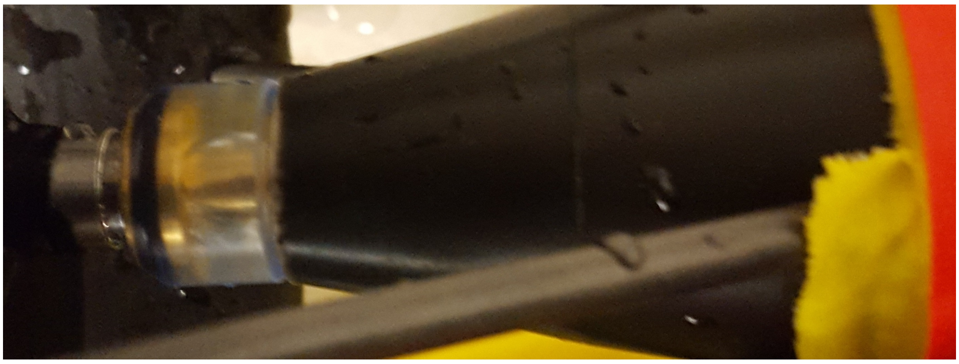

Figure 6.2: Cartridge seal: Air bubble "full" in the starboard thruster

The bubble is air, and its size determines how much oil is left. When the bubble covers 3/4 of the distance between each seal wall, the seal has to be changed immediately, or water may soon enter the thruster. What has happened here is that when the ROV has been used, enough care has not been taken by the users to check the seals. Videoray developer Andrew Goldstein states that the approximate life-span of a cartridge seal is unknown, and that some users have reported 100 hours, while others have experienced closer to 2-300 hours. How long the ROV has been in use is unknown, but that it has less than 100 run hours is for sure. Considering the starboard cartridge seal in Figure 6.2, it seems that this is causing the difficulties in controlling the starboard thruster, and hence, the heading. The main implication of missing oil and the big air bubble size is increased thruster dead-band and ramp-up time, in addition to possible oxidation. If this faulty behavior was investigated at an earlier instance, experimental results would most likely not have been carried out in this extent, in danger of damaging thruster and ROV.

Even though the heading error can be kept below 5 degrees as seen in Figure 5.5, especially for slowly varying or constant heading angles, even better performance could be achieved. The two most important and limiting properties of the thrusters that are affected here are the thruster dead-band and thrust ramp-up. In robotics, the dead-band represents a value that cannot be produced by the thrusters. What here has been called thrust ramp-up refers to how the speed of the propeller shaft is building up towards the commanded thrust value. This means that the commanded force is not produced right away, which otherwise would create wear and tear on the thrusters. Simple testing has shown that both the dead-band and the thrust ramp-up time for the port and starboard thruster varies slightly, summarized in Tables 6.2 and 6.3 below.

Table 6.2: Main thrusters dead-band and ramp-up time: producing low thrust

Thruster	**Thrust produced (%)**	**Dead-band area (%)**	**Thrust Ramp-up time**
Starboard	± 5-25	± 5	0.5-2 seconds
Port	± 12-25	± 12	Less than 0.5 seconds

Table 6.3: Main thrusters dead-band and ramp-up time: producing high thrust

Thruster	**Thrust produced (%)**	**Dead-band area (%)**	**Thrust Ramp-up time**
Starboard	Above 25 / Below -25	± 5	About 1-1.5 seconds
Port	Above 25 / Below -25	± 12	About 1-1.5 seconds

Performing simple testing with commanding the thrusters to produce an equal amount of surge thrust, the starboard thruster takes a while longer to respond. This means that the starboard thruster's ramp-up time is higher compared to the port thruster, and this time duration has proven to be longer when commanding a low thrust. The longer ramp-up time and increased dead-band for the starboard thruster are supported by the amount of oil left in the cartridge seal. It will take a little longer before the shaft is lubricated and until the surface is smooth enough for spinning. This means that heading control for small angles, i.e., when the yaw momentum to be generated is small, the port thruster will produce the designated thrust before the starboard thruster. Occasionally, the starboard thruster will not be able to produce thrust at all. The physical explanation to this phenomenon is that the static friction force in the starboard propeller shaft is larger compared to the port thruster shaft. Keeping the heading with a lower error than ± 5 therefore seems difficult. However, it is argued that maintaining the heading error within this area should be satisfactory in order to follow paths. In addition, it should be noted that the system always chooses the optimal turning angle and that the reference angle is reached quickly (see Section 3.3.1).

6.6 Underwater Path-Following

6.6.1 Zig-zag Pattern

Overall, the ROV's position deviates slightly from the path when following the zig-zag pattern. All of the waypoints are reached within an error smaller than the radius of acceptance, which concludes that the path-following waypoint guidance is successful. However, the zig-zag pattern case is very interesting, as deviation from the desired path makes arguments for path evaluation even for simple straight-lined paths. While it is desired to follow straight lines, smoother trajectories are taken by the ROV instead of strictly tracking the line segments. It is obvious to see this when studying the ROV's behavior as it closes in on a waypoint and the associated radius of acceptance, which triggers the next waypoint to be chosen as the desired location. While the ROV attempts to keep a constant thrust (velocity) on the path, the largest deviations from the path are due to the sharp corners. This suggests that sharper corners should have a higher radius of acceptance value in order for the ROV to change desired waypoint sooner, meaning that the actual path can be better followed and with less overshoot. The overshoot when reaching the waypoint is governed by the velocity of the craft being too high as it closes in on the waypoint, and overcoming thruster dead-band effects to keep a lower speed is very difficult. Slowing down the velocity before reaching the waypoint may result in the thruster attempting to produce thrust within the dead-band value. Therefore, the "constant thrust" technique has been applied, with focus on having a low thrust value. Keeping a low thrust is necessary for the vehicle to avoid pitching as well. To achieve less overshoot, it is possible to command slightly negative thrust right after the waypoint was traversed, but this was not attempted. The overshoot problem is presumed solved when also making use of the lateral thruster and regulate the cross-track error to zero, in addition to replacing the cartridge seal to allow control of lower craft speeds.

Not only does the zig-zag pattern present how an arbitrary straight-lined path is followed, but also how the radius of acceptance influences the ROV's capabilities in following the path with the constant thrust (velocity) control action. This does not only go for straight-lined paths, but seen in conjunction with the heading controller, how circular-shaped paths can be followed. Better performance could for instance be achieved if compromising to a lower radius of acceptance and if velocity could be controlled in a more satisfactory manner. Estimating the velocity and using it with a variable speed controller could be used to reduce the overshoot at the waypoints. By considering the remaining waypoints, the radius of acceptance may, for instance, be chosen as a function of the ROV's velocity to better smooth out the path. However, heading control and turning maneuvering aside, using the lateral thruster to close in on the path would ensure a smaller path deviation, as previously mentioned.

6.6.2 Underwater Lawnmower Pattern

The lawnmower path is better followed than the zig-zag pattern. This is related to the fact that the corners are blunter, resulting in the ROV having an easier time recovering from the overshoots. Another factor that may determine whether the path is easily followed has to do with the distance between each of the rows. The overshoots mainly occur where the row length is short. As each row is 0.5m long, having a larger row distance at for instance 1 m would reveal that better results may be achieved. However, attempting to perform path-following for a square pattern proved that this had little impact, as can be seen in Figure 6.3. The radius of acceptance still allowed the ROV to follow the consecutive path segments nicely. Overshoots due to the velocity being too high continue to haunt the system's path-following performance. Being forced to a high velocity and the thruster dead-band are held as responsible causes for most overshoots and path deviations. As previously stated, deviations from the path could be decreased by regulating the cross-track error to zero with the lateral thruster. Additionally, arguments can be made that the heading controller is not reacting fast enough to the change in desired heading. With the heading controller gains being high, especially for the proportional gain, it should be fast enough (see for instance the heading plot in Figures 5.6 and 5.9). Producing thrust within the dead-band range may however be the case here. Cable drag is not likely to be the reason for the overshoot, as some of the waypoints are reached almost perfectly, where barely any overshoot or path deviation can be observed at all.

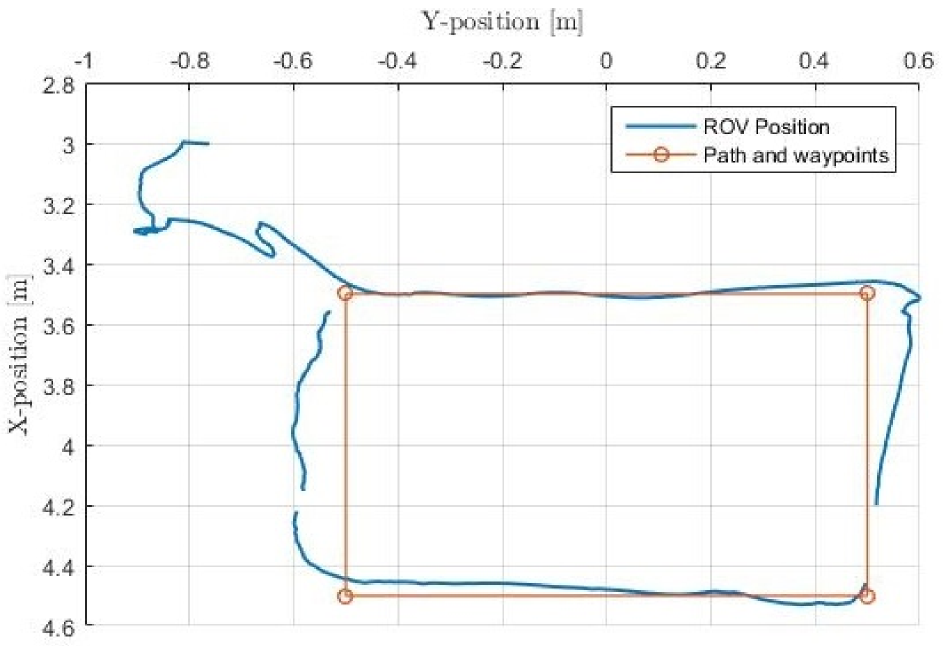

Figure 6.3: Underwater path-following for a square-shaped pattern

Again considering the lawnmower pattern in Figure 5.9, WP 5 is not successfully reached due to the radius of acceptance. The vehicle is here moving along the path being approximately 5 cm away from the path segment in x-direction. Some difficulties reaching the desired heading is observed in the time-frame 30-35 seconds. Qualisys measurement dropouts are obviously contributing to this error, imposing a tough challenge for the regulator and, thereby, an unexpected deviation from the path. Additionally, the system struggles slightly with keeping the depth. As discussed earlier, this is expected to happen during surge motions.

6.6.3 General Remarks and Observations

The velocity used to converge to the waypoint locations has been kept as low as possible to reduce the impact of overshoots. Therefore, the radius of acceptance is chosen a bit higher than desired, to compensate for the overshoots at the waypoints. The overshoots at the waypoints are important, because they are the main reason why the craft deviates from the path. In addition, difficulties keeping the desired heading imply that a higher radius of acceptance could have be chosen. A lower R_{acc} will, on the other hand, result in a lower probability that the ROV reaches the waypoint.

For achieving even smoother ROV motions and trajectories, the system could compromise with allowing the ROV to produce surge thrust when the heading angle error is greater, e.g., for an error below 7 degrees instead of 5 degrees. This means that less focus is put on following the actual path, but that traversing waypoints can be carried out in a quicker and smoother fashion. On the other hand, less deviation from the path could be achieved by restricting the threshold for when to produce surge. For instance, the limit may be constrained to 3 degrees, or maybe even 2 degrees, depending on how well the system is able to maintain a desired heading angle. However, this depends on the dead-band on the thrusters, especially affecting the starboard thruster, and an error of 5 degrees or less seems reasonable when considering the results.

Overall, the experimental results with performing path-following for both the zig-zag and the lawnmower pattern are very satisfying. The conclusion is that the ROV is able to follow any path consisting of straight-lined path segments. By manipulating the heading controller to take in a more time-varying reference and after replacing the cartridge seal, any arbitrary path can, in theory, be followed. More specifically, the heading can be controlled directly as a preliminary constraint for producing surge thrust. This further implies that tracking functionality, i.e., both dynamical (temporal) and geometrical constraints, can be implemented and fully adopted by the Videoray ROV.

CHAPTER 7
CONCLUSIONS AND SUGGESTIONS FOR FURTHER WORK

In the final chapter, conclusions are drawn based on all the work presented in the previous chapters. The conclusion makes a technically detailed and concise summary of the discussion made in Chapter 6, and will present the contributions that have been made in the thesis work. Finally, the chapter ends with proposals for further work with the Videoray Pro 4 ROV.

The path-following methodology built on lookahead-based LOS guidance is fully adopted by both the simulation model and the actual ROV. Different dead-band values for the starboard and port thruster are unfortunate, being the main factor preventing improved results to be achieved in the experiments. Nonetheless, the implemented controllers and LOS-based waypoint guidance system successfully achieve underwater (3-D) path-following in the MC-lab using the QMT system. The suggestions for further work in the final section present some interesting topics that may aid in achieving enhanced and more satisfactory control of the ROV. Additionally, this section provides ideas about features that can be inhabited by the small-sized craft in the future, and how these may be of significant value in a wider context.

7.1 Conclusions

In the following, conclusions are drawn based on information presented in the previous chapters. Experimental results show that the Videoray Pro 4 ROV achieves underwater path-following, and represents this master's thesis' scientific contribution. While a lot of work often is concerned with theory, this is a practical and experimental achievement, where the theory successfully has been applied. Even though complications were encountered, the system manages to satisfy the underwater path-following control objective, which has been both the main scope and the main accomplishment of this thesis.

First and foremost, a note should be given to the user interaction unit. This unit represents a hand-held controller, and can be used to manually steer the ROV. As described in Section 1.2.2, code for the hand-held controller was already developed at the beginning of the thesis work. During experiments with the ROV, a problem was experienced where the joystick software occasionally froze in the middle of an operation. This frozen state of the controller lasted for about 3.5 seconds, and limited manual control. When this happened, the thrusters produced the command that was last stored in the buffer of the controller for the before-mentioned time duration. Attempts to reproduce the phenomenon give ideas that the error could be linked to how the buffer stores and reads commands. The issue is presumed solved by first gaining access to the buffer, before finding a way to handle buffer command stacking and reading (popping) in an efficient way.

The lateral thruster communicates in a different fashion compared to the other thrusters, in a so-called round-robin fashion, or in a 1/2 duplex mode. Documentation on this thruster is scarce, and how to communicate with all thrusters at the same time is not addressed. Close contact with the main developer of this system has been kept since January, and sufficient help for achieving seamless control of all thrusters simultaneously was achieved in the later stages of the thesis work. Therefore, no experimental work with the lateral thruster has been carried out in the MC-lab. Code for controlling the lateral thruster using the hand-held controller has been developed, and can be used in further work with the ROV.

A method for using the Qualisys underwater positioning system in Matlab has been further developed, along with a guide for how to implement this in Linux OS. For Mac and Windows OS, this functionality already exists. The positioning system lets users extract an object's position and orientation in a straight-forward manner, with small delay and high accuracy. Position data are occasionally lost, but requires that the camera settings are set according to the guidelines and that markers are placed properly for object definition and recognition. To minimize the impact of data drop-out for fractions of a second, using the previously measured position if it was measured less than one second ago has proven to work very well. When position data is unavailable for longer than a second, the system waits for a response from Qualisys before performing an action. Overall, for a big region in the middle of the pool, measurements are received quite consistently. What this means is that the ROV never needs to wait for receiving position data, because they will always be available. Position and orientation data should be available whenever the markers and camera settings are set according to guidelines.

A simulation model of the ROV has been created using Matlab and Simulink. Both this simulation model and the Videoray Pro 4 ROV adopt PID control laws for heading and depth, and a lookahead-based LOS guidance law for path-following. Simulation and experimental results are great, presenting successful underwater waypoint guidance for straight-lined paths. The heading of the ROV is somewhat difficult to properly control. This is due to the starboard thruster's dead-band range being high and different from the port thruster. Even with this faulty behavior, the path can be followed quite nicely. The dead-band of this thruster was noticed as a problem in the later stages of experimental work with the ROV. Investigation of the cartridge seal, responsible for lubricating the propeller shaft, showed that the seal is worn out. There is not sufficient oil left to properly lubricate the shaft, meaning that the thruster struggles with overcoming the propeller shaft's static friction. In addition, there is not enough oil to withstand water from entering the pressure hull. While it seems that no water has entered the hull of the vehicle, the system seems to be in good shape, but the ROV should not be placed in the water before this cartridge seal is replaced. This seal is also prone to oxidation, and new cartridge seals have been ordered for replacement. It should be noted that, currently, this is only a problem for the starboard thruster cartridge seal, and neither the port nor heave thruster seals need to be changed yet. Here, the air bubbles are only halfway through the capsule.

It should be noted that the intention behind the simulation model was to simulate the expected response, which has been compared with the experimental response to verify that the experimental behavior was as anticipated. It is argued that the ROV has the potential to follow any arbitrary path, as it can change its heading on the spot. The scientific contribution is concluded by the implemented LOS path-following methodology in a Videoray Pro 4 ROV, used in experimental trials to successfully realize underwater waypoint guidance.

7.2 Further Work

Before any further work can take place, the starboard thruster cartridge seal has to be replaced. The other cartridge seals may be replaced as well, but this should not be necessary.

Full control of the lateral thruster was achieved in the end of the thesis work. In future work with the Videoray Pro 4 ROV, the lateral thruster can be used to realize smaller deviations from the path. More specifically, the impact of overshoots at waypoints can be reduced by controlling the relative cross-track error to zero with the lateral thruster. The thruster can also be controlled by the hand-held controller, see Figure 1.3. In addition, when speaking of the controller, the freezing issue should be looked into. This is likely to be related to software and not hardware, as the problem occurs for several different controllers.

The camera of the Videoray ROV is currently not operable, due to a missing connecting piece. When this module is put in place, it should be straight-forward to activate the camera functionality. Camera-based control can be used with, e.g., path-following, obstacle avoidance, target tracking and constant distance keeping to target. Using cameras for control is becoming a hot and growing technology with great potential (see, for instance, the work of Candeloro et al. (2015)). Another topic with increasing popularity within robotic control and mapping is called simultaneous localization and mapping (SLAM). It is concerned with constructing and/or updating a map of an unknown environment, and is probably better known as the technique used by the Google car for creating an interactive map. Below the surface, the use of SLAM techniques is limited by available localization sensors and the accumulation of error over long term operations (Hidalgo and Bräunl, 2015). SLAM and more simple camera-based control methods may also be used for seabed mapping, localization of ship wrecks or other items, inspection of pipelines and fishing nets, and so on.

Creating a new rig to hold the Videoray ROV in place can be used for estimating a new set of hydrodynamical parameter values in the MC-lab. These numerical values should better mimic the actual system. More accurate parameter values for the simulation model can then be used for testing and verification of model-based control strategies, before being implemented into the real system. With the Qualisys underwater positioning system, testing to verify the implemented control methods is encouraged. The rig will also aid in finding the mathematical relationship between produced force and commanded input, this being highly beneficial for the control and thrust allocation systems. An appropriate set of hydrodynamical parameters can be used for dead-reckoning as well, when position data are lost. Having the QMT system encourages the development of estimators and observers, as estimations easily can be compared to actual measurements. This may have a greater impact on the further development of control methods where the availability of measurements may be limited. In addition, using reference models to generate traceable trajectories has not been investigated, but may increase performance by all systems that deal with control of position and orientation (Fernandes et al., 2015).

BIBLIOGRAPHY

Aghili, F., June 2012. A prediction and motion-planning scheme for visually guided robotic capturing of free-floating tumbling objects with uncertain dynamics. IEEE Transactions on Robotics 28 (3), 634–649.

Anderson, E., Beard, R., McLain, T., May 2005. Real-time dynamic trajectory smoothing for unmanned air vehicles. Control Systems Technology, IEEE Transactions on 13 (3), 471–477.

Balchen, J. G., Andresen, T., Foss, B. A., 2003. Reguleringsteknikk.

Bennett, S., Jun 1996. A brief history of automatic control. IEEE Control Systems 16 (3), 17–25.

Blouin, S., Heard, G. J., Pecknold, S., May 2015. Autonomy and networking challenges of future underwater systems. In: Electrical and Computer Engineering (CCECE), 2015 IEEE 28th Canadian Conference on. pp. 1514–1519.

C. L. Bottasso, D. L., Savini, B., Nov 2008. Path planning for autonomous vehicles by trajectory smoothing using motion primitives. IEEE Transactions on Control Systems Technology 16 (6), 1152–1168.

Candeloro, M., Valle, E., Miyazaki, M. R., Skjetne, R., Ludvigsen, M., Sørensen, A. J., Oct 2015. Hmd as a new tool for telepresence in underwater operations and closed-loop control of rovs. In: OCEANS 2015 - MTS/IEEE Washington. pp. 1–8.

Chen, C., 1984. Linear system theory and design. HRW series in electrical and computer engineering. Holt, Rinehart, and Winston.
URL https://books.google.no/books?id=0gFRAAAAMAAJ

Christ, Wernli, S., Dec 2013. The ROV Manual: A User Guide for Remotely Operated Vehicles. 2nd Edition, Butterworth-Heinemann.

Dahl, A. R., 2013. Path planning and guidance for marine surface vessels. Master's Thesis, NTNU.

Dubins, L. E., 1957. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. American Journal of Mathematics 79 (3), 497–516.
URL http://www.jstor.org/stable/2372560

Eidsvik, O. A., 2015. Identification of hydrodynamical parameters for remotely operated vehicles. Master's Thesis, NTNU.

Fernandes, D. d. A., Sørensen, A. J., Donha, D. C., 2015. Path Generation for High-Performance Motion of ROVs Based on a Reference Model. Modeling, Identification and Control 36 (2), 81–101.

Fossen, T. I., 2011. Handbook of marine craft hydrodynamics and motion control. John Wiley and Sons.

Frazzolini, E., Dahleh, M. a. F. E., 2002. Real-time motion planning for agile autonomous vehicles. Control Systems Technology, IEEE Transactions on 25 (1), 116–129.

Fritsch, F. N., Carlson, R. E., 1980. Monotone piecewise cubic interpolation. SIAM Journal on Numerical Analysis 17 (2), 238–246.

From, P. J., Gravdahl, J. T., Abbeel, P., May 2010. On the influence of ship motion prediction accuracy on motion planning and control of robotic manipulators on seaborne platforms. In: Robotics and Automation (ICRA), 2010 IEEE International Conference on. pp. 5281–5288.

Hidalgo, F., Bräunl, T., Feb 2015. Review of underwater slam techniques. In: Automation, Robotics and Applications (ICARA), 2015 6th International Conference on. pp. 306–311.

Jekeli, C., 2001. Inertial navigation systems with geodetic applications. Berlin: Walter de Gruyter.

KumarRobotics/qualisys source code, [Accessed 10-March-2016]. Ros-community. https://github.com/KumarRobotics/motion_capture_system.

Lekkas, A. M., 2014. Guidance and Path-Planning Systems for Autonomous Vehicles. PhD Thesis, NTNU.

M. Candeloro, A. M. Lekkas, A. J. S., Fossen, T. I., 2013. Continuous curvature path planning using voronoi diagrams and fermat's spirals. In 9th IFAC Conference on Control Applications in Marine Systems, Osaka, Japan.

MathWorks, [Accessed 10-March-2016]. Create custom messages from ros package. http://se.mathworks.com/help/robotics/ug/create-custom-messages-from-ros-package.html.

MSS, 2010. Marine Systems Simulator. Viewed 25.04.2016.
URL http://www.marinecontrol.org/

Pettersen, B., 2007. Kompendium, Hydrodynamikk. Department of Marine Technology.

Roberts, G. N. (Ed.), 2006. Advances in Unmanned Marine Vehicles. Control, Robotics amp; Sensors. Institution of Engineering and Technology.
URL http://digital-library.theiet.org/content/books/ce/pbce069e

Skjetne, R., 2005. The maneuvering problem. PhD-thesis, 2005:1.

Skjetne, R., Smogeli, Ø. N., Fossen, T. I., 2004. A nonlinear ship manoeuvering model: Identification and adaptive control with experiments for a model ship. Modeling, Identification and control 25 (1), 3.

SNAME, 1950. The society of Naval Architects and Marine Engineers. Nomenclature for Treating Motion of a Submerged Body Through a Fluid. In: Technical and Research Bulletin No. 1–5.

Sørensen, A. J., 2012. Marine control systems propulsion and motion control of ships and ocean structures lecture notes.

Tsourdos, A., White, B., Shanmugavel, M., 2010. Cooperative path planning of unmanned aerial vehicles. John Wiley and Sons, Ltd, pp. 1–27.
URL http://dx.doi.org/10.1002/9780470974636.ch1

Vassev, E., Hinchey, M., Aug 2013. Autonomy requirements engineering. In: Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on. pp. 175–184.

Videoray homepage, Accessed: 2015, November 23rd. http://www.videoray.com/.

Zermelo, E., 1931. Über das navigationsproblem bei ruhender oder veränderlicher windverteilung. ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik 11 (2), 114–124.
URL http://dx.doi.org/10.1002/zamm.19310110205

APPENDIX A THE VIDEORAY PRO 4 ROV

Previously presented and derived information about the Videoray Pro 4 ROV are summarized here. In addition, the model parameters that are used in the simulation model have been given, along with the equations of motion and an overview of the Simulink model.

A.1 Thruster Configuration

The thruster configuration of the Videoray Pro 4 ROV can be seen in Figure A.1 below. These figures refer to Figures 1.4 and 3.7, but are restated here for convenience

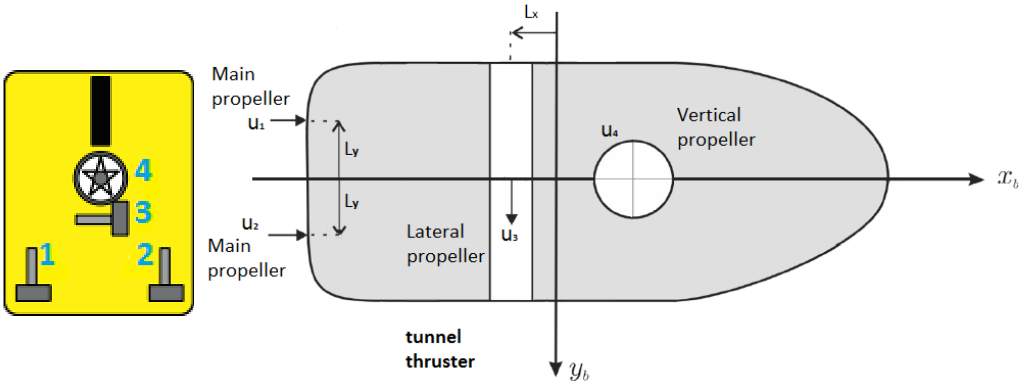

Figure A.1: Videoray Pro 4 ROV thruster configuration

A.2 Model Parameters

The hydrodynamical coefficents and model parameters for the Videoray Pro 4 can be seen below. They are intended for use when we wish to develop model-based control strategies. This information has been extracted from Eidsvik (2015). Simple assumptions have been made to choose values close to the actual values of the ROV where it is unclear what the actual values may be, only when looking at the tests that were performed.

Table A.1: Hydrodynamic derivatives and damping coefficients

Added mass	**Linear damping**	**Quadratic damping**
$X_{\dot{u}} = 2.5028\ kg$	$X_u = 2.31\ kg/s$	$X_{\lvert u \rvert u} = 18.5741\ kg/m$
$Y_{\dot{v}} = 7.1401\ kg$	$Y_v = 5.0326\ kg/s$	$Y_{\lvert v \rvert v} = 29.4535\ kg/m$
$Z_{\dot{w}} = 8.0125\ kg$	$Z_w = 6.6261\ kg/s$	$Z_{\lvert w \rvert w} = 60.139\ kg/m$
$N_{\dot{r}} = 0.0395\ kg\ m^2$	$N_r = 0.0288\ kg\ m/s$	$N_{\lvert r \rvert r} = 0.052\ kg\ m/s$

Table A.2: Weight and balance data

The weight and balance data has been found through simple analyses of the vehicle, and may not be exact. This data is based on a CAD model of the ROV in Solidworks made by Eidsvik (2015).

$x_g = 0.00\ m$	$x_b = 0.00\ m$	$m = 6.9\ kg$
$y_g = 0.00\ m$	$y_b = 0.00\ m$	$I_z = 0.0521\ kg\ m^2$
$z_g = 0.10\ m$	$z_b = 0.14\ m$	

A.3 Equations of Motion

The Videoray Pro 4 ROV equations of motion (4 DOF) can be written as

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \boldsymbol{R}(\psi)\boldsymbol{\nu} \\ \mathrm{M}\dot{\boldsymbol{\nu}} &= \boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} - \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} \end{aligned} \tag{A.1}$$

where the **M, C** and **D** matrices can be written as

$$\mathbf{M} = \begin{bmatrix} m + X_{\dot{u}} & 0 & 0 & 0 \\ 0 & m + Y_{\dot{v}} & 0 & 0 \\ 0 & 0 & m + Z_{\dot{w}} & 0 \\ 0 & 0 & 0 & I_z + N_{\dot{r}} \end{bmatrix} \tag{A.2}$$

$$\mathbf{C}(\nu) = \begin{bmatrix} 0 & 0 & 0 & -mv + Y_{\dot{v}}v \\ 0 & 0 & 0 & mu - X_{\dot{u}}u \\ 0 & 0 & 0 & 0 \\ mv - Y_{\dot{v}}v & -mu + X_{\dot{u}}u & 0 & 0 \end{bmatrix} \tag{A.3}$$

$$\boldsymbol{D}(\boldsymbol{\nu}) = \begin{bmatrix} X_u + X_{|u|u}|u| & 0 & 0 & 0 \\ 0 & Y_v + Y_{|v|v}|v| & 0 & 0 \\ 0 & 0 & Z_w + Z_{|w|w}|w| & 0 \\ 0 & 0 & 0 & N_r + N_{|r|r}|r| \end{bmatrix} \tag{A.4}$$

A.4 Simulink Model

The full mathematical model of the ROV has been created in Simulink, and an overview of the model can be seen in Figure A.2. A set of predefined waypoints is sent from the path planning system (red). The guidance system (light yellow) receives the waypoints along with the current position, estimates the position error and reference trajectories based on LOS guidance and feeds them to the control system (green). By using the position and velocity, the control system calculates necessary thruster force in order to track the reference signals from the guidance system. When the thruster forces are calculated, they are forwarded to the Videoray Pro 4 ROV kinetics and kinematics block (slightly darker yellow), responsible for producing the desired forces and providing new estimates for position and velocity. These estimates are made based on the mathematical relationships described in the previous Appendix section. Additionally, an observer block (blue) allows for state estimation.

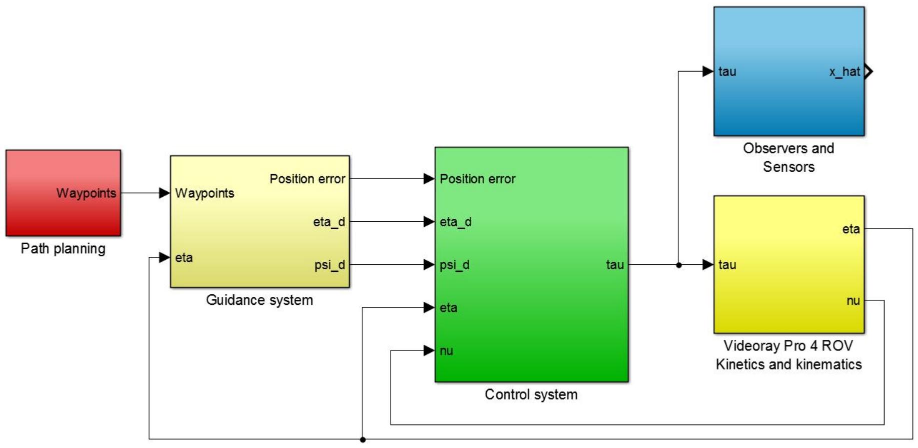

Figure A.2: Simulink model: System model design

APPENDIX B
RECEIVING QUALISYS DATA IN MATLAB

Appendix B describes how to receive Qualisys data in Matlab. This method is developed for computers that run on Linux OS. For Windows users, mex-functionality exists and may also be developed for private use. This method has been successfully used to receive Qualisys data measurements in Matlab R2016a on a computer running Linux OS. Figure B.1 below figuratively depicts the communication flow between Matlab and Qualisys.

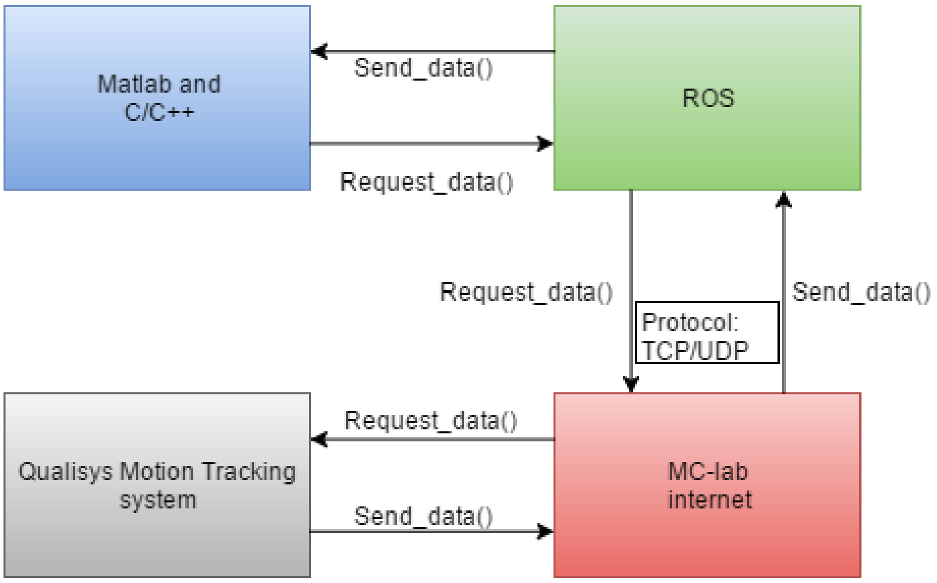

Figure B.1: Communication flow between Qualisys and Matlab

B.1 Installing ROS

Installing Robotic Operating System (ROS) can be done by simplify following the guide at $http : //wiki.ros.org/indigo/Installation/Ubuntu$. The reason for installing ROS is that ROS already has functionality for communication with the Qualisys Motion Tracking (QMT) system and Matlab/Simulink.

B.2 MC-lab Manual: Importing Data from Qualisys into MATLAB

Remark: In order to receive Qualisys data in Matlab, it is recommended to follow the guide given below. This guide has been made by Einar Skiftestad Ueland at NTNU, 2016, and has been changed for use with the underwater Qualisys system. Among the main deviations from the version created by Ueland are grammatical changes for improved readability, IP address that we will connect to and the object that will be called. It should be noted that the method is limited to Linux-operating systems. This is because the developer of this functionality was made by KumarRobotics in ROS (see kumarrobotics.org or KumarRobotics/qualisys source code ([Accessed 10-March-2016])), while the packages that allow for communication through Matlab already exists in the Matlab robotics toolbox.

B.2.1 About the Manual

This approach may be taken to read Qualisys data into ROS and MATLAB. The method is convenient for Qualisys data into to MATLAB independent on whether the rest of the system uses ROS or not.

The first part of the manual describes how to import Qualisys data into ROS, while the second part explains how to get the data from ROS into MATLAB/Simulink.

Please note the following:

- In the manual, the dollar sign $ indicate a line of text that should be written in the Linux- terminal window.
- In the manual, "gedit" is used as text editor. This can be replaced with the readers favourite text editor.
- The manual is written and tested for ROS-Indigo and MATLAB 2015b. It is based on MATLAB's manual for importing custom messages (MathWorks), which is and adapted and expanded to fit that of the MC lab and the Qualisys system.
- You need MATLAB version 2015a or newer in order to proceed with the MATLAB section of the manual.

B.2.2 Manual

If not already installed on the machine you should start by installing ROS. Follow the instructions on the ROS download page:
http://wiki.ros.org/indigo/Installation/Ubuntu

You should now make a ROS workspace in your home directory:

```
$mkdir −p ~/catkin_ws/src
$cd ~/catkin_ws/src
$catkin_init_workspace
```

Make sure that you are sourcing the setup.bash file in your ROS workspace each time you open your terminal window. This can be done by changing the bash file:

```
$ echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
```

Now import the Qualisys driver from GitHub. (The driver (KumarRobotics/qualisys source code) is avaiable through the Apache License)

```
$ cd ~/catkin_ws/src
$ git clone https://github.com/KumarRobotics/qualisys
$ cd ~/catkin_ws
$ catkin_make
```

Open the qualisys.launch file in a text-editor

```
$ sudo gedit ~/catkin_ws/src/qualisys/launch/qualisys.launch
```

Edit the ip address and port number for the Qualisys system (as of May 2016 the IP is: 192.168.0.20 and the port is 22222). The driver should now be set for interfacing with Qualisys in ROS. To test it, first check that you are able to ping the Qualisys system over the MC lab WiFi.

```
$ ping 192.168.0.10
```

If you successfully pinged the Qualisys system, it should now be possible to listen to the data from the Qualisys system. Note that the Qualisys system need to recognize the IR-markers in the MC lab in order to transmit data. It may be smart to first to check that the computer running Qualisys software in the MC-Lab is able to capture the markers

```
$ roslaunch qualisys qualisys.launch
$ rostopic list
```

The command "rostopic list" prints the active ROS topics. It should now be printed a qualisys topic in terminal. The name will depend on the name set on the Qualisys computer. In this manual the topic is named /qualisys/videoray. You can now listen to the data as it is published to ROS by entering

```
$ rostopic echo /qualisys/videoray
```

Getting Qualisys data to MATLAB

The message sent from the Qualisys system is a custom message that MATLAB does not recognize (most messages in ROS are not custom, and will be recognized by MATLAB). In order to get the Qualisys data into MATLAB, you need to make MATLAB recognize the custom message.

Start by creating a new folder ∼/qualisysDir. Now copy the folder named qualisys, located in ∼/catkin_ws/src and paste it into the folder ∼/qualisysDir. Next, edit the package file so that MATLAB recognizes the messages.

```
Sudo gedit ∼/qualisysDir/qualisys/package.xml
```

Add the following two lines somewhere in the main body of the package.xml file.

```
<build_depend>geometry_msgs</build_depend>
<build_depend>std_msgs</build_depend>
```

Open Matlab, and download the ROS custom message package. Type the following lines into the MATLAB command window, and follow the instructions to download the ROS custom message package.

```
roboticsAddons (in MATLAB 2016)
roboticsSupportPackages (in MATLAB 2015)
```

When the download is finished, type the following into the Matlab console:

```
folderpath= '∼/qualisysDir'
rosgenmsg(folderpath)
```

Follow the instructions generated by MATLAB in order generate the needed message type. Warning: You may need allow writing permission to the file "pathdef.m"

Remember that the Qualisys node ALWAYS needs to be launched before reading signals in MATLAB. Open a new terminal, and enter:

```
roslaunch qualisys qualisys.launch
```

You can now get the data into Simulink by the Subscriber block, or to MATLAB workspace by typing the following commands:

```
Subb = rossubscriber('/qualisys/videoray');
posedata = receive(Subb,10);
```

APPENDIX C ATTACHMENTS

Appendix C contains the attachments to this thesis, stored in a ZIP file. This collection comprises the following contents and folders.

Code files: C/C++

This folder contains all C/C++ code files that manage ROV. In addition, C++ files that are augmented with the mex interface are placed here.

Code files: Matlab

All Matlab scripts and functions are stored within the Matlab code folder.

Code files: Simulink

The Simulink files and Matlab scripts for running the Simulink model are contained within this folder. The model is created in Matlab and Simulink versions R2015b.

Poster

The A2 poster ($2\times$A3) has been included in .pdf-format.

Readme.txt

The readme.txt file contains a simple step-by-step method with a sufficient amount of information for setting up the system on a local computer.