Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Marine Technology

Roar Stavnes

Vision-Aided Inertial Navigation for Underwater Vehicles Using ArUco Markers

Master's thesis in Marine Technology
Supervisor: Asgeir J. Sørensen
Co-supervisor: Henrik Schmidt-Didlaukies and Erlend A. Basso
June 2021

NTNU
Norwegian University of
Science and Technology

Roar Stavnes

Vision-Aided Inertial Navigation for Underwater Vehicles Using ArUco Markers

Master's thesis in Marine Technology
Supervisor: Asgeir J. Sørensen
Co-supervisor: Henrik Schmidt-Didlaukies and Erlend A. Basso
June 2021

Norwegian University of Science and Technology
Faculty of Engineering
Department of Marine Technology

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

MASTER OF TECHNOLOGY THESIS DEFINITION (30 SP)

Name of the candidate:	Roar Stavnes
Field of study:	Marine cybernetics
Thesis title (Norwegian):	**Maskinsynkorrigert treghetsnavigasjon for undervannsfartøy ved bruk av ArUco markører**
Thesis title (English):	**Vision-Aided Inertial Navigation for Underwater Vehicles Using ArUco Markers**

Background

The transition from traditional offshore energy production towards subsea production has resulted in increasing interest in autonomy. By intelligent reasoning based on sensory information, autonomous systems can perform tasks with little to no human interaction. In other words, autonomous robots are an excellent choice for permanently subsea resident robots, being ready 24/7 for planned on-demand operations as well as being available as first response for unplanned operations. Replacing traditional work class ROVs has enormous potential and impact by eliminating tethered connections to surface vessels - lowering the C02 footprint and increases marine operations' efficiency. However, in the transition towards autonomous inspection, maintenance, and repair (IMR) operations, navigation is one of the problems that remain to be solved.

As the modern literature and research on underwater navigation for IMR vehicles are lean, this thesis's primary focus and objective are deriving a system for accurate pose (position and orientation) estimation near subsea intervention and maintenance locations.

Scope of Work

1. Perform a background study to provide information and relevant references on:
    - Navigation systems for underwater vehicles.
    - Methods of computer vision and its application for computer vision-based navigation of robots and vehicles.
    - Review the available and open-source computer vision software libraries.

2. Consider how to estimate the vehicle's pose based on vision sensors and landmarks.

3. Perform a simulation study to determine the estimation accuracy of the computer vision application and evaluate its performance against the existing solutions.

4. Propose a method to close the loop between the vision-based pose estimation and the state-of-the-art inertial navigation system.

5. Present the simulation and experimental setup, including the hardware layout.

6. Perform a simulation study to test and verify the navigation system proposed in 4.

7. Implement the design proposed in 4 on the BlueROV2 and test the implementations experimentally at the Marine Cybernetics Laboratory at the Norwegian University of Science and Technology.

Specifications

The student shall at startup provide a maximum 2-page week plan of work for the entire project period, with main activities and milestones. This should be updated on a monthly basis in agreement with supervisor.

Every weekend throughout the project period, the candidate shall send a status email to the supervisor and co-advisors, providing two brief bulleted lists: 1) work done recent week, and 2) work planned to be done next week.

The scope of work may prove to be larger than initially anticipated. By the approval from the supervisor, described topics may be deleted or reduced in extent without consequences with regard to grading.

The candidate shall present personal contribution to the resolution of problems within the scope of work. Theories and conclusions should be based on mathematical derivations and logic reasoning identifying the steps in the deduction.

The report shall be organized in a logical structure to give a clear exposition of background, problem/research statement, design/method, analysis, and results. The text should be brief and to the point, with a clear language. Rigorous mathematical deductions and illustrating figures are preferred over lengthy textual descriptions. The report shall have font size 11 pts., and it is not expected to be longer than 70 A4-pages, 100 B5-pages, from introduction to conclusion, unless otherwise agreed. It shall be written in English (preferably US) and contain the elements: Title page, abstract, preface (incl. description of help, resources, and internal and external factors that have affected the project process), acknowledgement, project definition, list of symbols and acronyms, table of contents, introduction (project background/motivation, objectives, scope and delimitations, and contributions), technical background and literature review, problem formulation, method, results and analysis, conclusions with recommendations for further work, references, and optional appendices. Figures, tables, and equations shall be numerated. The original contribution of the candidate and material taken from other sources shall be clearly identified. Work from other sources shall be properly acknowledged using quotations and a Harvard citation or Vancouver reference style (e.g. natbib Latex package). The work is expected to be conducted in an honest and ethical manner, without any sort of plagiarism and misconduct, which is taken very seriously by the university and will result in consequences. NTNU can use the results freely in research and teaching by proper referencing, unless otherwise agreed.

The thesis shall be submitted with an electronic copy to the main supervisor and department according to NTNU administrative procedures. The final revised version of this thesis definition shall be included after the title page. Computer code, pictures, videos, data series, etc., shall be included electronically with the report.

Start date: 15 January, 2021 **Due date:** 10 June, 2021

Supervisor: Asgeir J. Sørensen
Co-advisor(s): Henrik Schmidt-Didlaukies and Erlend A. Basso

Trondheim, 09.06.2021

[illegible]

Professor Asgeir J. Sørensen
Supervisor

Preface

The research presented in this Master's thesis is the result of the author's work from January to June of 2021 at the Department of Marine Technology (IMT) at the Norwegian University of Science and Technology (NTNU). The independent work was done with supervision from Professor Asgeir J. Sørensen at IMT NTNU and the co-advisors Henrik Schmidt-Didlaukies at IMT NTNU and Erlend A. Basso at the Department of Engineering Cybernetics (ITK) at NTNU.

Writing a thesis in the middle of the COVID-19 pandemic has been challenging. However, the supervisor's and co-advisors' adaptability in terms of digital guidance made this thesis possible, despite the lockdown restrictions experienced.

The experiment performed in this thesis was carried out in collaboration with the Ph.D. candidates Henrik Schmidt-Didlaukies and Erlend A. Basso through three weeks at the Marine Cybernetics Laboratory (MC-lab) at NTNU in May of 2021. The interdisciplinary collaboration has been decisive for the product of this thesis. Thus, the practical approach has been a treasured experience in order to gain a deeper understanding of the complexities of a physical system.

Acknowledgements

I am grateful to have Professor Asgeir J. Sørensen at IMT NTNU as my supervisor, and I would like to thank him for his excellent guidance throughout the work of this Master's thesis. His multidisciplinary knowledge has been essential in defining the thesis and challenged my way of thinking. Furthermore, the discussions with Sørensen and his effort to make the Master exceptional, such as the invitation to the field test of the Eelume-robot, have been extremely motivating.

I want to thank my two co-advisors, Ph.D. candidate Henrik Schmidt-Didlaukies and Ph.D. candidate Erlend A. Basso for their eagerness and inspiring ideas. I am thankful for the exciting discussions and the fruitful weeks at the MC-lab; it has been an educational journey.

Trondheim, June 10, 2021
Roar Stavnes

Abstract

The transition from traditional offshore energy production towards subsea production has caused a rapidly increasing need for underwater operations in the recent decade. The rapid transition has motivated autonomous operations in order to increase marine operations' efficiency and lower CO_2 footprint. In the transition towards autonomous operations, navigation is one of the main challenges which has to be solved, motivating the work of this thesis.

This thesis presents a method for fusing computer vision with state-of-the-art inertial navigation systems, focusing on improved pose estimation near intervention and maintenance locations. It starts with the presentation of the necessary hardware in Chapter 3. Moreover, Chapter 4 motivates the use ArUco markers for robust detection and pose estimation using a simple setup consisting of a single camera. Before moving on to the observer design, a simple simulation study is carried out, evaluating the accuracy of detection and pose estimation of the ArUco markers. Furthermore, the fusion of computer vision system and navigation system is derived in Chapter 5, based on similar approaches in the literature. At the end, the implementations are tested and validated through a simulation and experimental study, described in Chapter 6.

The principal contributions of the thesis are related to the extension upon an existing open-source simulation software for testing computer vision applications fused with navigation and control. Moreover, an experimentally validated navigation system design for improved localization using fiducial ArUco markers is derived. The proposed navigation system obtained satisfactory results, with a standard deviation in position of 5.61 cm and a mean position error of 10 cm for the experimental case. Compared to the existing geophysical and acoustic transponders and modems techniques, the proposed navigation system has proven high performance and accuracy with potential to replace the existing navigations systems in the vicinity of intervention and maintenance locations.

Sammendrag

Skiftet fra tradisjonell olje- og gassproduksjon mot produksjonenheter på havbunnen har skapt et økt behov for undervannsoperasjoner. Dette har motivert bruken av autonomi for å effektivisere operasjoner og redusere miljøavtrykk. Navigasjon er en av utfordringene som står uløst i overgangen mot autonome undervannsoperasjoner. Dette har motivert arbeidet bak denne hovedoppgaven.

Avhandlingen presenterer en metode for å fusjonere maskinsyn med dagens treghetsnavigasjonssystem, hvor hovedfokuset ligger i forbedring av posisjon og orienteringsestimering i nærheten av operasjonelle og vedlikeholdsområder. Oppgaven starter med å presentere de nødvendige sensorene i Kapittel 3, etterfulgt av motivasjon for bruk av lettgjenkjennelige ArUco markører for estimering av posisjon og orientering i Kapittel 4. Før oppgaven går videre til designet av navigasjonssystemet er det utført en simulasjonsstudie for å avgjøre nøyaktigheten til maskinsynestimatet for videre anvendelse. Kapittel 5 utleder det fusjonerte navigasjonssystemet basert på lignende implementasjoner i litteraturen, hvor orienteringsestimatet er forkastet til fordel for et magnetometer. Til slutt er metoden testet og validert gjennom simulering og eksperiment ved det marinkybernetiske laboratoriet som tilhører Norges teknisk-naturvitenskapelige universitet, som er beskrevet i Kapittel 6.

Hovedbidraget til denne avhandlingen er en utvidelse av en eksisterende åpen kildekode for simulering av metoder i maskinsyn fusjonert med navigasjon og kontroll, med videre potensial for simulering av autonome undervannsoperasjoner. Videre har oppgaven bidratt med et design av et navigasjonssystem for forbedret posisjon- og orienteringsestimat ved bruk av lettgjenkjennelige markører. Designet oppnådde høy ytelse gjennom både simulering og eksperiment, hvor posisjonsstandardavvik kom på 5.61 cm med en gjennomsnittsfeil på 10 cm. Sammenlignet med eksisterende navigeringssystem har denne løsningen en stort potensial for bruk som lokal navigasjon i nærheten av operasjonelle soner som krever høy posisjons- og orienteringsnøyaktighet.

Table of Contents

List of Figures

List of Tables

Notation and Acronyms

AINS	Aided-inertial navigation system
AUV	Autonomous underwater vehicle
DOF	Degrees of freedom
GNSS	Global Navigation Satellite System
IMR	Inspection, maintenance, and repair
IMU	Inertial measurement unit
INS	Inertial navigation system
EKF	Extended Kalman filter
MC-lab	Marine Cybernetics Laboratory
MEKF	Multiplicative extended Kalman filter
MEMS	Microelectromechanical systems
NED	North-East-Down
NTNU	Norwegian University of Science and Technology
PWM	Pulse-Width-Modulation
ROS	Robot operating system
ROV	Remotely operated vehicles
RPi	Raspberry Pi
SLAM	Simultaneous localization and mapping
UUV	Unmanned underwater vehicle
VINS	Vision-aided inertial navigation system
$\mathcal{F}_n$	The North-East-Down frame

$\mathcal{F}_b$	The Body-fixed frame
$\mathcal{F}_s$	The sensor frame
$\mathcal{F}_c$	The camera frame
$\mathcal{F}_{m_i}$	The coordinate frame of marker i
$\mathcal{F}_i$	The coordinate frame i
SO(3)	The Special Orthogonal Group of order three
SE(3)	The Special Euclidean Group of order three
so(3)	The Lie Algebra of SO(3)
$R^i_j \in \mathrm{SO}(3)$	The rotation matrix describing the orientation of $\mathcal{F}_j$ with respect to $\mathcal{F}_i$
$H^i_j \in \mathrm{SE}(3)$	The homogeneous transformation matrix describing the transformation from $\mathcal{F}_j$ to $\mathcal{F}_i$
$[\boldsymbol{p}]_\times$	The skew-symmetric representation of a vector $\boldsymbol{p} \in \mathbb{R}^3$
$\bar{\boldsymbol{p}} = [\boldsymbol{p}^T \ 1]^T$	The homogeneous representation of a vector $\boldsymbol{p} \in \mathbb{R}^n$
I, I_n	The $(n \times n)$ identity matrix
$0_{m\times n}$	The $(m \times n)$ zero matrix
0_n	The $(n \times n)$ zero matrix
$\mathrm{Exp}(\phi \boldsymbol{u})$	The exponential map of unit-quaternions

Chapter 1

Introduction

This chapter will give an introduction to the motivation and background, objectives, research questions, methodology and main contributions.

1.1 Background and Motivation

Underwater operations have been increasingly important in the last decades, ever since the oil adventure in the 1960s. It started as a discipline where divers executed all deepwater operations, even at depths of 300 meters below the surface of the North Sea. The extreme deepwater conditions resulted in 17 deaths and more than 100 serious injuries at the North Sea from the 1960s to the 1980s [39]. These challenges highly motivated the development of underwater robots in order to reduce human interaction in such extreme environments.

The vast majority of underwater vehicles utilized for subsea operations are work-class remotely operated vehicles (ROVs), characterized by their box shape and tether-communication with the operator. The GNSS-denied environment requires a wired connection to have human-in-the-loop control, resulting in larger surface vessel following and assisting the ROV. Consequently, it leads to high operational costs and environmental impact. These disadvantages have challenged traditional thinking and led to research on new ways to design and operate underwater vehicles.

Autonomous underwater operations are one of the fields that have gained a lot of interest. An autonomous system can perform tasks with little to no human interaction by intelligent reasoning based on sensory information. In other words, underwater robots can operate tetherless without human-in-the-loop control. This makes vehicle-manipulators, such as Eelume, an excellent choice for permanently subsea resident robots, being ready 24/7 for planned on-demand intervention operations as well as being available as a first response for unplanned operations [27, 44]. The potential and impact of autonomous underwater operations are enormous. It will, among others, cause significant reductions in operational costs and reduce environmental impact by less offshore service vessel activity. The increasing interest among researchers and industry partners has led to the recent establishment of the NTNU-VISTA Centre for Autonomous Operations Subsea. One of the main aspects of the center is how to navigate AUVs between specified installations to do inspections and, if necessary, perform intervention tasks [45].

1.1.1 Problem Formulation

Subsea inspection, maintenance, and repair (IMR) operations are challenging to solve autonomously. In contrast to other AUV tasks, such as oceanographic surveys, the requirement for accurate navigation is distinct. Surveys require accurate navigation of the entire operation, while IMR vehicles tolerate more inaccuracies during transitions between the subsea sites but require high precision during intervention operations. For example, suppose the IMR vehicle is closing a valve autonomously; it is then essential to know the exact location of the manipulator's arm.

As the modern literature and research on underwater navigation for IMR vehicles are lean, the primary focus of this thesis is deriving a system for accurate pose (position and orientation) estimation near subsea intervention and maintenance locations.

1.2 Existing Theory

AUV navigation has gained considerable interest among researchers since the first development in earnest in the 1970s. Underwater navigation is particularly challenging compared to surface navigation due to the rapid attenuation of the GNSS and radio-frequency signals.

For the interested reader, [35] provides a review of the general techniques in AUV navigation and localization. According to the article, the navigation techniques can be classified based on Figure 1.1.

1.2.1 Inertial Navigation

Inertial navigation is one of the three main categories of AUV navigation and localization. This technique achieves the pose estimate using a kinematic model of rigid-body motion, where measured accelerations and angular velocities propagate the current state. The accelerations and angular rates are measured using an inertial measurement unit (IMU), consisting of accelerometers and gyroscopes. Nevertheless, every single method in this category suffers position error growth that is unbounded, caused by the integration of measurement noise.

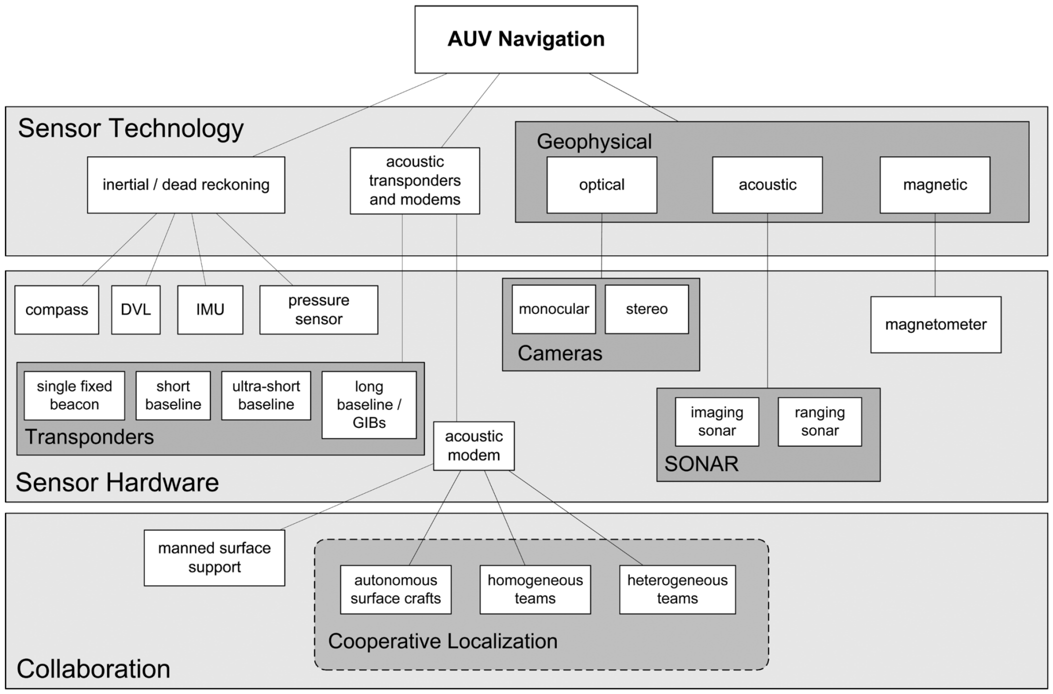


Figure 1.1: Outline of underwater navigation classifications. Courtesy of [35].

Hence, the inertial navigation solution will drift rapidly with time. As a result, the performance of IMUs is extremely variable, from expensive high-precision accelerometers and ring laser gyroscopes to cheap MEMS-based sensors (see Table 1.2). Chapter 5 introduces the rigid-body motion kinematic model and describes inertial navigation in detail.

1.2.2 Acoustic Transponders and Modems

The second technique, acoustic transponders and modems, estimates position using transponders located in the environment and mounted on the vehicle. The acoustic navigation system works in one out of two ways.

- The first principle is similar to the GNSS, where it measures the range, i.e., the time of flight (TOF), from three or more transponders. The measured distances are then used to estimates the position based on triangular relations. This principle is known as either long baseline systems (LBL) or short baseline systems (SBL), depending on the distance between the transponders [24].

- The second principle is ultra-short baseline systems (USBL), consisting of one transceiver mounted on, e.g., a surface vessel. The method estimates the position by measuring the range and the bearing, i.e., the angle at which the acoustic signal approaches the vehicle.

The bearing is calculated based on the difference in phase of the signal arriving at the vehicle's transceiver [35].

Note that the position is estimated relative to the environmental transponders. However, the vehicle's global position can easily be computed if the locations of the external transponders are known. Thus, acoustic positioning and inertial navigation system are often combined to eliminate the unbounded position drift, known as aided-inertial navigation systems (AINS) [21, 35]. The accuracy of acoustic positioning is presented in [21], and the result are reproduced in Table 1.1. Note that the equipment used in producing the results is high-quality sensors developed by Kongsberg Maritime and the Norwegian Defence Research Establishment. Hence, a cheaper system may have significantly higher uncertainties.

Table 1.1: AUV horizontal position uncertainty (1σ) due to ship attitude uncertainty of $0.01°$ (1σ) in roll and pitch, and $0.1°$ (1σ) in heading. The ship attitude is $\phi = 0°$, $\theta = 0°$, $\psi = 0°$. The relative horizontal position between the AUV and the USBL transducer is $x = 50$ m, $y = 50$ m [21].

AUV depth [m]	50	100	500	1000	3000
AUV position uncertainty [m]	0.12	0.13	0.17	0.28	0.75

1.2.3 Geophysical Navigation

The last category determine the vehicle's position by detecting and identifying prior known environmental features. Hence, geophysical navigation relies on onboard sensors only. Thus, in contrast to acoustic transponders and modems, geophysical navigation allows low-cost operations where the AUV can be deployed in unknown waters without further human interaction until pick-up. For underwater navigation, simultaneous localization and mapping (SLAM) are widely used. SLAM is the process of a robot autonomously building a map of its environment through features and, at the same time, localizing itself by detecting the known features. See, e.g., [35] for various SLAM approaches. Extended Kalman filter (EKF) SLAM is one approach where each of the n features is included as a state in the filter. Hence, the EKF-SLAM is computationally expensive due to its $\mathcal{O}(n^2)$ time scaling. Moreover, misinterpretation of landmarks is critical for the covariance correction in the Kalman filter, resulting in occasionally poor performance.

One important remark is that the SLAM estimate will drift with the INS solution until the vehicle detects a prior known feature. Moreover, new feature observations get a position uncertainty equal to the vehicle's uncertainty at the instant of observation. In other words, the navigation system has no better accuracy than its most reliable observation.

Table 1.2: Navigational sensors for underwater applications [35].

Sensor	Description	Performance
Compass	A compass provides a globally bounded heading reference. For cheap MEMS-based compasses, the heading reference is obtained by measuring the magnetic field. Consequently, a magnetic compass does not point to the geographic north and is subject to bias in the presence of magnetic objects. Therefore, a gyroscope measuring the earth's rotation by using three gimbals is an alternative for high precision underwater navigation. This technique is unaffected by metallic objects and points to true north.	Accuracy within 1° to 2° for a modestly priced unit.
IMU	An inertial measurement unit (IMU) consists of gyroscopes and accelerometers, sometimes with magnetometers, fused to estimate the sensor platform's orientation, velocity, and gravitational forces. • Gyroscope: Measures the angular rate. The accuracy comes at the cost of price, where ring laser and fiber optic are expensive high-precision alternatives, while MEMS-based gyroscopes are the cheap option. • Accelerometer: Measures the specific force required to accelerate a proof mass. The working principle includes pendulum, MEMS, and vibrating beam, along with others. Since the IMU measures acceleration and angular rates, the estimated orientation and velocity will drift in time.	The drift of the gyroscope varies with a range from 0.0001 °/hr for a ring laser gyroscope to 60 °/hr or more for a MEMS-based sensor. Moreover, accelerometer bias varies from 0.01 mg for a MEMS-based unit to 0.001 mg for a pendulum [10].

1.3 Research Questions and Methodology

This Master's thesis aims to improve underwater navigation using vision sensors. It involves deriving a method for pose estimation using computer vision and fuse it with state-of-the-art inertial navigation systems. The purpose of research and its focus is summarized in the following questions:

1. What effect does the underwater environment have on vision sensors?

2. How can vision sensors be applied to determine the relative distance and orientation between the sensor and observed objects/landmarks, and how accurate is the method?

3. How can state-of-the-art inertial navigation systems be fused with the information about position and orientation in 2?

The research methodology followed in the present work includes a review of relevant literature and simulations and experiments. The relevant literature has opened new doors and inspired some of the concepts in this thesis, while the simulation and experiments have proved some concepts wrong. Simulations discovered bugs in the initial design and were later used as a validation tool by identifying weaknesses through extensive testing. In this project work, simulation was exceptionally crucial due to the short availability of the experimental facility.

Initially, the feasibility and implementation of computer vision for underwater applications was analyzed by reviewing the relevant literature and open-source libraries. Moreover, a simulation study of the computer vision system determined the accuracy of the position and attitude estimation. Furthermore, the fusion of the vision-based pose estimate and inertial measurements, referred to as a vision-aided inertial navigation system (VINS), was derived based on similar approaches in the literature [15, 40]. Finally, along with the derivation, the system was thoroughly tested and, in the end, verified at the experimental facility—the Marine Cybernetics Laboratory at the Norwegian University of Science and Technology.

1.4 Main Contributions

This thesis contributes to the transition towards subsea resident robots and addresses state estimation close to the subsea operational zones where a high precision position estimate is required. To summarize this work, the contributions are listed below:

- Design of a hardware architecture for UUV motion estimation.
- Extension upon existing open-source simulation software for testing vision-based navigation systems with the necessary sensors.
- Implementation of a vision-aided inertial navigation system capable of obtaining high precision pose estimates using ArUco markers.
- Implementation of a simple joystick controller for human-in-the-loop testing.
- Extensive simulation and experimental study of the system, including the main observations.
- Several suggestions regarding future work and applications of the system.

1.5 Outline

The thesis is organized in a manner where it systematically addresses and motivates the implementation. The text is divided into chapters describing each of the project's main aspects, arranging for a pleasant reading flow and look-up structure. The following chapters are included in the thesis:

Chapter 2 introduces the essential preliminaries behind the thesis.

Chapter 3 presents the essential hardware components; how it is modeled, implemented, and calibrated.

Chapter 4 introduces the computer vision approach and gives the advantages and disadvantages based on the relevant literature. Moreover, it presents a simulation study to determine the accuracy of computer vision system in order to determine its applicability in the navigation system.

Chapter 5 proposes a methodology for implementing a vision-aided inertial navigation system using the Kalman filter.

Chapter 6 describes the simulation and experimental setup, including the test facility and the experimental platform.

Chapter 7 presents the result of the simulation and experimental study described in Chapter 6.

Chapter 8 discusses the main observations and results from Chapter 7.

Chapter 9 concludes the project in the context of the research questions and scope of work. Moreover, it suggests further work and applications of the system.

Appendix A presents the specifications of the BlueROV2.

Chapter 2

Preliminaries

Unlike most surface vehicles, the motion of underwater robots usually considers six degrees of freedom. In other words, at least six independent coordinates are necessary to determine the position and orientation of the vehicle. In the marine community, the most common notation adopts from the Society of Naval Architects and Marine Engineers (SNAME) in [38], see Table 2.1.

The geometry of rigid motions and the three-dimensional space plays a central role in robotics. Throughout this thesis, the Lie groups and their associated Lie algebras represent rigidbody transformations in the three-dimensional space. See, e.g., [16, 19] for more details.

Table 2.1: The notation adopted from the Society of Naval Architects and Marine Engineers [38].

DOF		Forces and moments	Linear and angular velocities	Position and Euler angles relative to the inertial frame
1	Surge	X	u	x
2	Sway	Y	v	y
3	Heave	Z	w	z
4	Roll	K	p	ϕ
5	Pitch	M	q	θ
6	Yaw	N	r	ψ

2.1 Coordinate Frames

A large part of kinematics is concerned with establishing various coordinate frames to represent the position and orientation of rigid objects. In order to describe the position and

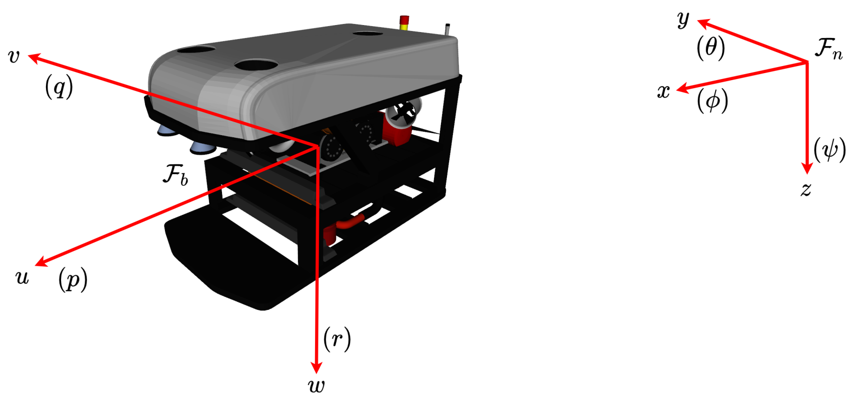


Figure 2.1: Reference frames for underwater vehicles.

orientation of the underwater vehicle, it is convenient to define a Body-fixed frame, $\mathcal{F}_b$, and an earth-fixed inertial frame. Based on the assumption of local navigation in confined areas, the North-East-Down (NED) frame $\mathcal{F}_n$ is used exclusively as the inertial frame in this thesis. Note that the NED frame is, strictly speaking, not inertial as the frame is fixed on the rotating earth. However, for low-speed applications, this is a common and justified simplification [15]. These two coordinate frames are defined as

$$\mathcal{F}_n := (o_n, \boldsymbol{i}_n, \boldsymbol{j}_n, \boldsymbol{k_n})$$
$$\mathcal{F}_b := (o_b, \boldsymbol{i}_b, \boldsymbol{j}_b, \boldsymbol{k_b}),$$

where $o \in \mathbb{R}^3$ defines the origin, and $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k} \in \mathbb{R}^3$ denotes the right-handed perpendicular set of unit basis vectors of the respective frame. The first-mentioned frame is inertial and earth-fixed, while the latter is fixed to the rigid body, as shown in Figure 2.1. Furthermore, suppose the UUV is equipped with sensors, such as an inertial measurement unit (IMU) and a vision sensor. In that case, one coordinate frame is assigned for each sensor to easily transform the measurements to $\mathcal{F}_b$ where the motion is expressed. Three additional types of coordinate frames are encountered in the thesis,

$$\mathcal{F}_s := (o_s, \boldsymbol{i}_s, \boldsymbol{j}_s, \boldsymbol{k}_s)$$
$$\mathcal{F}_c := (o_c, \boldsymbol{i}_c, \boldsymbol{j}_c, \boldsymbol{k}_c)$$
$$\mathcal{F}_m := (o_m, \boldsymbol{i}_m, \boldsymbol{j}_m, \boldsymbol{k}_m),$$

where $\mathcal{F}_s$ represents the sensor frame with axis aligned with the IMU-axis, $\mathcal{F}_c$ represents the camera frame, and $\mathcal{F}_m$ represents the marker frame.

2.2 Generalized Coordinates

For a marine craft, two common vectors that being used in defining the underwater vehicle state vector are $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ [14]. The vector $\boldsymbol{\eta}$ is called the generalized position, defined as

$$\boldsymbol{\eta} = [x\ y\ z\ \phi\ \theta\ \psi]^T \in \mathbb{R}^6. \tag{2.1}$$

The generalized position is commonly divided into two parts, $\boldsymbol{p} = [x\ y\ z]^T$ and $\boldsymbol{\Theta} = [\phi\ \theta\ \psi]^T$, where $\boldsymbol{p}$ denote the vehicles position relative to the earth fixed frame and $\boldsymbol{\Theta}$ denote the vehicles orientation relative to the earth-fixed frame. Moreover, the generalized velocities are the time derivatives of the generalized position of the system:

$$\dot{\boldsymbol{\eta}} = [\dot{x}\ \dot{y}\ \dot{z}\ \dot{\phi}\ \dot{\theta}\ \dot{\psi}]^T = [\boldsymbol{v}^T\ \boldsymbol{\omega}^T]^T. \tag{2.2}$$

These quantities are all expressed in the inertial earth-fixed frame. However, it is advantageous to express the velocities in $\mathcal{F}_b$ when deriving the equations of motion. The velocities in $\mathcal{F}_b$ is denoted as

$$\boldsymbol{\nu} = [u\ v\ w\ p\ q\ r]^T \in \mathbb{R}^6. \tag{2.3}$$

Similarly to the generalized position, the velocity vector is divided into two parts, $\boldsymbol{\nu}_1 = [u\ v\ r]^T$ and $\boldsymbol{\nu}_2 = [p\ q\ r]^T$, where $\boldsymbol{\nu}_1$ is the body-fixed velocity vector and $\boldsymbol{\nu}_2$ is the body-fixed angular velocity vector.

2.3 Lie Groups

The kinematics of a rigid body can be derived globally in terms of Lie group and Lie algebra structures. More specifically, an element of the Lie group corresponds to a rigid body configuration, while elements of the Lie algebra express velocity. In other words, the state space may be written in terms of the Lie group and algebra [16].

A valuable property of groups in general is that two elements of a group can be combined to produce an element in the same group. This operation is defined by the group operator, commonly denoted as multiplication. However, it does not necessarily represent multiplication in the ordinary sense, which is the case for the quaternion product introduced in the following subsection. Furthermore, the inverse of an element of the group is also an element of the same group. The general matrix multiplication defines the group operator for matrix groups, while the inverse matrix represents the opposite operation. All these properties make the concept of groups a potent tool for describing rigid body motions [16].

It is desirable to use a smoothly differentiable group for describing rigid motions, i.e., groups in which there are no singularities that can take motions to infinity. This leads to the definition of the Lie groups:

Definition 2.3.1 (Lie Group) *A Lie Group is a group G which is also a smooth manifold for which the group operator and the inverse are smooth mappings.*

A Lie group of immediate interest is the *Special Orthogonal Group* of order three, denoted SO(3). The group consists of the rigid rotations about a fixed point in $\mathbb{R}^3$

$$\mathrm{SO}(3) := \left\{ R \in \mathbb{R}^{3\times 3} \mid \det(R) = 1,\ R^{-1} = R^T \right\}.$$

The rotation matrix $R \in \mathrm{SO}(3)$ describes the orientation of one frame expressed in another, where the notation R^i_j denote the orientation of $\mathcal{F}_j := (o_j, \boldsymbol{i}_j, \boldsymbol{j}_j, \boldsymbol{k}_j)$ expressed in $\mathcal{F}_i := (o_i, \boldsymbol{i}_i, \boldsymbol{j}_i, \boldsymbol{k}_i)$. The opposite or inverse transformation, $(R^i_j)^{-1}$, are unique and belongs to the same group. Geometrically, it can be interpreted as the opposite rotation, i.e., R^j_i. The rotation matrix is found by the projection of the unit vectors of $\mathcal{F}_j$ onto $\mathcal{F}_i$

$$R^i_j = \begin{bmatrix} \boldsymbol{i}_j \cdot \boldsymbol{i}_i & \boldsymbol{j}_j \cdot \boldsymbol{i}_i & \boldsymbol{k}_j \cdot \boldsymbol{i}_i \\ \boldsymbol{i}_j \cdot \boldsymbol{j}_i & \boldsymbol{j}_j \cdot \boldsymbol{j}_i & \boldsymbol{k}_j \cdot \boldsymbol{j}_i \\ \boldsymbol{i}_j \cdot \boldsymbol{k}_i & \boldsymbol{j}_j \cdot \boldsymbol{k}_i & \boldsymbol{k}_j \cdot \boldsymbol{k}_i \end{bmatrix}, \tag{2.4}$$

where $(\cdot)$ denote the dot product. It is convenient to derive the the principal rotation matrices (one-axis rotations), $R_{\boldsymbol{\lambda},\beta}$, where $\boldsymbol{\lambda}$ and β denote the axis and angle of rotation, respectively. From (2.4) it is clear that

$$R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \mathrm{c}_\phi & -\mathrm{s}_\phi \\ 0 & \mathrm{s}_\phi & \mathrm{c}_\phi \end{bmatrix}, R_{y,\theta} = \begin{bmatrix} \mathrm{c}_\theta & 0 & \mathrm{s}_\theta \\ 0 & 1 & 0 \\ -\mathrm{s}_\theta & 0 & \mathrm{c}_\theta \end{bmatrix}, R_{z,\psi} = \begin{bmatrix} \mathrm{c}_\psi & -\mathrm{s}_\psi & 0 \\ \mathrm{s}_\psi & \mathrm{c}_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.5}$$

where c_α and s_α is the short notation for $\cos(\alpha)$ and $\sin(\alpha)$, respectively. The Lie algebra of the Special Orthogonal group, SO(3), may be represented as

$$\mathrm{so}(3) := \left\{ \Omega \in \mathbb{R}^{3\times 3} \mid \Omega = -\Omega^T \right\}.$$

The group is identified with $\mathbb{R}^3$ through the skew-symmetric representation, which is useful for calculation of relative velocity transformation between coordinate frames [42]. The skew-symmetric representation of a vector $\boldsymbol{\omega} = [\omega_1\ \omega_2\ \omega_3]^T \in \mathbb{R}^3$ is defined as

$$[\boldsymbol{\omega}]_\times = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, \tag{2.6}$$

where $[\cdot]_\times : \mathbb{R}^3 \to \mathrm{so}(3)$. The skew-symmetric representation possesses some useful properties,

$$[\boldsymbol{a}]_\times \boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b} \tag{2.7}$$

$$[\boldsymbol{a}]_\times \boldsymbol{b} = -[\boldsymbol{b}]_\times \boldsymbol{a}, \tag{2.8}$$

where $\boldsymbol{a}$ and $\boldsymbol{b}$ are any vectors belonging to $\mathbb{R}^3$. Now consider two rigidly attached frames, where the time-varying rotation matrix R_j^i represents the rotation of $\mathcal{F}_j$ relative to $\mathcal{F}_i$. Assuming R_j^i is continuously differentiable, one may write the time derivative as

$$[\boldsymbol{\omega}]_\times = \dot{R}_j^i (R_j^i)^T, \tag{2.9}$$

where the vector $\boldsymbol{\omega} \in \mathbb{R}^3$ is the time-varying angular velocity of $\mathcal{F}_j$ relative to $\mathcal{F}_i$.

Consider now a rigid motion, which is a pure translation together with a pure rotation. Let R_j^i be the rotation matrix that specifies the orientation of $\mathcal{F}_j$ with respect to $\mathcal{F}_i$, and $\boldsymbol{d}_{ij}^i$ be the vector from the origin of $\mathcal{F}_i$ to the origin of $\mathcal{F}_j$ expressed in $\mathcal{F}_j$. Suppose the point p is rigidly attached to $\mathcal{F}_j$, with $\boldsymbol{p}^j$ denoting the vector from $\mathcal{F}_j$ to p. One may now express the coordinates of p with respect to $\mathcal{F}_i$ using

$$\boldsymbol{p}^i = R_j^i \boldsymbol{p}^j + \boldsymbol{d}_{ij}^i, \tag{2.10}$$

where $\boldsymbol{p}^i$ denote the vector from $\mathcal{F}_i$ to p expressed in $\mathcal{F}_i$. Hence, position and orientation may be expressed through the homogeneous representation

$$\bar{\boldsymbol{p}}^i = \begin{bmatrix} R_j^i & \boldsymbol{d}_{ij}^i \\ \boldsymbol{0}_{1\times 3} & 1 \end{bmatrix} \bar{\boldsymbol{p}}^j, \tag{2.11}$$

where $\bar{(\cdot)}$ denote the homogeneous coordinate representation, i.e., $\bar{\boldsymbol{p}} := [\boldsymbol{p}^T \;\; 1]^T$. The matrix encountered in (2.11) is referred to as the homogeneous transformation matrix

$$H_j^i = \begin{bmatrix} R_j^i & \boldsymbol{d}_{ij}^i \\ \boldsymbol{0}_{1\times 3} & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}, \tag{2.12}$$

with inverse

$$H_i^j = \left(H_j^i\right)^{-1} = \begin{bmatrix} R_i^j & -R_i^j \boldsymbol{d}_{ij}^i \\ \boldsymbol{0}_{1\times 3} & 1 \end{bmatrix}. \tag{2.13}$$

Homogeneous transformations inherits its name from the homogeneous coordinates, which form the transformation. The introduction of translation requires one additional dimension, meaning one can no longer rely on Cartesian coordinates. Instead, homogeneous coordinates are used to represent a point in space. The homogeneous coordinates of a point (x, y, z) in $\mathbb{R}^3$ are defined as $(\lambda x, \lambda y, \lambda z, \lambda)$ for a nonzero $\lambda \in \mathbb{R}$. By this definition, the multiplication of any nonzero factor λ represents the same point in space. Hence the name homogeneous coordinates.

Homogeneous transformation matrices are a way of representing the *Special Euclidean Group* of dimension three,

$$\mathrm{SE}(3) := \left\{ \begin{bmatrix} R & \boldsymbol{d} \\ \boldsymbol{0}_{1\times 3} & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4} \;\middle|\; R \in \mathrm{SO}(3), \boldsymbol{d} \in \mathbb{R}^3 \right\},$$

which is the group of rigid-body transformations on $\mathbb{R}^3$.

Furthermore, a useful property of rigid bodies in kinematic chains is the composition of multiple homogeneous transformations

$$H_j^i = \begin{cases} H_j^{j+1} H_{j+1}^{j+2} \cdots H_{i-1}^i & \text{if } j < i \\ I_4 & \text{if } j = i \\ \left(H_j^i\right)^{-1} & \text{if } j > i, \end{cases} \tag{2.14}$$

where I_4 denote the identity matrix of dimension 4 [42].

2.4 Attitude Representation

The attitude of the UUV relative to the inertial frame is given by the rotation matrix R_b^n. Because the rotation matrix is less intuitive and most observer designs do not combine vector and matrix for state representation, it is convenient to parameterize the rotation matrix. There are various attitude parametrizations in the literature, e.g., unit-quaternions and Euler angle representation [15, 40]. In this thesis, the unit-quaternion representation is used exclusively in the implementations due to its computational advantage and non-singularities. However, the Euler angle representation is used for debugging and presenting the results as it is intuitive and easy to comprehend.

2.4.1 Euler Angles

There are various Euler angle representations, e.g., the *zyx* convention, defined by the following sequence of rotations

$$R_b^n(\mathbf{\Theta}) := R_{z,\psi} R_{y,\theta} R_{x,\phi},$$

where $\mathbf{\Theta}$ denote the vector of body Euler-angle coordinate. Expanding the Euler angle representation with (2.5) yields

$$R_b^n(\boldsymbol{\eta}_2) = \begin{bmatrix} \mathrm{c}_\psi \mathrm{c}_\theta & -\mathrm{s}_\psi \mathrm{c}_\phi + \mathrm{c}_\psi \mathrm{s}_\theta \mathrm{s}_\phi & \mathrm{s}_\psi \mathrm{s}_\phi + \mathrm{c}_\psi \mathrm{c}_\phi \mathrm{s}_\theta \\ \mathrm{s}_\psi \mathrm{c}_\theta & \mathrm{c}_\psi \mathrm{c}_\phi + \mathrm{s}_\phi \mathrm{s}_\theta \mathrm{s}_\psi & -\mathrm{c}_\psi \mathrm{s}_\phi + \mathrm{s}_\theta \mathrm{s}_\psi \mathrm{c}_\phi \\ -\mathrm{s}_\theta & \mathrm{c}_\theta \mathrm{s}_\phi & \mathrm{c}_\theta \mathrm{c}_\phi \end{bmatrix}. \tag{2.15}$$

The vehicle's motion path relative to the inertial earth-fixed frame, $\mathcal{F}_n$, is given by the 6-DOF kinematic equation as follow [14]:

$$\dot{\boldsymbol{\eta}} = J_b(\mathbf{\Theta})\boldsymbol{\nu}, \tag{2.16}$$

where $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ are given in (2.1) and (2.3), and $J_b(\boldsymbol{\Theta}) : \mathbb{R}^3 \rightarrow \mathbb{R}^{6\times6}$ is the kinematics transformation matrix or Jacobian matrix, given as

$$J_b(\boldsymbol{\Theta}) = \begin{bmatrix} R_b^n(\boldsymbol{\Theta}) & 0_{3\times3} \\ 0_{3\times3} & T_b^n(\boldsymbol{\Theta}) \end{bmatrix}$$
$$= \begin{bmatrix} \mathrm{c}_\psi \mathrm{c}_\theta & -\mathrm{s}_\psi \mathrm{c}_\phi + \mathrm{c}_\psi \mathrm{s}_\theta \phi & \mathrm{s}_\psi \mathrm{s}_\phi + \mathrm{c}_\psi \mathrm{c}_\phi \mathrm{s}_\theta & 0 & 0 & 0 \\ \mathrm{s}_\psi \mathrm{c}_\theta & \mathrm{c}_\psi \mathrm{c}_\phi + \mathrm{s}_\phi \mathrm{s}_\theta \mathrm{s}_\psi & -\mathrm{c}_\psi \mathrm{s}_\phi + \mathrm{s}_\theta \mathrm{s}_\psi \mathrm{c}_\phi & 0 & 0 & 0 \\ -\mathrm{s}_\theta & \mathrm{c}_\theta \mathrm{s}_\phi & \mathrm{c}_\theta \mathrm{c}_\phi & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \frac{\mathrm{s}_\phi \mathrm{s}_\theta}{\mathrm{c}_\theta} & \frac{\mathrm{c}_\phi \mathrm{s}_\theta}{\mathrm{c}_\theta} \\ 0 & 0 & 0 & 0 & \mathrm{c}_\phi & -\mathrm{s}_\phi \\ 0 & 0 & 0 & 0 & \frac{\mathrm{s}_\phi}{\mathrm{c}_\theta} & \frac{\mathrm{c}_\phi}{\mathrm{c}_\theta} \end{bmatrix}, \tag{2.17}$$

where c_α, s_α and t_α are short notations for $\cos(\alpha)$, $\sin(\alpha)$ and $\tan(\alpha)$, respectively. Note that the Euler angles introduce a singularity when mapping from the body velocities to the generalized velocities expressed in the earth-fixed frame. From (2.17), it is clear that the matrix is singular for $\theta = \pm\frac{\pi}{2}$.

2.4.2 Unit Quaternions

An alternative representation that eliminate the Euler angle singularity is the non-minimal unit-quaternion representation [1]. A quaternion is defined as a complex number with one real part q_w and three imaginary parts given by the vector $\boldsymbol{q_v} = [q_x \; q_y \; q_z]^T$. The set $\mathcal{Q}$ of unit-quaternions is defined as

$$\mathcal{Q} := \left\{ \boldsymbol{q} \;\middle|\; \boldsymbol{q}^T \boldsymbol{q} = 1,\; \boldsymbol{q} = [q_w \; \boldsymbol{q_v}^T]^T,\; q_w \in \mathbb{R} \text{ and } \boldsymbol{q_v} \in \mathbb{R}^3 \right\}.$$

Due to its non-unique representation, linear addition and subtraction are not preferable as they may lead to possible issues in preserving the motion space of unit-quaternions. Alternatively, the quaternion product, known as the *Hamiltonian product*, is often used instead. The product is defined as

$$\boldsymbol{p} \otimes \boldsymbol{q} = \begin{bmatrix} p_w q_w - \boldsymbol{p}_v^T \boldsymbol{q}_v \\ p_w \boldsymbol{q}_v + q_w \boldsymbol{p}_v + [\boldsymbol{p}_v]_\times \boldsymbol{q}_v \end{bmatrix}, \tag{2.18}$$

where $\boldsymbol{p}, \boldsymbol{q} \in \mathcal{Q}$. Note that the presence of the cross-product revals that the hamiltonian product is not commutative in the general case. However, the quaternions are associative and distributive over the sum [40],

$$\boldsymbol{p} \otimes \boldsymbol{q} \neq \boldsymbol{q} \otimes \boldsymbol{p} \tag{2.19}$$

$$(\boldsymbol{p} \otimes \boldsymbol{q}) \otimes \boldsymbol{r} = \boldsymbol{p} \otimes (\boldsymbol{q} \otimes \boldsymbol{r}) \tag{2.20}$$

$$\boldsymbol{p} \otimes (\boldsymbol{q} + \boldsymbol{r}) = \boldsymbol{p} \otimes \boldsymbol{q} + \boldsymbol{p} \otimes \boldsymbol{r} \tag{2.21}$$

Similarly to the Euler angle representation, the kinematic relationship between the linear velocities expressed in the body-fixed frame and the velocities expressed in the inertial frame is given through the transformation

$$\dot{\boldsymbol{\eta}} = J_q(\boldsymbol{q})\boldsymbol{\nu} \tag{2.22}$$

where $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ are given in (2.1) and (2.3), and $J_q(\boldsymbol{q}) : \mathbb{R}^4 \to \mathbb{R}^{6\times 7}$ denotes the 6 DOF transformation matrix for the unit-quaternion and $\boldsymbol{q} = [q_w \ \boldsymbol{q}_w^T]^T$ is the attitude representation of $\mathcal{F}_b$ expressed in $\mathcal{F}_n$. Note that the additional differential equation in needed because of the unity constraint [15]. The differential equations are expressed as

$$J_q(\boldsymbol{q}) = \begin{bmatrix} R(\boldsymbol{q}) & 0_{3\times 3} \\ 0_{4\times 3} & T(\boldsymbol{q}) \end{bmatrix}, \tag{2.23}$$

where

$$R(\boldsymbol{q}) = I_3 + 2q_w[\boldsymbol{q_v}]_\times + 2[\boldsymbol{q_v}]_\times^2 \tag{2.24}$$

$$T(\boldsymbol{q}) = \begin{bmatrix} -\boldsymbol{q_v}^T \\ q_w I_3 + [\boldsymbol{q_v}]_\times \end{bmatrix}. \tag{2.25}$$

The rotational matrix $R \in \mathrm{SO}(3)$ and its corresponding quaternion $\boldsymbol{q}$ possess some useful properties [40],

$$R([1\ 0\ 0\ 0]^T) = I_3 \tag{2.26}$$

$$R(-\boldsymbol{q}) = R(\boldsymbol{q}) \tag{2.27}$$

$$R(\boldsymbol{q}^{-1}) = R(\boldsymbol{q})^T \tag{2.28}$$

$$R(\boldsymbol{q}_1 \otimes \boldsymbol{q}_2) = R(\boldsymbol{q}_1)R(\boldsymbol{q}_2) \tag{2.29}$$

According to Euler's theorem on finite rotations, one rotation about a certain axis is sufficient to describe any arbitrary sequence of rotations is space. Let $\boldsymbol{u} = [u_1\ u_2\ u_3]^T \in \mathbb{R}^3$ define the unit-vector that represents the axis of rotation, and $\phi \in \mathbb{R}$ denote the angle of rotation, the unit-quaternion may now be represented by an exponential map [40];

$$\boldsymbol{q} := \mathrm{Exp}(\phi\boldsymbol{u}) = e^{\phi\boldsymbol{u}/2} = \cos(\frac{\phi}{2}) + \boldsymbol{u}\sin(\frac{\phi}{2}) = \begin{bmatrix} \cos(\frac{\phi}{2}) \\ \boldsymbol{u}\sin(\frac{\phi}{2}) \end{bmatrix}. \tag{2.30}$$

This exponential map is useful in many situations and will be used in the derivation of the unit-quaternion error-state Kalman filter.

Chapter 3

Hardware

This chapter presents the hardware utilized to test the implementations. Each component is presented in a systematic way, beginning with the mathematical modeling, experimental implementation, and, at the end, calibration.

3.1 Inertial Measurement Unit

The inertial measurement unit (IMU) is a key component in most navigation systems [15]. It consists of a three-axis accelerometer and a three-axis gyroscope, sometimes with additional measurements such as a three-axis magnetometer. The working principle of accelerometers is a mass-spring-damper system mounted at each of the three axes of the senor frame $\mathcal{F}_s$, i.e., $(\boldsymbol{i}_s, \boldsymbol{j}_s, \boldsymbol{k}_s)$ which spans the three-dimensional Euclidean space ($\mathbb{R}^3$). The accelerometer measures the force required to accelerate a proof mass. Note that force is directly related to acceleration through Newton's second law. One important observation is the case where the accelerometer is attached to a non-accelerating body. Here, the gravitation will induce a force on the proof mass, which results in a measured acceleration. Therefore, gravity has to be estimated and subtracted from the measurement in order to get the correct acceleration relative to the inertial frame.

3.1.1 Sensor Model

For any sensor application in motion control systems, it is crucial to have an accurate mathematical description of the measurement. Like most models encountered in control theory, the measurement model of the IMU is described using a linear model [13, 15],

$$\boldsymbol{\omega}^s_{imu}(t) = \boldsymbol{\omega}^s_t(t) + \boldsymbol{b}^s_{gyro}(t) + \boldsymbol{w}^s_{gyro}(t) \tag{3.1a}$$

$$\boldsymbol{a}^s_{imu}(t) = R^s_n(t)\left(\boldsymbol{a}^n_t(t) - \boldsymbol{g}^n\right) + \boldsymbol{b}^s_{acc}(t) + \boldsymbol{w}^s_{acc}(t) \tag{3.1b}$$

$$\boldsymbol{m}^s_{mag}(t) = R^s_n(t)\boldsymbol{m}^n_t(t) + \boldsymbol{b}^s_{mag} + \boldsymbol{w}^s_{mag}(t), \tag{3.1c}$$

where $\boldsymbol{\omega}^s_{imu} \in \mathbb{R}^3$ represents the measured angular body-velocities, $\boldsymbol{\omega}^s_t \in \mathbb{R}^3$ is the true angular velocities, $\boldsymbol{a}^s_{imu} \in \mathbb{R}^3$ is the measured body-acceleration, $\boldsymbol{a}^n_t \in \mathbb{R}^3$ is the true body-acceleration, $\boldsymbol{m}^s_{mag} \in \mathbb{R}^3$ is the measured magnetic field strength, $\boldsymbol{m}^s_t$ is the true magnetic field strength, $\boldsymbol{g}^n \in \mathbb{R}^3$ is the gravitational vector, and $R^s_n \in \mathrm{SO}(3)$ is the rotation of $\mathcal{F}_s$ with respect to $\mathcal{F}_n$. Moreover, superscript s and n denote quantities expressed in the sensor frame $\mathcal{F}_s$ and North-East-Down (NED) frame $\mathcal{F}_n$, respectively. It should be noted that the assumption of the linear sensor model is an oversimplification; thus, it is necessary to include:

- $\boldsymbol{w}^s_{gyro}, \boldsymbol{w}^s_{acc}, \boldsymbol{w}^s_{mag} \in \mathbb{R}^3$ - measurement noise modelled as Gaussian white noise.
- $\boldsymbol{b}^s_{gyro}, \boldsymbol{b}^s_{acc}, \boldsymbol{b}^s_{mag} \in \mathbb{R}^3$ - slowly time-varying bias (drift) modelled as a *Brownian motion*,

where Brownian motion is produced by integration of white noise, i.e.,

$$\dot{\boldsymbol{b}}^s_{gyro} = \boldsymbol{w}^s_{b,gyro} \tag{3.2}$$

$$\dot{\boldsymbol{b}}^s_{acc} = \boldsymbol{w}^s_{b,acc} \tag{3.3}$$

$$\dot{\boldsymbol{b}}^s_{mag} = \boldsymbol{w}^s_{b,mag}. \tag{3.4}$$

The Gaussian white noise accounts for the disturbances in the sensor measurements, which are uncorrelated to the measured state. On the other hand, the bias removes any potential constant offset from the modeling.

3.1.2 Implementation

The IMU used for experiments, BNO055, is delivered by Bosch and consists of three MEMS-based accelerometers, three magnetometers, three gyros, and one internal temperature sensor. The high-speed ARM Cortex-M0-based processor abstracts the sensor fusion and ensures a high refresh rate and performance. The output from the IMU is absolute orientation in quaternion or Euler representation, angular velocity vector, acceleration vector, magnetic field strength, linear acceleration vector, gravity vector, and temperature in degrees Celcius.

Another feature of the BNO055 is the possibility of rejecting the ARM-based sensor fusion for better performance, with the cost of losing the built-in calibration software. However, as dead-reckoning was not the focus of this thesis, the high precision alternative was rejected in favor of the sensor fusion and simple calibration

An open-source driver[1] was used for communication between the ROS network and the IMU, ensuring an update rate of 100Hz.

Table 3.1: IMU specifications.

Parameter	Value	Unit
Sensor	BNO055	
Measurement frequency accelerometer	100	Hz
Measurement frequency gyroscope	100	Hz
Measurement frequency magnetormeter	20	Hz
Bandwidth accelerometer f_{-3dB}	63	Hz
Bandwidth gyroscope f_{-3dB}	47	Hz
Alignment error	< 0.5	deg
Noise density accelerometer N_{acc}	$1.86 \cdot 10^{-3}$	$\frac{\text{m/s}^2}{\sqrt{\text{Hz}}}$
Noise density gyroscope N_{gyro}	$2.55 \cdot 10^{-4}$	$\frac{\text{rad/s}}{\sqrt{\text{Hz}}}$
Noise magnetometer σ_{mag}	1.0	μT

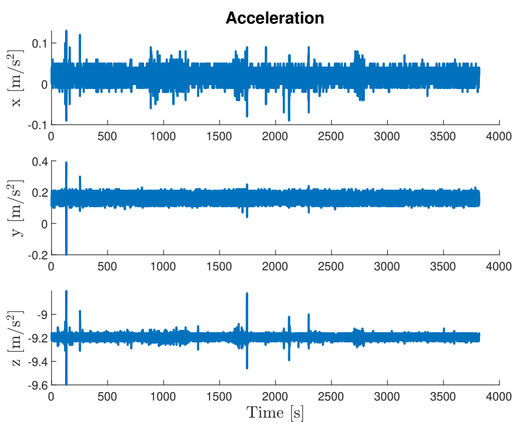
3.1.3 Calibration

The IMU BNO055 has software that runs the calibration and sensor fusion of the accelerometer, gyroscope, and magnetometer. At startup, the gyroscope calibrates by placing the vehicle in a single stable position for a few seconds. Moreover, the accelerometer calibrates by placing the vehicle in six different stable positions. The sensor must be lying at least once perpendicular to the x, y, and z axis, and it has to be slow movement between at least two stable positions. Finally, the magnetometer calibration involves inducing random motions, e.g., drawing the number "8" on air. During the calibration, one can read a status message which rates the calibration from 1 to 3. However, it was observed that, even with a calibration status of 3, the IMU happened to have a poor performance.

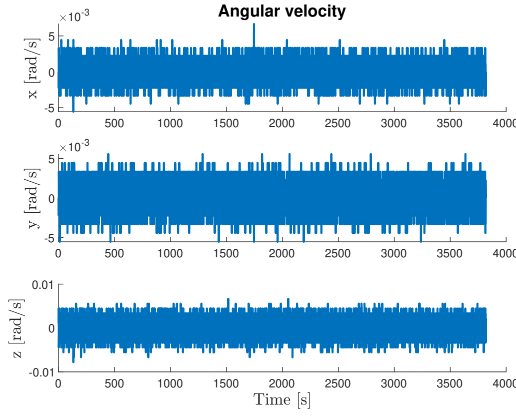
For low price MEMS-based sensors, the magnetometer is usually the faultiest sensor. Inaccurate magnetic field measurements are primarily due to magnetic influences from hard-iron and soft-iron, in addition to current-induced magnetic fields. These disturbances may lead to poor heading estimates for cheap MEMS-based magnetometers.

Further, determining the noise characteristics is convenient for the initial Kalman filter

[1] `http://wiki.ros.org/ros_imu_bno055`



(a) Raw acceleration measurement.



(b) Raw angular velocity measurement.

Figure 3.1: Raw IMU measurements.

tuning. There are various ways to estimate and verify the noise variances, e.g., the Allan variance experiment [36]. However, this study requires several hours of acceleration and angular rate data, in order to compute the root means square (RMS) random drift error as a function averaging time. Accurate inertial measurements were not crucial for the performance of the implementations in this thesis, as dead reckoning was not one of the objectives. Thus, a simple variance study was performed in order to determine some of the measurement noise variances. In order to calculate the measurement noise variances, a ROS bag recorded the output data from the accelerometer and gyroscope for a time interval of one hour (see Figure 3.1). The standard deviation of the output measurement data is given in Table 3.2. The datasheet's

Table 3.2: Standard deviation of the IMU measurements.

	x	y	z	Unit
σ_{acc}	$1.10 \cdot 10^{-2}$	$1.72 \cdot 10^{-2}$	$1.36 \cdot 10^{-2}$	m/s^2
σ_{gyro}	$6.64 \cdot 10^{-4}$	$8.27 \cdot 10^{-4}$	$1.00 \cdot 10^{-3}$	rad/s

measure of noise is *noise density*, defined as the power of noise per unit of bandwidth. The BNO055 has a bandwidth of 47 Hz for the gyroscope and 63 Hz for the accelerometer, interpreted as the maximum frequency to which the sensor responds. The noise density is defined as

$$N = \frac{\sigma}{\sqrt{f_{-3dB}}}, \tag{3.5}$$

where σ and f_{-3dB} are the signal standard deviation and frequency bandwidth, respectively. The comparison of the datasheet and measurement noise densities is shown in Table 3.3. It appears that the accelerometer underperforms, while the gyroscope has better performance than the values given in the datasheet. This is presumably caused by the calibration of the accelerometer, emphasizing its importance for achieving good results.

Table 3.3: Noise density of the IMU measurements.

	Datasheet	Measurement			Unit
		x	y	z	
N_{acc}	$1.86 \cdot 10^{-3}$	$1.39 \cdot 10^{-3}$	$2.16 \cdot 10^{-3}$	$1.71 \cdot 10^{-3}$	$\frac{\text{m/s}^2}{\sqrt{\text{Hz}}}$
N_{gyro}	$2.55 \cdot 10^{-4}$	$9.69 \cdot 10^{-5}$	$1.21 \cdot 10^{-4}$	$1.46 \cdot 10^{-4}$	$\frac{\text{rad/s}}{\sqrt{\text{Hz}}}$

Another relevant observation is the deviation between the measured acceleration and the gravitational acceleration, as seen from Figure 3.1. This emphasizes the importance of including biases, ensuring that $R_s^n(\boldsymbol{a}_{imu}^s - \boldsymbol{b}_{acc}^s) - \boldsymbol{g}^n \approx 0$ for the stationary case.

The sensor's datasheet reports an alignment error of 0.5° of the IMU-axis for a well-calibrated device. Moreover, the sensor might have some installation miss-alignments resulting in the axes of $\mathcal{F}_s$ not coinciding with $\mathcal{F}_b$. This source of error is reduced by computing the exact rotation matrix from the sensor frame to the body frame. By placing the UUV on a horizontal ground with zero heading-angle, one can use the measured acceleration vector to obtain the rotation matrix by the TRIAD algorithm (5.10). Because $\mathcal{F}_n$ and $\mathcal{F}_b$ share the same axis when the roll, pitch and, yaw angle are equal to zero, it is clear that $R_s^n = R_s^b$. Hence, the gravitational vector expressed in $\mathcal{F}_n$ can be normalized and used as reference unit-vector in order to compute the rotation matrix.

3.2 Camera

A camera is a vision sensor used for image capturing, that is, essentially, the process of projecting 3D data into a 2D plane. Because of the projection, one dimension of information is lost in the capturing. However, detecting objects with known size may regain this information - given the relative pose between the camera and the object is known.

3.2.1 Sensor Model

In order to utilize the camera for computer vision applications, it is necessary to have a mathematical model of the process of capturing images. A simple but useful model of how this happens is the pinhole camera model, see, e.g., [29]. Imagine an imaginary wall at the optical center, o, which is orthogonal to the *optical axis*. Let the wall have a tiny aperture in the center, forcing all rays to go through the optical center. Consequently, only one single ray

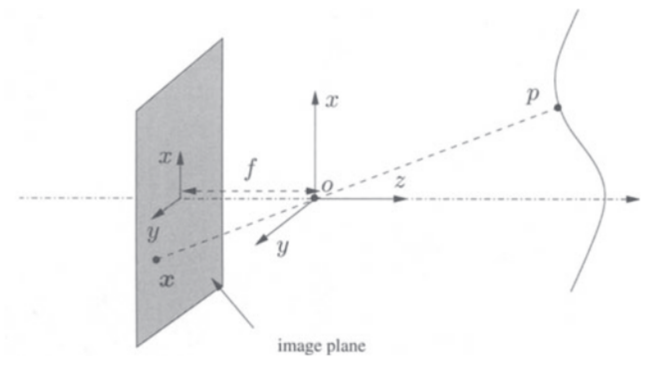


Figure 3.2: The pinhole camera model. Courtesy of [29].

passes from every single point p in space. In the pinhole model, this point is projected onto the *image plane* at $\boldsymbol{x} = [x\ y]^T$, referred to as the *image point.* If the point p has coordinates $\boldsymbol{X} = [X\ Y\ Z]^T$ relative to the camera frame centered at the optical center o, it is immediate to see that the coordinates of p and its image $\boldsymbol{x}$ are related by the so-called ideal *perspective projection* (see Figure 3.2)

$$x = -f\frac{X}{Z}, \quad y = -f\frac{Y}{Z} \tag{3.6}$$

where f is the *focal length.* The projection is often described by the map π:

$$\pi : \mathbb{R}^3 \to \mathbb{R}^2; \quad \boldsymbol{X} \mapsto \boldsymbol{x} \tag{3.7}$$

An alternative way to represent the pinhole camera model is to flip the image plane: $(x, y) \mapsto (-x, -y)$, which appears to be a more convenient mathematical representation. The new image yields the same triangle relationship without the negative sign. Hence, the relationship between the image point $\boldsymbol{x}$ and the point p with coordinates $\boldsymbol{X}$ (relative to the camera frame) may be written as

$$Z\bar{\boldsymbol{x}} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \bar{\boldsymbol{X}}, \tag{3.8}$$

where $\bar{\boldsymbol{x}} := [x\ y\ 1]^T$ and $\bar{\boldsymbol{X}} := [X\ Y\ Z\ 1]^T$ are in homogeneous representation. The coordinate Z is often written as an arbitrary positive scalar $\lambda \in \mathbb{R}_+$ because the depth of a point p is usually unknown. Moreover, the above matrix is conveniently decomposed into

$$\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

The two matrices above are defined as

$$K_f := \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3}, \quad \Pi_0 := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 4}. \tag{3.9}$$

Let $\boldsymbol{X}_0 = [X_0 \ Y_0 \ Z_0]^T \in \mathbb{R}^3$ denote the coordinates of the point p relative to the world reference frame. As described in Chapter 2, the coordinates of the same point p relative to the camera frame may be expressed as a rigid body transformation using (2.11)

$$\bar{\boldsymbol{X}} = H\bar{\boldsymbol{X}}_0, \tag{3.10}$$

where $\bar{\boldsymbol{X}}_0, \bar{\boldsymbol{X}} \in \mathbb{R}^4$ are in homogeneous representation, and

$$H = \begin{bmatrix} R & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \in \mathrm{SE}(3). \tag{3.11}$$

$R \in \mathrm{SO}(3)$ and $\boldsymbol{t} \in \mathbb{R}^3$ denote the rotation matrix and the translation between the world reference frame and the camera frame, respectively. Using the above notation, the model for an ideal pinhole camera can be described as

$$\lambda\bar{\boldsymbol{x}} = K_f\Pi_0\bar{\boldsymbol{X}} = K_f\Pi_0 H\bar{\boldsymbol{X}}_0 \tag{3.12}$$

In a physical camera, measurements are obtained in terms of pixels, typically with the origin of the image frame in the upper-left corner of the image. In order to use the model (3.12) for a digital camera, it is necessary to derive a relationship between the retinal plane coordinate frame and the pixel array. It can be shown that the ideal image coordinates $\bar{\boldsymbol{x}} = [x \ y \ 1]^T$ can be transformed to actual image coordinates $\bar{\boldsymbol{x}}' = [x' \ y' \ 1]^T$ through the following transformation [29]

$$\bar{\boldsymbol{x}}' = K_s\bar{\boldsymbol{x}} = \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \bar{\boldsymbol{x}}, \tag{3.13}$$

where s_x, s_y are scaling factors and o_x, o_y are the translation (in pixels) of the origin relative to the optical axis.

Now, combining the ideal pinhole model with the scaling and translation yields a more realistic camera model applicable for digital cameras

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}. \tag{3.14}$$

The physical camera parameters are conveniently divided into extrinsic and intrinsic parameters. The extrinsic parameters is the rotation R and translation $\boldsymbol{t}$ which relates the world coordinate system to the camera coordinate system. Furthermore, the intrinsic camera parameters are the internal camera geometric and optical characteristics, including the focal length f and the coefficients s_x, s_y, o_x, c_y. These internal parameters are contained in the *camera intrinsics matrix* (or *camera matrix*):

$$K := K_s K_f := \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} fs_x & 0 & o_x \\ 0 & fs_x & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.15}$$

.

Without loss of generality, it is convenient to assume the object plane is on $Z_0 = 0$ of the world coordinate system [48]. For simplicity, let $\Pi_0 H = [\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{r}_3\ \boldsymbol{t}] \in \mathbb{R}^{3\times 4}$. Note that (3.14) may now be reduced to

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = K[\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{r}_3\ \boldsymbol{t}] \begin{bmatrix} X_0 \\ Y_0 \\ 0 \\ 1 \end{bmatrix} = K[\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{t}] \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix}.$$

From now on, by abuse of notation, $\boldsymbol{X}_0 = [X_0\ Y_0]^T$ and $\bar{\boldsymbol{X}}_0 = [X_0\ Y_0\ 1]^T$ since Z_0 is always equal to zero by assumption. Therefore, the spacial point $\boldsymbol{X}_0$ and its image point $\boldsymbol{x}'$ is related by a *homography* $\hat{H}$:

$$\lambda \bar{\boldsymbol{x}}' = \hat{H} \bar{\boldsymbol{X}}_0 \quad \text{with} \quad \hat{H} = K[\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{t}] \in \mathbb{R}^{3\times 3}. \tag{3.16}$$

In practice, very little light goes through the pinhole. Hence, such arrangements require time to accumulate enough light to process the image. For computer vision applications, it is desirable to have a high refresh rate on the imager, meaning that light must be gathered towards the pinhole. This is typically accomplished by utilizing a lens. However, the faster rate comes at the cost of introducing distortion [6]. Moreover, the miss-alignment between the glass dome and the camera, and refraction at the air-glass and glass-water interface introduces additional distortion [28].

The literature typically divides the distortion into *radial distortion* and *tangential distortion.* Radial distortion is the phenomenon of rays bending due to the curvature of the lens, where rays far from the center of the lens are bent more than the ones at the edges. Thus, the radial distortion is a polynomial function of distance away from the optical center, given by the following equation

$$\begin{aligned} \hat{x}_{radial} &= x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6), \\ \hat{y}_{radial} &= y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6), \end{aligned} \tag{3.17}$$

where x and y denote the location on the image plane, and $\hat{x}_{radial}$ and $\hat{y}_{radial}$ are the corrected location. On the other hand, tangential distortion is caused by production and assembly defects. The distortion is a result of the lens not being exactly parallel to the imaging plane, described by the equation [8]

$$\begin{aligned} \hat{x}_{tangential} &= x + [2p_1 y + p_2(r^2 + 2x^2)], \\ \hat{y}_{tangential} &= y + [p_1(r^2 + 2y^2) + 2p_2 x], \end{aligned} \tag{3.18}$$

The process of determining the intrinsic and/or extrinsic camera parameters, known as camera calibration, is presented in the coming section.

3.2.2 Implementation

The camera delivered with the BlueROV 2, Sony IMX322, was used to test the implementations. The vision sensor was connected directly to the Raspberry Pi 4 through a USB. Initially, the data was processed and foreword as a `sensor_msgs/image` message to the ROS network using a ROS Driver for V4L USB Cameras[2]. However, it turned out to be a computationally heavy implementation resulting in a significant delay and high CPU utilization. Even when the camera was the single node running on the Raspberry Pi, the topside computer experienced a image delay of more than 5 seconds. Hence, the pipeline-based multimedia framework GStreamer[3] was implemented for the benefit of low latency and low computational expense. In contrast, the new implementation efficiently streamed real-time video with 30 fps with minor delays, allowing for running the computer vision algorithms on the topside computer due to the low latency.

The lens and camera specifications are listed in Table 3.4. On the BlueROV2, the camera is places inside a water-tight glass housing, which introduces additional water-glass-air distortion. However, the glass housing is dome-formed in order to minimize this type of distortion, meaning the setup should be close to an ideal pinhole camera, even under water [28].

3.2.3 Calibration

Camera calibration is a critical factor in the overall performance of the computer vision system. Calibration is simply the process of determining the intrinsic matrix and the distortion parameters introduced for the camera sensor model. OpenCV utilizes a calibration method where the camera targets a known structure with many individual and identifiable points, such as a rectangular chessboard. By targeting the structure from various angles and positions, it is possible to compute the relative location and orientation and the camera's intrinsic

[2]`http://wiki.ros.org/usb_cam`

[3]`https://gstreamer.freedesktop.org/`

Table 3.4: Camera specifications.

Parameter	Value
Sensor	Sony IMX322
Number of recording pixels (HxB)	1980x1080
Frame rate	30
Readout mode	HD 1080p
Sensor format	1/2.9"
Focal length	2.97mm
Field of view (horizontal)	80°
Field of view (vertical)	64°
Distortion	1%

parameters. The algorithm used to determine the calibration parameters are base on two different methods; the calibration of focal lengths and offsets is based on Zhang's method [48] while the distortion parameters are determined utilizing the method derived in [7].

The first-mentioned method takes an image of the object plane (the chessboard) as an input. Given the image, an homography $\hat{H}$ can be estimated [48]. From (3.16) it is clear that

$$[\boldsymbol{h}_1\ \boldsymbol{h}_2\ \boldsymbol{h}_3] = sK[\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{t}]$$

where $s = 1/\lambda$ is an arbitrary scalar and $\hat{H} = [\boldsymbol{h}_1\ \boldsymbol{h}_2\ \boldsymbol{h}_3]$. Since any rotation matrix $R \in \mathrm{SO}(3)$ are orthogonal by construction and the scale is extracted from the rotation vectors it follows that $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ is orthonormal. This yield the two basic constraints on the intrinsic parameters, given one homography $\hat{H}$

$$\boldsymbol{h}_1^T K^{-T} K^{-1} \boldsymbol{h}_2 = 0 \tag{3.19}$$

$$\boldsymbol{h}_1^T K^{-T} K^{-1} \boldsymbol{h}_1 = \boldsymbol{h}_2^T K^{-T} K^{-1} \boldsymbol{h}_2. \tag{3.20}$$

Since each homography yields 8 degrees of freedom and there are six extrinsic parameters (three for rotation and three for translation), one can only obtain two constraints for intrinsic parameters.

For simplicity, let

$$
\begin{aligned}
B = K^{-T}K^{-1} :&= \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{(fs_x)^2} & 0 & \frac{-o_x}{(fs_x)^2} \\ 0 & \frac{1}{(fs_y)^2} & \frac{-o_y}{(fs_y)^2} \\ \frac{-o_x}{(fs_x)^2} & \frac{-o_y}{(fs_y)^2} & \frac{o_x^2}{(fs_x)^2} + \frac{o_y^2}{(fs_y)^2} + 1 \end{bmatrix}.
\end{aligned} \tag{3.21}
$$

Note that B is a symmetric matrix, defined by the vector

$$
\boldsymbol{b} = [B_{11}\ B_{12}\ B_{22}\ B_{13}\ B_{23}\ B_{33}]^T. \tag{3.22}
$$

The symmetry make it possible to write it as one six-dimensional vector dot product

$$
\begin{aligned}
\boldsymbol{h}_i^T B \boldsymbol{h}_j &= \boldsymbol{v}_{ij}^T \boldsymbol{b} \\
&= \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j2} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix} \boldsymbol{b},
\end{aligned} \tag{3.23}
$$

where $\boldsymbol{h}_i = [h_{i1}\ h_{i2}\ h_{i3}]^T$ is the ith column vector of $\hat{H}$. The two constraint (3.19) and (3.20) may now, for a given homography, be rewritten as two homogeneous equations in $\boldsymbol{b}$:

$$
\begin{bmatrix} \boldsymbol{v}_{12}^T \\ (\boldsymbol{v}_{11} - \boldsymbol{v}_{22})^T \end{bmatrix} \boldsymbol{b} = \boldsymbol{0}. \tag{3.24}
$$

If n images of the calibration object are observed, it is possible to stack n such equations as (3.24) into the matrix $V \in \mathbb{R}^{2n \times 6}$, such that

$$
V\boldsymbol{b} = \boldsymbol{0}. \tag{3.25}
$$

This implies that $n = 2$ images of the calibration object is enough to estimate $\boldsymbol{b}$. However, high quality results require at least ten images of 7-by-8 or lager chessboard, in addition to images with a "rich" set of views [6]. Once $\boldsymbol{b}$ is estimated, the intrinsic parameters can easily be extracted

$$\begin{aligned}
o_y &= (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2) \\
s &= B_{33} - [B_{13}^2 + o_y(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \\
fs_x &= \sqrt{s/B_{11}} \\
fs_y &= \sqrt{sB_{11}/(B_{11}B_{22} - B_{12}^2)} \\
o_x &= -B_{13}(fs_x)^2/s.
\end{aligned}$$

After the camera intrinsics are known, the extrinsic parameters for each image is readily computed. From (3.16), it is clear that

$$\boldsymbol{r}_1 = sK^{-1}\boldsymbol{h}_1, \quad \boldsymbol{r}_2 = sK^{-1}\boldsymbol{h}_2, \quad \boldsymbol{r}_3 = \boldsymbol{r}_1 \times \boldsymbol{r}_2, \quad \boldsymbol{t} = sK^{-1}\boldsymbol{h}_3.$$

with $s = 1/||K^{-1}\boldsymbol{h}_1|| = 1/||K^{-1}\boldsymbol{h}_2||$. Some care is required because noisy measurement will prevent the rotation vectors $\boldsymbol{r}$ to form an exact rotation matrix for which the properties of SO(3) holds. To get around this problem, singular value decomposition of R is taken.

Finally, the distortion parameters has to be determined. The first step starts with an initial guess to start solving a larger set of equations, using the camera intrinsics found previously together with the distortion parameters set to 0. The observed points (x_d, y_d) on the image plane will deviate from the location on a perfect pinhole camera owing to distortion. Let (x_p, y_p) be the point's location for a perfect pinhole camera, then

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} (fs_x)\dfrac{X_W}{Z_W} + o_x \\ (fs_y)\dfrac{Y_W}{Z_W} + o_y \end{bmatrix}, \tag{3.26}$$

where $\boldsymbol{X}_W = [X_W \ Y_W \ Z_W]^T$ is the spacial coordinate of an observed object, equivalent to the estimated translation $\boldsymbol{t}$. Further, from (3.17) and (3.18) it appears that the distortion can be represented as

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = (1 + k_1r^2 + k_2r^4 + k_3r^6)\begin{bmatrix} x_d \\ y_d \end{bmatrix} + \begin{bmatrix} 2p_1x_dy_d + p_2(r^2 + 2x_d^2) \\ p_1(r^2 + 2u_d^2) + 2p_2x_dy_d \end{bmatrix}. \tag{3.27}$$

Hence, the distortion parameters can be estimated by solving a larger collected list of these equations, in parallel with re-estimating the intrinsics and extrinsics [6]. This is all done in the single function `calibrateCamera()` in OpenCV.

Before calling the calibration function in OpenCV, it is necessary to follow some simple steps. Firstly, the colored camera image is converted to a grayscaled image using the function `cvtColor()`. The grayscale image is a robust way to improve the detection and separate the chessboard from the environment. Further, the function `findChessboardCorners()` takes the chessboard size as input and returns the number of corners detected and their location. The returned corners are only approximate as the locations are accurate only to within the limits of the imaging device; in other words, one pixel precision. In order to compute the exact locations of the corners, the function `cornerSubPix()` computes the locations to subpixel precision given the approximated corner locations. Subpixel refinement is an essential step in the camera calibration process, and neglection can cause substantial calibration errors. Further, drawing the detected corners using `drawChessboardCorners()` is desirable for debugging in order to verify that the detected corner locations correspond to the correct location on the chessboard (see Figure 3.3a).

Simulation

The UUV simulator [30] was desirable to use in this thesis because of its visualization feature. The simulator uses Gazebo to visualize the robot's motion in a virtual environment, with the possibility to generate a simulated camera view (see Figure 3.3a). Similarly to a real camera, the simulated view is based on a pinhole model, which requires calibration. The chessboard shown in Figure 3.3a was implemented using an open-source package[4]. The lens distortion introduced in (3.17) and (3.18) was estimated to be

$$(k_1\ k_2\ p_1\ p_2\ k_3) = (-0.0027 \quad 0.0298 \quad 0.0003 \quad 0.0011 \quad -0.1081), \tag{3.28}$$

after capturing 15 images of the chessboard with various viewing angles. Moreover, the calibration obtained the following camera matrix introduced in (3.15):

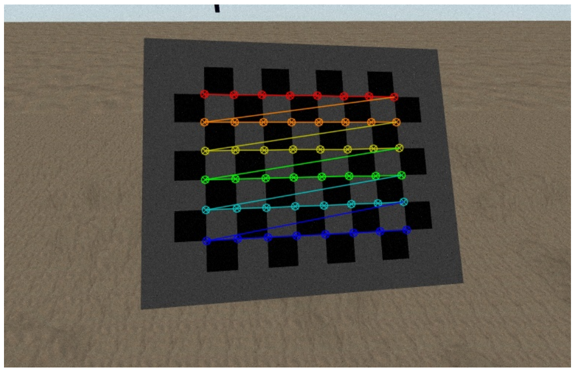
$$K = \begin{bmatrix} 407.1576 & 0 & 384.7448 \\ 0 & 407.0851 & 245.5095 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.29}$$

Experiment

The calibration of the Sony IMX322 camera was done using a slightly different toolbox, namely the ROS Kalibr toolbox[5]. The reason why rejecting the OpenCV method was the available calibration target at the test facility at NTNU. Instead, an Aprilgrid created by previous master's students in [12] was used to calibrate the camera (see Figure 3.3b). The advantage of utilizing this calibration board is its large scale and aluminum frame, yielding a completely straight surface. Moreover, the calibration target consists of $5x5$ Apriltags, with a marker size of 8.03 cm and a white spacing of 2.24 cm. Note that the Kalibr toolbox rejects the radial distortion parameter k_3 introduced in (3.17).

[4] `https://github.com/oKermorgant/calibration_gazebo`

[5] `https://github.com/ethz-asl/kalibr`


(a) Simulated camera calibration target. Detected corners and their locations are marked on the simulated camera image.

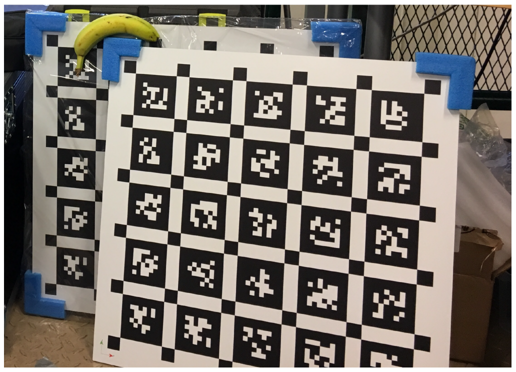

(b) 5x5 Aprilgrid calibration boards printed on aluminum frames. The board size is 60x60 cm with a banana for scale. Courtesy of [12].

Figure 3.3: Camera calibration targets.

Another valuable feature of the Kalibr toolbox is the calibration report obtained. The final calibration ended with a reprojection error of ± 0.2524 and ± 0.2110 for the x and y pixels. The toolbox states that a reprojection below 0.25 is acceptable; however, there is no clear boundary between "good" and "bad" calibration. Moreover, the camera matrix introduced in (3.15) was estimated to

$$K = \begin{bmatrix} 454.3053 & 0 & 347.3016 \\ 0 & 453.7248 & 228.7860 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.30}$$

while the distortion parameters parameters introduced in (3.17) and (3.18) were estimated to

$$(k_1 \; k_2 \; p_1 \; p_2 \; k_3) = (-4.49 \quad 13.19 \quad 0.29 \quad 4.94 \quad 0.00) \cdot 10^{-3}. \tag{3.31}$$

Finally, it should be noted that the calibration was performed in air. Hence, the calibration does not account for the effect of air-glass-water refractions. However, as argued in [28], the glass dome should ideally eliminate most of the refraction effect.

Chapter 4

Computer Vision

Vision is vital for humans to observe and orient themselves in the world. Similarly, computer vision is valuable for autonomous motion control of robots, such as in collision avoidance and state estimation. This chapter presents the application of computer vision for position and attitude estimation using the open-source computer vision library OpenCV.

The image projection results in the loss of one dimension in the image capturing. Hence, it is necessary to determine a correspondence between the points in the real environment and the 2D projection. These challenges have led to numerous publications in the field of *pattern recognition* for image analysis. One approach is the use of binary square fiducial markers, with the main benefit of a single marker providing enough correspondence to obtain the camera pose, see, e.g., [18].

4.1 ArUco Marker

ArUco is a squared marker composed of a wide black border and an inner binary matrix that determines its identifier (id). The black edge contributes to its fast detection in the image, while the binary codification allows its identification and the application of error detection and correction techniques [33]. Moreover, the marker size determines the size of the internal matrix. For instance, the $5x5$ marker in Figure 4.1 is composed by 25 bits.

The ArUco module in OpenCV is based on the ArUco library [17] for the detection of square fiducial markers developed by [18].

4.1.1 Marker Detection

Detection of ArUco markers comprises several steps to detect rectangle-shaped marker candidates and extract the binary code from them. The detection approach employed by [18] are described in the following:

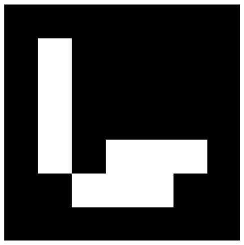


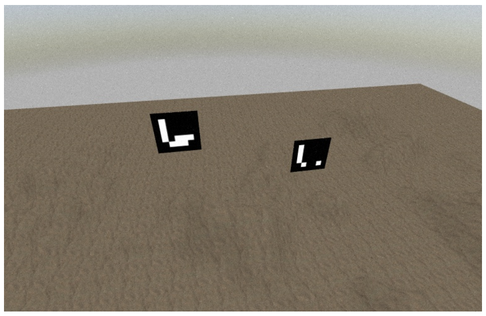
Figure 4.1: ArUco marker.

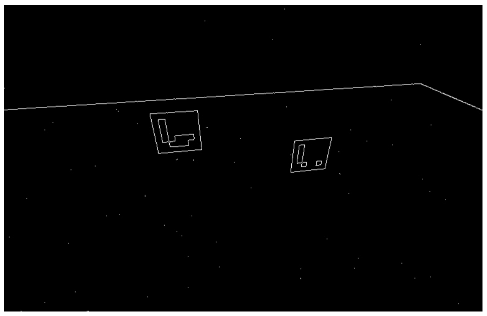
- *Image segmentation*: The first step extracts the most prominent contours in the gray-scale image. One approach is the Canny edge detection [9], shown in Figure 4.2b. However, this method is computationally extensive for real-time purposes. Therefore, the ArUco library utilizes a local adaptive threshold approach which has proven to be robust to different lighting conditions [18].

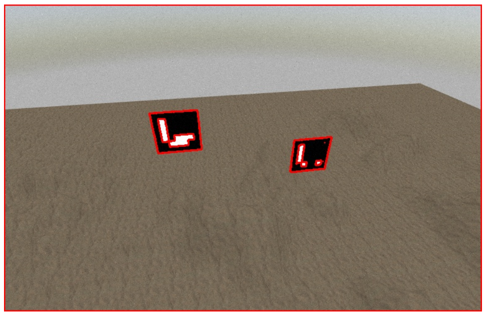
- *Contour extraction and filtering*: Further, the Suzuki and Abe [43] algorithm is applied to extract the contours from the thresholded image. It produces the set of image contours, most of which are irrelevant for marker detection (see Figure 4.2c). A polygonal approximation is then applied using the Douglas-Peucker [11] algorithm. The algorithm filter out contours that are not approximated to 4-vertex polygons, leaving us with potential marker contour candidates.

- *Marker Code extraction*: The next step consists of analyzing the inner region of the marker candidates to extract its internal code. First, the homography matrix is computed and applied to remove the perspective projection. The resulting image is thresholded using Otsu's method [34], which provides the optimal threshold value given the image distribution is bimodal (which holds true in the case of binary square markers). The binarized image is then divided into a grid depending on the marker size. Each grid element holds the value 0 or 1 depending on the values of the majority of pixels into it. A first rejection test consists of detecting the black border. Then, if the black frame is present, the inner grid is analyzed using the method described below.

- *Marker identification*: The last step is determining which of the marker candidates obtained belongs to the ArUco dictionary, $\mathcal{D}$. Once the marker code is extracted, four identifiers are obtained (one for each possible rotation). If any of the candidates are found in $\mathcal{D}$, it is considered a valid marker. The dictionary is sorted as a balanced binary tree to speed up the detection process. To that aim, the marker is represented by the integer value obtained by concatenating all its bits. This process has a logarithmic complexity of $\mathcal{O}(4\log_2(|\mathcal{D}|))$, where the factor 4 is a direct consequence of the four


(a) Simulated camera image.


(b) Canny edge detection.


(c) Contour extraction and filtering

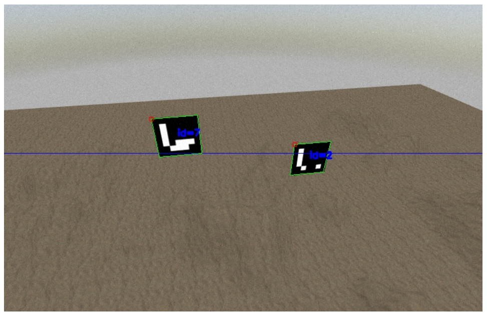

(d) Marker detection and identification.

Figure 4.2: ArUco marker detection using the Gazebo-based UUVsimulator. The ArUco marker model is adopted from an open-source package[1].

necessary rotations for each search. Moreover, a correction method is applied if no match is found, see [18].

The detection process is all done in a single OpenCV function, `detectMarker()`, which takes a grayscaled camera image as input, and return the id(s) and an array of corners of the detected marker(s).

4.1.2 Pose Estimation

When a marker is detected, it is possible to estimate its pose relative to the camera frame by solving the *Perspective-n-Point* (PnP) problem. Given the camera intrinsic parameters and the n marker corners, the PnP problem is solved by iteratively minimizing the reprojected error of the corners. There are multiple approaches for pose estimation, e.g., the Levenberg-Marquardt algorithm [31, 20].

Among the various algorithms for solving the PnP problem, there are often different objectives. The performance measures are typically run-time and rotation error, where one comes at the cost of the other. While time-efficient algorithms use fewer iteration, the high

[1] `https://github.com/joselusl/aruco_gazebo`

accuracy algorithms require high-resolution images, making the increasing number of iterations computational heavy. Like most other computer vision applications, the accuracy of the camera calibration is crucial for the accuracy of the solution. Moreover, the number of correspondences n, which in the case of ArUco markers corresponds to the number of corners, is crucial for the rotation estimate. Additionally, the spacing affects the pose estimate, where increasing the number and spacing between the correspondences leads to higher accuracy. For the interested reader, see, e.g., [26].

In OpenCV, the pose estimation is done in the function `estimatePoseSingleMarker()`, inherited from the ArUco library. Similarly to the camera calibration, it is necessary to perform a corner refinement in order to get sub-pixel precision. To account for this, the ArUco library has opted for doing a linear regression of the marker side pixels to calculate their intersection. The function takes the marker size, the corner locations, and the camera matrix and distortion parameters as input. Given this information, the function computes the rotation and translation vectors of the marker relative to the camera frame.

The PnP-problem estimates the relative pose between the camera frame $\mathcal{F}_c$ and the target frame $\mathcal{F}_m$ (ArUco marker). However, the vehicle's attitude can be computed by the recursive relation given in (2.14) by considering the coordinate frames as one rigid body at the time instant of the pose measurement. Given the measured translation $\boldsymbol{t} = [t_x \ t_y \ t_z]^T \in \mathbb{R}^3$ and rotations $\boldsymbol{r} = [r_x \ r_y \ r_z]^T \in \mathbb{R}^3$, the homogeneous transformation from $\mathcal{F}_c$ to $\mathcal{F}_m$ is given as

$$H_m^c = \begin{bmatrix} R & \boldsymbol{t} \\ 0_{3\times 1} & 1 \end{bmatrix},$$

where the rotation matrix are formed by the measured rotations following the zyx-convention, i.e., $R = R_{z,r_z} R_{y,r_y} R_{x,r_x} \in \mathrm{SO}(3)$. Because the map from $\mathcal{F}_b$ to $\mathcal{F}_c$ is known from the vehicle's geometry, the global pose estimate is computed by the recursive relation,

$$\begin{aligned} H_b^n &= H_m^n H_c^m H_b^c \\ &= \begin{bmatrix} R_b^n & \boldsymbol{d}_{nb}^b \\ 0_{3\times 1} & 1 \end{bmatrix} \in \mathrm{SE}(3). \end{aligned} \tag{4.1}$$

4.2 Camera Coordinate Frame

The camera-measured translation and rotation are expressed relative to the camera frame $\mathcal{F}_c$. In photogrammetry, the z-axis is commonly defined perpendicular to the image plane and centered at the optical center [6, 29]. Moreover, the x and y axes span the image plane; see Figure 3.2. However, OpenCV uses a rather strange definition of the x and y axes, pointing to the right and down direction of the image plane, respectively. For example, suppose the

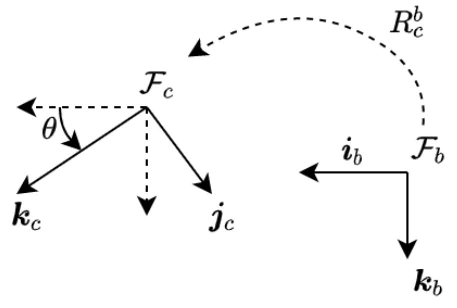


Figure 4.3: 2D illustration of the Body-fixed frame $\mathcal{F}_b = (o_b, \boldsymbol{i}_b, \boldsymbol{j}_b, \boldsymbol{k}_b)$ and the camera frame $\mathcal{F}_c = (o_c, \boldsymbol{i}_c, \boldsymbol{j}_c, \boldsymbol{k}_c)$.

camera is placed such that the image plane is perpendicular to the x-axis of $\mathcal{F}_b$, i.e., it captures what's in front of ROV. The rotation matrix from $\mathcal{F}_b$ to $\mathcal{F}_c$ is then

$$R = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}. \tag{4.2}$$

As the camera has a field of view that forms a triangle, a horizontally mounted camera will capture parts of what's above the UUV. Hence, it is convenient to to tilt the camera frame about the y-axis to capture more of what's below the vehicle (see Figure 4.3). The new rotation matrix is now

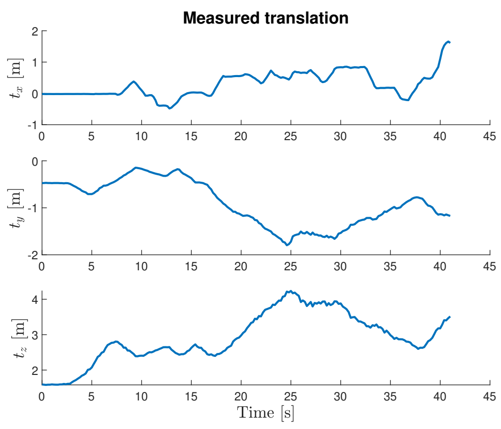
$$R_c^b = RR_{y,\theta}, \tag{4.3}$$

where R, $R_{y,\theta}$, and θ are the rotation matrix in (4.2), the principal rotation matrix given in (2.5), and the angle of rotation, respectively.

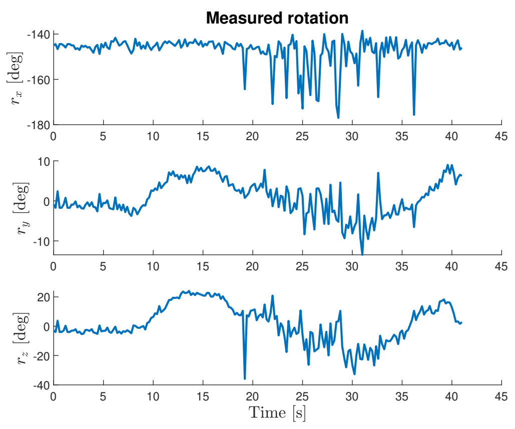
4.3 Accuracy

Before moving on to the observer design, it is necessary to determine the accuracy of the ArUco pose measurement. Therefore, a simple simulation study where an ArUco marker was placed in front of a simulated camera (similar to Figure 4.2) was used to evaluate the accuracy of the raw pose measurements. The measured translation $\boldsymbol{t} = [t_x \; t_y \; t_z]^T \in \mathbb{R}^3$ and rotations $\boldsymbol{r} = [r_x \; r_y \; r_z]^T \in \mathbb{R}^3$ for some random motions are shown in Figure 4.4. Apperently, the raw rotation measurement suffers significant inaccuracies with large variations.

Additionally, one extra simulation case was performed to discover the influence of the rotation measurement noise on the final pose estimate. It involved computing the vehicle's pose based on the forward kinematic equation derived in (4.1), and the result is shown in Figure



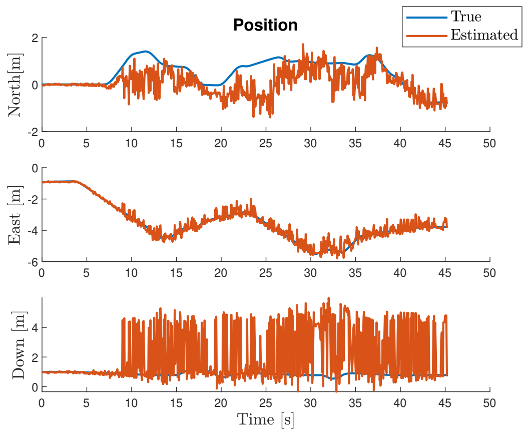
(a) Raw translation measurement.



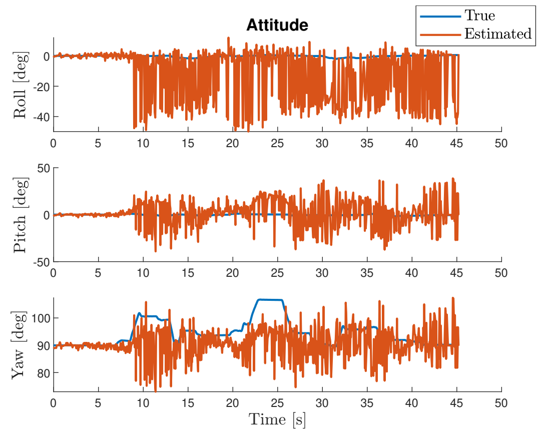
(b) Raw rotation measurement.

Figure 4.4: Measured translation and rotation of the ArUco marker frame $\mathcal{F}_m$ relative to the camera frame $\mathcal{F}_c$. The translation and rotation are measured using the ArUco library in OpenCV.

4.5. During the simulation, the vehicle with the mounted camera was kept stationary for the first 5 seconds, followed by slow motions in surge, sway, heave, and yaw. The two remaining degrees were approximately zero as the simulated vehicle model was strongly stabilizable in roll and pitch, and the controller was not designed to actuate these degrees.



(a) Position estimate.



(b) Attitude estimate.

Figure 4.5: Pose estimation using raw measurements from the ArUco library. The blue and orange line represents the true and estimated value, respectively. Each plot represents one of the six degrees of freedom where the attitude is parameterized using Euler angle representation.

From Figure 4.5, it is clear that the estimation suffers significant inaccuracies in both position and attitude when the vehicle starts moving. Interpretation of the result in Figure 4.4, the inaccuracy is primarily caused by the measured rotation R, which implicitly affects the position estimation. Comparing the result with Table 1.2, it is clear that the system would benefit from using a compass rather than the computer vision attitude estimate. On the other hand, the measured translation turned out to be significantly less noisy than the rotation measurement (see Figure 4.4a). Hence, it is desirable to include the translation and reject the rotation measurement in the design of the navigation system.

Chapter 5

Navigation System

Position and attitude estimation are crucial in motion control of vehicles, either it is aerospace, surface, or underwater applications. An essential fact of motion control systems is that the closed-loop system has no better performance than its poorest subsystem. Hence, it is no reason to have a high-performance control system if the state estimate has significant uncertainties. This chapter proposes an observer design to fuse the state-of-the-art inertial navigation system and the computer vision pose estimate using an extended Kalman filter (EKF).

5.1 Inertial Navigation Systems

The major challenge in underwater navigation is radio waves rapidly fading in water. A consequence of the infeasibility of radio waves is the Global Navigation Satellite Systems (GNSS), a critical component in surface navigation systems, being unsuitable for underwater applications. Additionally, it restricts wireless communication with the vehicle, requiring autonomous control using onboard sensors. For underwater vehicles, a widely used approach is inertial navigation through measurements (in the body-fixed frame $\mathcal{F}_b$) of some vectors in the inertial frame. Accelerometers, gyroscopes, and magnetometers included in small micro-electromechanical systems (MEMS)-based IMU sensors are examples of sensors used to observe such vectors. To get a brief introduction to inertial navigation systems, see, e.g., [47].

Let $R_b^n \in \mathrm{SO}(3)$ be the attitude of the body-fixed frame $\mathcal{F}_b$ with respect to the inertial NED frame $\mathcal{F}_n$, the kinematic model of a rigid-body motion in a 3-dimensional space is then given by

$$\begin{aligned}\dot{R}_b^n &= R_b^n[\boldsymbol{\omega}]_\times \\ \dot{\boldsymbol{p}} &= \boldsymbol{v} \\ \dot{\boldsymbol{v}} &= \boldsymbol{g}^n + R_b^n\boldsymbol{a},\end{aligned} \tag{5.1}$$

where $\boldsymbol{\omega} \in \mathbb{R}^3$ denotes the angular velocity, $\boldsymbol{p} \in \mathbb{R}^3$ denotes the position, $\boldsymbol{v} \in \mathbb{R}^3$ denoted the velocity, and $\boldsymbol{a} \in \mathbb{R}^3$ denote the apparent acceleration capturing all non-gravitational forces. All vector quantities given in $\mathcal{F}_b$ and expressed relative to $\mathcal{F}_n$, except the gravitational acceleration, $\boldsymbol{g}^n \in \mathbb{R}^3$, which is independent of $\mathcal{F}_b$.

5.1.1 Vision-based Position Aiding

From (3.1), it is clear that the inherent error terms in the angular velocity and acceleration model of the IMU lead to unbounded drift in the propagated navigation solution. Hence, it is common practice to include position aiding to eliminate the unbounded drift using either acoustic transponders and modems or geophysical techniques. However, as mentioned in Chapter 1, acoustic transponders require externally located transponders and lack high precision. On the other side, the geophysical SLAM approaches occasionally suffer poor estimates due to misinterpretations of similar landmarks.

In contrast to most SLAM-based AUVs, subsea-resident robots operate in a confined area near the subsea site. In other words, instead of using random features on the seafloor, which is challenging to detect and classify, it is possible to equip the site with binary square fiducial markers introduced in Chapter 4 (see Figure 4.1). Hence, instead of the robot autonomously building the map consisting of features with uncertain positions, it is desirable to predefine a map with the exact location of the fiducial markers. Furthermore, the robust and fast detection of such markers eliminates the challenge of classifying and distinguishing different features.

The pose estimate is more accurate when the vehicle is close to the landmark; thus, it is reasonable to have a higher trust in the position measurements of the nearest markers. This is accounted for by a weighted discounted average for each observation i

$$w_i = \frac{\gamma_i}{\sum_{j=1}^{N} \gamma_j}, \tag{5.2}$$

where N is the number of detected markers, and $\gamma_i \in (0, 1]$ is the discount factor of the position estimate for marker i. Note that a small γ_i results in observation i being less influential on the resulting estimate. In order to give higher trust to the nearest markers, the discount factor is based on the inverse Euclidean norm of the vector between $\mathcal{F}_s$ and the i-th marker frame

$\mathcal{F}_{m_i}$

$$\gamma_i = \begin{cases} ||\boldsymbol{y}_{p,i}||^{-1}, & \text{if} \quad ||\boldsymbol{y}_i||^{-1} > 1 \\ 1, & \text{otherwise,} \end{cases} \tag{5.3}$$

where $\boldsymbol{y}_{p,i} \in \mathbb{R}^3$ is the measured marker position expressed in the sensor frame $\mathcal{F}_s$, and $||\cdot||$ is the Euclidean norm defined as $||\boldsymbol{x}|| = \sqrt{\boldsymbol{x}^T\boldsymbol{x}}$ for a vector $\boldsymbol{x} \in \mathbb{R}^n$. Now, let $\boldsymbol{p}_i \in \mathbb{R}^3$ be the position $\mathcal{F}_{m_i}$ expressed in $\mathcal{F}_n$. Because the translation $\boldsymbol{t}_i \in \mathbb{R}^3$ is measured in $\mathcal{F}_c$, it is necessary to transform the measurement into $\mathcal{F}_s$ using (2.10)

$$\boldsymbol{y}_{p,i} = R_c^s \boldsymbol{t}_i + \boldsymbol{d}_{sc}^s, \tag{5.4}$$

where R_c^s is the rotational matrix which transforms vectors expressed in $\mathcal{F}_c$ to $\mathcal{F}_s$, and $\boldsymbol{d}_{sc}^s$ is the translation vector from $\mathcal{F}_s$ to $\mathcal{F}_c$ expressed in $\mathcal{F}_s$. Given the weights w_i and the position measurements $\boldsymbol{y}_{p,i}$, the position estimate of $\mathcal{F}_s$ with respect to $\mathcal{F}_n$ is given by

$$\boldsymbol{y}_p = \sum_{i=1}^{N} w_i(\boldsymbol{p}_i - \hat{R}_s^n \boldsymbol{y}_{p,i}), \tag{5.5}$$

where $\hat{R}_s^n$ is the estimated rotation matrix. The advantage of transforming the measurements into $\mathcal{F}_s$ rather than $\mathcal{F}_b$ is a simplified observer design, avoiding the transformation of the inertial measurements [15].

5.1.2 Vector-based Attitude Aiding

Attitude determination is the problem of estimating the rotation matrix relative to the inertial world frame. Due to its great importance, the attitude determination problem gained a lot of interest, especially among the spacecraft communities during the space race in the late '50s. One of the first principles in spacecraft attitude determination was to use observed features, such as stars, to estimate the attitude. The mathematical problem of finding the rotation matrix, A, between two coordinate systems from a set of vector observations (measurements) was first posed by Grace Wahba in [46],

$$A\boldsymbol{V}_i = \boldsymbol{W}_i \quad \text{for } i = 1, ..., n, \tag{5.6}$$

where $\boldsymbol{W}_1, ..., \boldsymbol{W}_n$ and $\boldsymbol{V}_1, ..., \boldsymbol{V}_n$ are a set of observed unit-vectors and reference unit-vectors in $\mathbb{R}^3$, respectively. Because the observation unit-vectors are corrupted by a error, a solution of A does not exist in general, not even for $n = 2$ [37]. Hence, Wahba's problem becomes a optimization problem that seeks to minimize the following cost function

$$L(A) = \frac{1}{2}\sum_{i=1}^{N} a_i||\boldsymbol{W}_i - A\boldsymbol{V}_i||^2, \tag{5.7}$$

where the weight a_i decides how influential observation i is on the attitude estimate.

The TRIAD Algorithm

Given two nonparallel reference unit-vectors $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ and the corresponding observation unit-vectors $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, there exist a solution of (5.6) because the matrix A is overdetermined [37]. By constructing two triads of manifestly orthonormal reference and observation vectors according to

$$\boldsymbol{r}_1 = \boldsymbol{V}_1, \quad \boldsymbol{r}_2 = \frac{[\boldsymbol{V}_1]_\times \boldsymbol{V}_2}{\|\ [\boldsymbol{V}_1]_\times \boldsymbol{V}_2\ \|}, \quad \boldsymbol{r}_3 = \frac{[\boldsymbol{V}_1]_\times([\boldsymbol{V}_1]_\times \boldsymbol{V}_2)}{\|\ [\boldsymbol{V}_1]_\times \boldsymbol{V}_2\ \|} \tag{5.8}$$

$$\boldsymbol{s}_1 = \boldsymbol{W}_1, \quad \boldsymbol{s}_2 = \frac{[\boldsymbol{W}_1]_\times \boldsymbol{W}_2}{\|\ [\boldsymbol{W}_1]_\times \boldsymbol{W}_2\ \|}, \quad \boldsymbol{s}_3 = \frac{[\boldsymbol{W}_1]_\times([\boldsymbol{W}_1]_\times \boldsymbol{W}_2)}{\|\ [\boldsymbol{W}_1]_\times \boldsymbol{W}_2\ \|} \tag{5.9}$$

there exist a unique orthogonal matrix A given by

$$A = M_{\text{obs}} M_{\text{ref}}^T, \tag{5.10}$$

with

$$M_{\text{ref}} = [\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{r}_3], \quad M_{\text{obs}} = [\boldsymbol{s}_1\ \boldsymbol{s}_2\ \boldsymbol{s}_3]. \tag{5.11}$$

This attitude determination method is widely used in navigation and is known as the TRIAD algorithm.

5.2 Multiplicative Extended Kalman Filter

The widely used Kalman Filter (KF) is an estimator for the instantaneous state of a linear dynamic system perturbed by Gaussian white noise, known as the linear quadratic Gaussian (LQG) problem. The filter accounts for the measurement noise resulting in a statistically optimal estimator with respect to any quadratic function of estimation error. R. E. Kalman first introduced the approach in [23], and it is considered one of the most significant discoveries in the history of statistical estimation theory.

Although Kalman filters are designed for linear dynamic systems, most encountered systems possess a nonlinear dynamic modeled stochastically in the form

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}, t) + E(\boldsymbol{x})\boldsymbol{n}_x \tag{5.12a}$$

$$\boldsymbol{y} = h(\boldsymbol{x}, t) + N(\boldsymbol{x})\boldsymbol{n}_y, \tag{5.12b}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^m$, and $\boldsymbol{y} \in \mathbb{R}^p$ are the state, input and output, respectively. $\boldsymbol{n}_x \sim \mathcal{N}(0, Q)$ and $\boldsymbol{n}_y \sim \mathcal{N}(0, V)$ are independent zero-mean Gaussian noises, and $E(\boldsymbol{x}) \in \mathbb{R}^{n\times n}$ and $N(\boldsymbol{x}) \in \mathbb{R}^{p\times p}$ are matrix-valued function of $\boldsymbol{x}$. Due to this fact, a modified version that continuously linearizes the system about the current state to achieve the linear dynamics was introduced, namely the *extended Kalman filter* (EKF). However, the rotation

$R \in \mathrm{SO}(3)$ cannot be applied directly as the EKF assumes that all state variables are vectors. Hence, the literature [15, 40, 47] proposes several parametrizations of the rotation matrix R, such as the Euler angle representation and the unit-quaternion representation. However, all three-parameter Euler angle representations possess singularities that unit-quaternions do not. Depending on the dynamical system, the Euler representation is suitable for objects operating far from the singularities. Underwater vehicles are typically not such a system, as they are modeled and highly actuated in all 6-DOFs. Therefore, the globally non-singular unit-quaternion representation is the preferable parametrization of R (see Chapter 2 for more details about unit-quaternions).

The non-unique attitude representation leads to possible issues in preserving the motion space of the unit-quaternion when using linear estimation error $\boldsymbol{q} - \hat{\boldsymbol{q}}$. In order to avoid this issue, the attitude estimation can be represented using a multiplicative quarternion estimation error, thereby the name *multiplicative extended Kalman filter* (MEKF) [25]. The multiplicative estimation error is defined as

$$\tilde{\boldsymbol{q}} = \boldsymbol{q}_t \otimes \boldsymbol{q}^{-1}, \tag{5.13}$$

where $\otimes$ denote the quaternion product given in (2.18). $\boldsymbol{q}$ may be thought of as a perturbed orientation due to measurement noise, comprised of an unperturbed orientation $\boldsymbol{q}_t$ with a small local perturbation $\tilde{\boldsymbol{q}}$. Let the quaternion error be expressed through the exponential map given in (2.30)

$$\tilde{\boldsymbol{q}} = \mathrm{Exp}(\tilde{\phi}\tilde{\boldsymbol{u}}) = \begin{bmatrix} \cos(\tilde{\phi}/2) \\ \tilde{\boldsymbol{u}}\sin(\tilde{\phi}/2) \end{bmatrix}, \tag{5.14}$$

where $\tilde{\phi} \in \mathbb{R}$ represent the angle displacement due to the perturbation, and $\tilde{\boldsymbol{u}} \in \mathbb{R}^3$ represent the axis of rotation. If the local perturbation ϕ is sufficiently small, the estimation error can be approximated to

$$\tilde{\boldsymbol{q}} \approx \begin{bmatrix} 1 \\ \tilde{\boldsymbol{u}}\tilde{\phi}/2 \end{bmatrix} = \begin{bmatrix} 1 \\ \tilde{\boldsymbol{\theta}}/2 \end{bmatrix}, \tag{5.15}$$

by the small-angle approximation; $\cos(\theta) \approx 1$ and $\sin(\theta) \approx \theta$. For conviniency, the short notation $\tilde{\boldsymbol{\theta}} := \tilde{\phi}\tilde{\boldsymbol{u}}$ is used exclusively in the rest of the thesis. In order to satisfy the unity constraint, the approximation has to be normalized

$$\tilde{\boldsymbol{q}} \approx \frac{1}{\sqrt{4 + ||\tilde{\boldsymbol{\theta}}||^2}} \begin{bmatrix} 2 \\ \tilde{\boldsymbol{\theta}} \end{bmatrix}. \tag{5.16}$$

Table 5.1 summarizes all the variables in the proposed error-state MEKF . The system has error-state $\tilde{\boldsymbol{x}}$, governed by the input from the IMU readings $\boldsymbol{u}$, and perturbed by the

Table 5.1: All variables in the error-state MEKF.

	True	Nominal	Error	Composition	Measured	Noise
State	$\boldsymbol{x}_t$	$\boldsymbol{x}$	$\tilde{\boldsymbol{x}}$	$\boldsymbol{x}_t = \boldsymbol{x} \oplus \tilde{\boldsymbol{x}}$		
Position	$\boldsymbol{p}_t$	$\boldsymbol{p}$	$\tilde{\boldsymbol{p}}$	$\boldsymbol{p}_t = \boldsymbol{p} + \tilde{\boldsymbol{p}}$	$\boldsymbol{y}_p$	$\boldsymbol{w}_p$
Velocity	$\boldsymbol{v}_t$	$\boldsymbol{v}$	$\tilde{\boldsymbol{v}}$	$\boldsymbol{v}_t = \boldsymbol{v} + \tilde{\boldsymbol{v}}$		
Quaternion	$\boldsymbol{q}_t$	$\boldsymbol{q}$	$\tilde{\boldsymbol{q}}$	$\boldsymbol{q}_t = \boldsymbol{q} \otimes \tilde{\boldsymbol{q}}$		
Angles vector			$\tilde{\boldsymbol{\theta}}$	$\tilde{\boldsymbol{q}} = \mathrm{Exp}(\tilde{\boldsymbol{\theta}})$		
Accelerometer bias	$\boldsymbol{b}_{acc,t}$	$\boldsymbol{b}_{acc}$	$\tilde{\boldsymbol{b}}_{acc}$	$\boldsymbol{b}_{acc,t} = \boldsymbol{b}_{acc} + \tilde{\boldsymbol{b}}_{acc}$		$\boldsymbol{w}_{b,acc}$
Gyrometer bias	$\boldsymbol{b}_{gyro,t}$	$\boldsymbol{b}_{gyro}$	$\tilde{\boldsymbol{b}}_{gyro}$	$\boldsymbol{b}_{gyro,t} = \boldsymbol{b}_{gyro} + \tilde{\boldsymbol{b}}_{gyro}$		$\boldsymbol{w}_{b,gyro}$
Acceleration	$\boldsymbol{a}_t$				$\boldsymbol{a}_{imu}$	$\boldsymbol{w}_{acc}$
Angular rate	$\boldsymbol{\omega}_t$				$\boldsymbol{\omega}_{imu}$	$\boldsymbol{w}_{gyro}$
Magnetic field	$\boldsymbol{m}_t$				$\boldsymbol{m}_{mag}$	$\boldsymbol{w}_{mag}$

impulses vector $\boldsymbol{n}_x$

$$\dot{\tilde{\boldsymbol{x}}} = \begin{bmatrix} \dot{\tilde{\boldsymbol{p}}} \\ \dot{\tilde{\boldsymbol{v}}} \\ \dot{\tilde{\boldsymbol{\theta}}} \\ \dot{\tilde{\boldsymbol{b}}}^s_{acc} \\ \dot{\tilde{\boldsymbol{b}}}^s_{gyro} \end{bmatrix}, \quad \boldsymbol{u} = \begin{bmatrix} \boldsymbol{a}^s_{imu} \\ \boldsymbol{\omega}^s_{imu} \end{bmatrix}, \quad \boldsymbol{n}_x = \begin{bmatrix} \boldsymbol{w}^s_{acc} \\ \boldsymbol{w}^s_{gyro} \\ \boldsymbol{w}^s_{b,acc} \\ \boldsymbol{w}^s_{b,gyro} \end{bmatrix}, \tag{5.17}$$

where the superscript s denote the sensor frame $\mathcal{F}_s$ of which the states are measured, more precisely, the position and orientation of the inertial measurement unit (IMU). Combining the kinematic model given in (5.1) and the IMU model in (3.1) yield the following nonlinear stochastic model [40],

$$f(\tilde{\boldsymbol{x}}, \boldsymbol{u}) = \begin{pmatrix} \tilde{\boldsymbol{v}} \\ -R^n_s[\boldsymbol{a}^s_{imu} - \boldsymbol{b}^s_{acc}]_\times \tilde{\boldsymbol{\theta}} - R^n_s \tilde{\boldsymbol{b}}^s_{acc} \\ -[\boldsymbol{\omega}^s_{imu} - \boldsymbol{b}^s_{gyro}]_\times \tilde{\boldsymbol{\theta}} - \tilde{\boldsymbol{b}}^s_{gyro} \\ 0 \\ 0 \end{pmatrix}, \quad E(\tilde{\boldsymbol{x}}) = \begin{pmatrix} 0 \\ -R^n_s \boldsymbol{w}^s_{acc} \\ -\boldsymbol{w}^s_{gyro} \\ \boldsymbol{w}^s_{b,acc} \\ \boldsymbol{w}^s_{b,gyro} \end{pmatrix} \tag{5.18}$$

where R^n_s is the rotation matrix defining the orientation of $\mathcal{F}_s$ relative to $\mathcal{F}_n$.

The inertial information, namely the acceleration and angular rate measurements, serves to make predictions to the error-state Kalman filter, referred to as dead reckoning. On the other hand, the information from the remaining sensors contributes to correct, or aid, the prediction, thereby the name *aided inertial navigation system.* The measurements have error-state $\tilde{\boldsymbol{y}}$ perturbed by the measurement noise vector $\boldsymbol{n}_y$,

$$\tilde{\boldsymbol{y}} = \begin{bmatrix} \tilde{\boldsymbol{y}}_p \\ \tilde{\boldsymbol{y}}_g \\ \tilde{\boldsymbol{y}}_{mag} \end{bmatrix}, \quad \boldsymbol{n}_y = \begin{bmatrix} \boldsymbol{w}_p^s \\ \boldsymbol{w}_g^s \\ \boldsymbol{w}_{mag}^s \end{bmatrix}, \tag{5.19}$$

where subscript p, g, and mag represents the measured position, gravitation vector, and magnetic field vector, respectively. In order to utilize the aiding sensors in the error-state Kalman filter, it is necessary to derive the error-measurement equations. Starting with vision-aided position measurement,

$$\begin{aligned} \tilde{\boldsymbol{y}}_p &= \boldsymbol{y}_{\boldsymbol{p}} - \hat{\boldsymbol{p}} \\ &= \tilde{\boldsymbol{p}} + \boldsymbol{w}_p^s \end{aligned} \tag{5.20}$$

where $\hat{\boldsymbol{p}} \in \mathbb{R}^3$ is the estimated position, and $\boldsymbol{y}_{\boldsymbol{p}} := \boldsymbol{p}_t + \boldsymbol{w}_p^s \in \mathbb{R}^3$ is the measured position from (5.5), which is composed by the true position and the zero-mean Gaussian noise. Moreover, the error-measurement equations for the vector-based attitude aiding are on the form

$$\begin{aligned} \tilde{\boldsymbol{y}}_i &= (\boldsymbol{W}_i^s + \boldsymbol{w}_i^s) - (\hat{R}_s^n)^T \boldsymbol{V}_i^n \\ &= (R_s^n)^T \boldsymbol{V}_i^n + \boldsymbol{w}_i^s - (\hat{R}_s^n)^T \boldsymbol{V}_i^n, \end{aligned} \tag{5.21}$$

where $\hat{\boldsymbol{W}}_i^s = (\hat{R}_s^n)^T \boldsymbol{V}_i^n \in \mathbb{R}^3$ represent the estimated measurement vector, $\boldsymbol{W}_i^s \in \mathbb{R}^3$ denote the noise-free observed unit-vector, and $\boldsymbol{w}_i \in \mathbb{R}^3$ denote the Gaussian white measurement noise. Similarly to the quaternion estimation error, one can use the multiplicative rotational estimation error $\tilde{R} := R\hat{R}^T$. The measurement equation can now be further simplified,

$$\begin{aligned} \tilde{\boldsymbol{y}}_i &= (\tilde{R}_s^n \hat{R}_s^n)^T \boldsymbol{V}_i^n + \boldsymbol{w}_i^s - (\hat{R}_s^n)^T \boldsymbol{V}_i^n \\ &\approx (\hat{R}_s^n)^T (I_3 - [\tilde{\boldsymbol{\theta}}]_\times) \boldsymbol{V}_i^n - (\hat{R}_s^n)^T \boldsymbol{V}_i^n + \boldsymbol{w}_i^s \\ &= [(\hat{R}_s^n)^T \boldsymbol{V}_i^n]_\times \tilde{\boldsymbol{\theta}} + \boldsymbol{w}_i^s, \end{aligned} \tag{5.22}$$

where the skew-symmetric property (2.8) and $\tilde{R} \approx I_3 + [\tilde{\boldsymbol{\theta}}]_\times$ are applied. The approximation of (2.24) is based on the assumption of small perturbation angles, i.e., $[\tilde{\boldsymbol{\theta}}]_\times^2 \approx \boldsymbol{0}$. It is now possible to establish the nonlinear dynamics for the error-measurement equations

$$h(\tilde{\boldsymbol{x}}) = \begin{pmatrix} \tilde{\boldsymbol{p}} \\ [(\hat{R}_s^n)^T \boldsymbol{V}_g^n]_\times \tilde{\boldsymbol{\theta}} \\ [(\hat{R}_s^n)^T \boldsymbol{V}_{mag}^n]_\times \tilde{\boldsymbol{\theta}} \end{pmatrix}, \quad N(\tilde{\boldsymbol{x}}) = \begin{pmatrix} \boldsymbol{w}_p^s \\ \boldsymbol{w}_g^s \\ \boldsymbol{w}_{mag}^s \end{pmatrix}, \tag{5.23}$$

where $\boldsymbol{V}_g^n \in \mathbb{R}^3$ and $\boldsymbol{V}_{mag}^n \in \mathbb{R}^3$ are the reference unit-vectors for the gravitation and magnetic field, respectively.

5.2.1 Linearized Model

As previously mentioned, linear dynamics is obtained by continuously linearizing the nonlinear functions (5.18) and (5.23) around the current estimate, i.e., $\tilde{\boldsymbol{x}} = \mathbf{0}$. Thus, it is necessary to evaluate the Jacobian at each time step,

$$A_t = \left.\frac{\partial f(\tilde{\boldsymbol{x}}, \boldsymbol{u}, t)}{\partial \tilde{\boldsymbol{x}}}\right|_{\tilde{\boldsymbol{x}}=0, \boldsymbol{n}_x=0} \tag{5.24}$$

$$E_t = \left.\frac{\partial E(\tilde{\boldsymbol{x}}, t)}{\partial \boldsymbol{n}_x}\right|_{\tilde{\boldsymbol{x}}=0, \boldsymbol{n}_x=0} \tag{5.25}$$

$$C_t = \left.\frac{\partial h(\tilde{\boldsymbol{x}}, t)}{\partial \tilde{\boldsymbol{x}}}\right|_{\tilde{\boldsymbol{x}}=0} \tag{5.26}$$

$$N = \left.\frac{\partial N(\tilde{\boldsymbol{x}})}{\partial \boldsymbol{n}_y}\right|_{\tilde{\boldsymbol{x}}=0, \boldsymbol{n}_y=0} \tag{5.27}$$

The linearized model for the estimation and measurement error are then given by

$$\dot{\tilde{\boldsymbol{x}}} = \overbrace{\begin{bmatrix} 0_3 & I_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & -\hat{R}_s^n[\boldsymbol{a}_{imu}^s - \boldsymbol{b}_{acc}^s]_\times & -\hat{R}_s^n & 0_3 \\ 0_3 & 0_3 & -[\boldsymbol{\omega}_{imu}^s - \boldsymbol{b}_{gyro}^s]_\times & 0_3 & -I_3 \\ 0_3 & 0_3 & 0_3 & -\frac{1}{T_{acc}} I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & -\frac{1}{T_{gyro}} I_3 \end{bmatrix}}^{A_t} \tilde{\boldsymbol{x}} + \overbrace{\begin{bmatrix} 0_3 & 0_3 & 0_3 & 0_3 \\ -\hat{R}_s^n & 0_3 & 0_3 & 0_3 \\ 0_3 & -I_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & I_3 \end{bmatrix}}^{E_t} \boldsymbol{n}_x, \tag{5.28}$$

and

$$\tilde{\boldsymbol{y}} = \overbrace{\begin{bmatrix} I_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & [\hat{R}_n^s \boldsymbol{V}_g^n]_\times & 0_3 & 0_3 \\ 0_3 & 0_3 & [\hat{R}_n^s \boldsymbol{V}_{mag}^n]_\times & 0_3 & 0_3 \end{bmatrix}}^{C_t} \tilde{\boldsymbol{x}} + \overbrace{\begin{bmatrix} I_3 & 0_3 & 0_3 \\ 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & I_3 \end{bmatrix}}^{N} \boldsymbol{n}_y, \tag{5.29}$$

respectively. Note that R_s^n, which is implicit dependent on the state variable $\tilde{\boldsymbol{\theta}}$, is included in (5.18). In order to avoid a non-zero contribution in the Jacobian, a common and justified simplification is $R_s^n \approx \hat{R}_s^n$ [15]. A continuous-discrete version of the MEKF for inertial navigation is given in Algorithm 1. Recall the process and measurement noises, $\boldsymbol{n}_x \in \mathbb{R}^{12}$ and $\boldsymbol{n}_x \in \mathbb{R}^9$, which are zero-mean Gaussian noises with covariances Q and V, respectively. The symmetric covariance matrix of the estimation error P evolves between measurements as

$$\dot{P}_t = A_t P_t + P_t A_t^T + E_t Q E_t^T. \tag{5.30}$$

When a new aiding measurement is available, the state estimate and error covariance update according to the equations

$$K_k = P_{t_k} C_{t_k}^T (C_{t_k} P_{t_k} C_{t_k}^T + NVN^T)^{-1} \tag{5.31}$$

$$P_{t_k}^+ = P_{t_k} - K_k C_{t_k} P_{t_k} \tag{5.32}$$

$$\hat{\boldsymbol{x}}_{t_k}^+ = \hat{\boldsymbol{x}}_{t_k} \oplus K_k(\boldsymbol{y}_{t_k} - C_{t_k}\hat{\boldsymbol{x}}_{t_k}), \tag{5.33}$$

where t_k and K_k are the time instant of the new aiding measurement and the Kalman gain, respectively [3]. The symbol $\oplus$ indicates a generic composition (see Table 5.1). Furthermore, the MEKF leads to matrices A_t and C_t depending on the trajectory, as seen from (5.28) and (5.29). In other words, the estimation may diverge for some initial conditions, P_0 and $\boldsymbol{x}_0$ [2, 47].

Note the presence of a time constant for the accelerometer and gyroscope bias. In dead-reckoning, it is desirable to keep the bias estimation unchanged until new aiding measurements are available, as the estimate might deviate significantly from the true position [15].

As the MEKF is to be implemented in a computer-controlled system, numerical integration is necessary [41]. A typical integration method for the systems encountered in control theory is based on the finite-differences method for the computation of the derivative, i.e.,

$$\dot{\tilde{\boldsymbol{x}}} = \lim_{\delta t \to 0} \frac{\tilde{\boldsymbol{x}}(t+\delta t) - \boldsymbol{x}(t)}{\delta t} \approx \frac{\tilde{\boldsymbol{x}}_{n+1} - \tilde{\boldsymbol{x}}_n}{\Delta t}, \tag{5.34}$$

which is known as the Euler method. Using the linearized model (5.28) at beginning of the integration interval leads to

$$\tilde{\boldsymbol{x}}_{n+1} \approx \tilde{\boldsymbol{x}}_n + \dot{\tilde{\boldsymbol{x}}}(t_n)\Delta t = (I_{15} + A_{t_n}\Delta t)\tilde{\boldsymbol{x}}_n + E_{t_n}\Delta t \boldsymbol{n}_x. \tag{5.35}$$

Hence, the discrete-time error-state matrices become

$$A_d = I_{15} + A_{t_n}\Delta t \tag{5.36}$$

$$E_d = E_{t_n}\Delta t, \tag{5.37}$$

for a time step Δt. It is further shown in [15] that the discrete error-measurement matrix become

$$C_d = C_{t_n}. \tag{5.38}$$

Moreover, the corresponding discrete version of (5.30) is given by

$$P_{t+\Delta t} = A_d P_t A_d^T + E_d Q E_d^T. \tag{5.39}$$

In real life, the positioning aiding is sporadic, meaning there is no fixed measurement rate. In addition, the position measurement may disappear due to sensor failure or lost tracking of

Algorithm 1: Continuous-discrete MEKF for Inertial Navigation

```
Input: $\boldsymbol{a}^s_{imu}, \boldsymbol{\omega}^s_{imu}$ for all $t \geq 0$, and $\boldsymbol{y}_p(t_k), \boldsymbol{W}^s_g(t_k), \boldsymbol{W}^s_{mag}(t_k)$ with $k \in \mathbb{N}_{>0}$.
Output: $\hat{\boldsymbol{p}}(t), \hat{\boldsymbol{v}}(t), \hat{\boldsymbol{q}}(t), \hat{\boldsymbol{b}}_{acc}(t), \hat{\boldsymbol{b}}_{gyro}(t)$ for all $t \geq 0$.
Initialize: $[\hat{\boldsymbol{p}}^T(0)\ \hat{\boldsymbol{v}}^T(0)\ \hat{\boldsymbol{q}}^T(0)\ \hat{\boldsymbol{b}}^T_{acc}(0)\ \hat{\boldsymbol{b}}^T_{gyro}(0)]^T \leftarrow \boldsymbol{x}_0$
Initialize: $P \leftarrow P_0$
for $k \geq 1$ do
    while $t \in [t_{k-1}, t_k]$ do
        $\dot{\hat{\boldsymbol{p}}} = \hat{\boldsymbol{v}}$
        $\dot{\hat{\boldsymbol{v}}} = g + \hat{R}^n_s(\boldsymbol{a}^s_{imu} - \boldsymbol{b}^s_{acc})$
        $\dot{\hat{\boldsymbol{q}}} = \frac{1}{2}\hat{\boldsymbol{q}} \otimes (\boldsymbol{\omega}^s_{imu} - \boldsymbol{b}^s_{gyro})$
        $\dot{P}_t = A_t P_t + P_t A_t^T + E_t Q E_t^T$
    $K_k \leftarrow P_{t_k} C^T_{t_k}(C_{t_k} P_{t_k} C^T_{t_k} + V)^{-1}$
    $\boldsymbol{z} \leftarrow [(\boldsymbol{y}_{p,t_k} - \hat{\boldsymbol{p}}_{t_k})^T, (\boldsymbol{W}^s_{g,t_k} - (\hat{R}^n_s)^T \boldsymbol{V}^n_g)^T, (\boldsymbol{W}^s_{mag,t_k} - (\hat{R}^n_s)^T \boldsymbol{V}^n_{mag})^T]^T$
    obtain $\delta\tilde{\boldsymbol{p}}, \delta\tilde{\boldsymbol{v}}, \delta\tilde{\boldsymbol{\theta}}, \delta\tilde{\boldsymbol{b}}_{acc}, \delta\tilde{\boldsymbol{b}}_{gyro} \in \mathbb{R}^3$ from $[\delta\tilde{\boldsymbol{p}}^T, \delta\tilde{\boldsymbol{v}}^T, \delta\tilde{\boldsymbol{\theta}}^T, \delta\tilde{\boldsymbol{b}}^T_{acc}, \delta\tilde{\boldsymbol{b}}^T_{gyro}]^T \leftarrow K_k \boldsymbol{z}$
    obtain $\delta\tilde{\boldsymbol{q}}$ from (5.16)
    $\hat{\boldsymbol{p}}^+_{t_k} \leftarrow \hat{\boldsymbol{p}}_{t_k} + \delta\tilde{\boldsymbol{p}}$
    $\hat{\boldsymbol{v}}^+_{t_k} \leftarrow \hat{\boldsymbol{v}}_{t_k} + \delta\tilde{\boldsymbol{v}}$
    $\hat{\boldsymbol{q}}^+_{t_k} \leftarrow \hat{\boldsymbol{q}}_{t_k} \otimes \delta\tilde{\boldsymbol{q}}$
    $\hat{\boldsymbol{b}}^+_{acc,t_k} \leftarrow \hat{\boldsymbol{b}}_{acc} + \delta\tilde{\boldsymbol{b}}_{acc}$
    $\hat{\boldsymbol{b}}^+_{gyro,t_k} \leftarrow \hat{\boldsymbol{b}}_{gyro} + \delta\tilde{\boldsymbol{b}}_{gyro}$
    $P^+_{t_k} \leftarrow P_{t_k} - K_k C_{t_k} P_{t_k}$
```

the landmarks. Hence, it is necessary to account for attitude aiding alone, i.e.,

$$C_t = \begin{bmatrix} 0_3 & 0_3 & [\hat{R}^s_n \boldsymbol{V}^n_g]_\times & 0_3 & 0_3 \\ 0_3 & 0_3 & [\hat{R}^s_n \boldsymbol{V}^n_{mag}]_\times & 0_3 & 0_3 \end{bmatrix}. \tag{5.40}$$

Reducing the matrix C_t requires a corresponding reduction of the measurement noise covariance matrix V. This is achieved by pre and post multiplying a reduction matrix $H_{3:9} \in \mathbb{R}^{6\times 9}$,

$$R_d = H_{3:9} V H^T_{3:9}, \qquad H_{3:9} = \begin{bmatrix} 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & I_3 \end{bmatrix}. \tag{5.41}$$

On the other hand, the magnetometer is an onboard sensor independent of external features (disregarding external magnetic interference). Hence, it is reasonable to assume available magnetometer measurements whenever a new position measurement is received. A continuous-discrete version of the MEKF with numerical integration is given in Algorithm 2.

Algorithm 2: Continuous-discrete MEKF for Inertial Navigation with numerical integration for the continuous part.

```
Input: $\boldsymbol{a}^s_{imu,t_n}, \boldsymbol{\omega}^s_{imu,t_n}$ with $n \in \mathbb{N}_{>0}$, and $\boldsymbol{y}_p(t_p)$, $\boldsymbol{W}^s_g(t_k)$, $\boldsymbol{W}^s_{mag}(t_k)$ with
  $p, k \in \mathbb{R}_{>0}$.
Output: $\hat{\boldsymbol{p}}_{t_n}, \hat{\boldsymbol{v}}_{t_n}, \hat{\boldsymbol{q}}_{t_n}, \hat{\boldsymbol{b}}_{acc,t_n}, \hat{\boldsymbol{b}}_{gyro,t_n}$ with $n \in \mathbb{N}_{>0}$.
Initialize: $[\hat{\boldsymbol{p}}^T_{t_0}\ \hat{\boldsymbol{v}}^T_{t_0}\ \hat{\boldsymbol{q}}^T_{t_0}\ (\hat{\boldsymbol{b}}_{acc,t_0})^T\ (\hat{\boldsymbol{b}}_{gyro,t_0})^T]^T \leftarrow \boldsymbol{x}_0$
Initialize: $P \leftarrow P_0$
for $i \to N$ do
    $t_n \leftarrow i\Delta t$
    $A_d \leftarrow I_{15} + A_{t_n}\Delta t$
    $E_d \leftarrow E_{t_n}\Delta t$
    $Q_d \leftarrow Q$
    $\hat{R}^n_s \leftarrow R(\hat{\boldsymbol{q}}_{t_n})$,                                   // $R(\boldsymbol{q})$ given in (2.24)
    if new aiding measurement then
        if new position and attitude aiding then
            $C_d \leftarrow C_{t_n}$ from (5.29)
            $\boldsymbol{z} \leftarrow [(\boldsymbol{y}_{p,t_n} - \hat{\boldsymbol{p}}_{t_n})^T, (\boldsymbol{W}^s_{g,t_n} - (\hat{R}^n_s)^T\boldsymbol{V}^n_g)^T, (\boldsymbol{W}^s_{mag,t_n} - (\hat{R}^n_s)^T\boldsymbol{V}^n_{mag})^T]^T$
            $R_d \leftarrow V$
        else
            $C_d \leftarrow C_{t_n}$ from (5.40)
            $\boldsymbol{z} \leftarrow [(\boldsymbol{W}^s_{g,t_n} - (\hat{R}^n_s)^T\boldsymbol{V}^n_g)^T, (\boldsymbol{W}^s_{mag} - (\hat{R}^n_s)^T\boldsymbol{V}^n_{mag,t_n})^T]^T$
            $R_d \leftarrow H_{3:9}VH^T_{3:9}$
        $K_n \leftarrow P_{t_n}C^T_d(C_dP_{t_n}C^T_d + R_d)^{-1}$
        obtain $\delta\tilde{\boldsymbol{p}}, \delta\tilde{\boldsymbol{v}}, \delta\tilde{\boldsymbol{\theta}}, \delta\tilde{\boldsymbol{b}}_{acc}, \delta\tilde{\boldsymbol{b}}_{gyro} \in \mathbb{R}^3$ from $[\delta\tilde{\boldsymbol{p}}^T, \delta\tilde{\boldsymbol{v}}^T, \delta\tilde{\boldsymbol{\theta}}^T, \delta\tilde{\boldsymbol{b}}^T_{acc}, \delta\tilde{\boldsymbol{b}}^T_{gyro}]^T = K_n\boldsymbol{z}$
        obtain $\delta\tilde{\boldsymbol{q}}$ from (5.16)
        $\hat{\boldsymbol{p}}^+_{t_n} \leftarrow \hat{\boldsymbol{p}}_{t_n} + \delta\tilde{\boldsymbol{p}}$
        $\hat{\boldsymbol{v}}^+_{t_n} \leftarrow \hat{\boldsymbol{v}}_{t_n} + \delta\tilde{\boldsymbol{v}}$
        $\hat{\boldsymbol{q}}^+_{t_n} \leftarrow \hat{\boldsymbol{q}}_{t_n} \otimes \delta\tilde{\boldsymbol{q}}$
        $\hat{\boldsymbol{b}}^+_{acc,t_n} \leftarrow \hat{\boldsymbol{b}}_{acc} + \delta\tilde{\boldsymbol{b}}_{acc}$
        $\hat{\boldsymbol{b}}^+_{gyro,t_n} \leftarrow \hat{\boldsymbol{b}}_{gyro} + \delta\tilde{\boldsymbol{b}}_{gyro}$
        $P^+_{t_n} \leftarrow P_{t_n} - K_nC_dP_{t_n}$
    $\hat{\boldsymbol{p}}_{t_{n+1}} \leftarrow \hat{\boldsymbol{p}}_{t_n} + \hat{\boldsymbol{v}}_{t_k}\Delta t$
    $\hat{\boldsymbol{v}}_{t_{n+1}} \leftarrow \hat{\boldsymbol{v}}_{t_n} + \hat{R}^n_s(\boldsymbol{a}^s_{imu,t_n} - \hat{\boldsymbol{b}}_{acc,t_n}) + \boldsymbol{g}^n)\Delta t$
    $\hat{\boldsymbol{q}}_{t_{n+1}} \leftarrow \hat{\boldsymbol{q}}_{t_n} + T(\hat{\boldsymbol{q}}_{t_n})(\boldsymbol{\omega}^s_{imu,t_n} - \hat{\boldsymbol{b}}_{gyro})\Delta t$,        // $T(\boldsymbol{q})$ given in (2.25)
    $P_{t_{n+1}} \leftarrow A_dP_{t_n}A^T_d + E_dQ_dE^T_d$
```

Chapter 6

Implementation

This chapter presents the simulation and experimental setup, including the basic experimental hardware utilized.

6.1 Simulation Setup

Simulation studies are a powerful and vital tool for testing and validating robotic systems. Initial testing on the physical system can result in instability and cause damage to the system due to a lack of robustness in the initial design. Moreover, experimental testing of the implementations is expensive, time-consuming, and requires suitable testing facilities. As the simulated behavior is similar to the physical system, it uncovers most of the same initial bugs as the experimental study. In addition, it reduces the probability of damaging actuators and other components.

Unlike most simulation studies of marine crafts, the implementations of this thesis are primarily dependent on the vision sensor. In other words, it is not sufficient to use a simulation model without visualization to test the vision-aided navigation system. Thus, the Gazebo-based UUV simulator from [30] was preferable due to its simulation of vehicle motion and simultaneous visualization of the environment (see Figure 6.1).

The simulation model is based on the Sperre 30K ROV and derived in [4]. The vehicle description and the necessary launch files to simulate the ROV in the simulation environment were obtained from an open-source git repository[1]. A PID joy-stick controller from the same git repository was used as it was out of scope to cover the control design of the simulator. In addition, a feedback control system requires a robust observer design in order to obtain the error states accurately enough to avoid instabilities. Thus, a joy-stick controller with human-in-loop is a good first approach to test and validate the observer design.

[1]`https://github.com/uuvsimulator/rexrov2`

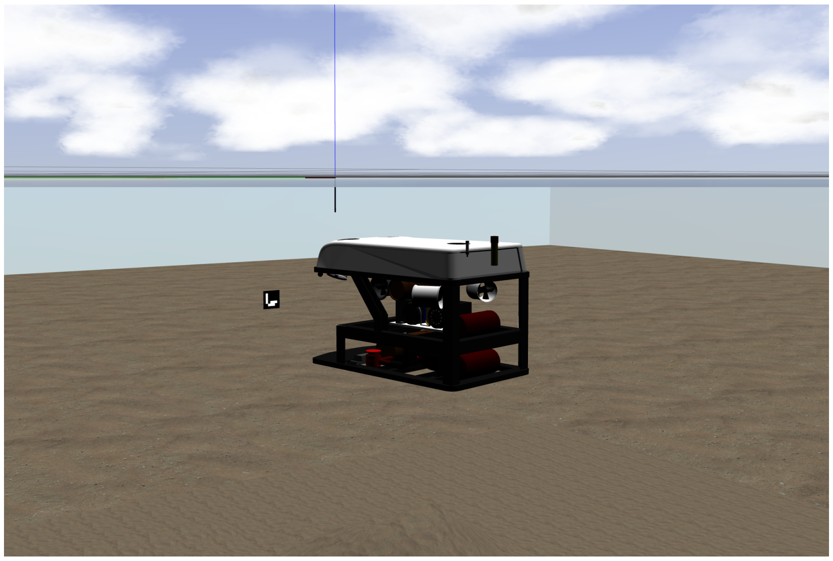


Figure 6.1: UUV simulator

The simulator comes with several Gazebo worlds implemented as YAML files. In order to add ArUco markers to the environment, the YAML file was modified using an open-source package[2].

In order to test the vision-aided positioning, a virtual camera was implemented on the ROV. The camera was placed with a lever arm $\boldsymbol{d}_{bc}^{b} = [1.15\ 0\ -0.4]^T$ from $\mathcal{F}_b$. As mentioned in Chapter 4, it is desirable to tilt the camera to capture more of what's below the vehicle. Hence, the camera was tilted $\theta = -0.6$ rad according to (4.3). In other words, the homogeneous transformation from F_b to F_c is

$$H_c^b = \begin{bmatrix} 0.0000 & 0.5646 & 0.8253 & 1.1500 \\ 1.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.8253 & 0.5646 & -0.4000 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \end{bmatrix}. \tag{6.1}$$

Note that the simulated IMU share coordinate frame with the Body-fixed frame, i.e., $\mathcal{F}_b = \mathcal{F}_s$. Moreover, an artificial ArUco marker was placed at $(x, y, z) = (0, 2, 2)$ pointing towards the west direction (see Figure 6.1). The resulting homogeneous transformation from $\mathcal{F}_n$ to $\mathcal{F}_m$ is then

$$H_m^n = \begin{bmatrix} 0.0000 & -1.0000 & 0.0000 & 0.0000 \\ -1.0000 & 0.0000 & 0.0000 & 2.0000 \\ 0.0000 & 0.0000 & -1.0000 & 2.0000 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 \end{bmatrix}. \tag{6.2}$$

[2]`https://github.com/joselusl/aruco_gazebo`

It should be noted that Gazebo does not accept any other standard than East-North-Up (ENU) as the inertial world frame, contrary to the common notation of underwater robotics, which uses the NED frame. Despite the inconvenient notation, the UUV simulator has extended its software to allow for the NED frame as the inertial world frame. However, some bugs were discovered while experimenting with the simulation tool. The bugs were primarily related to the simulated sensors, i.e., the accelerometer and the magnetometer. The major bug was the ROS plugin for the simulated inertial measurement unit, where the contribution from the gravity vector was faulty. The gravity was rotated to the body frame using the incorrect attitude representation, yielding unexpected acceleration measurements due to the gravity. The modified simulator and vision-aided inertial navigation system implementations may be found at the author's git repository[3].

6.2 Experimental Setup

The experimental study was carried out at the Marine Cybernetics Laboratory (MC-Lab) at the Norwegian University of Science and Technology (NTNU) in Trondheim (see Figure 6.2). The laboratory consists of a tank with dimensions L x B x D = 40m x 6.45m x 1.5m, including a towing carriage and wave-maker. Moreover, several Qualisys underwater motion capture cameras capture and analyze the actual pose of the underwater vehicle in 6 degrees of freedom. The key components of the motion capture system are the Oqus cameras and the Qualisys Track Manager (QTM) software [32].

6.2.1 Hardware

The experimental platform used to test the vision-aided inertial navigation system is an ROV delivered by BlueRobotics, more precisely, the BlueROV2 Heavy Configuration. The specifications of the BlueROV2 are given in Appendix A, while the essential hardware is given Figure 6.3. Onboard the drone, a Raspberry Pi 4 (RPi4) runs a Robotic Operating System, ROS2 foxy, to communicate and fuse the sensory information. A USB connects the camera (Sony IMX322) and the IMU (Bosch BNO055) to the Raspberry Pi, ensuring both power and communication. As mentioned in Chapter 3, the IMU data is published as a `sensor_msgs/Imu` message to the ROS network, while the camera image is streamed using Gstreamer. Moreover, the Adafruit PWM driver, PCA9685, converts the thruster instructions to Pulse-Width-Modulation (PWM) signals and sends them to the engine speed controllers (ESCs). The drone is equipped with eight T200 thrusters with a thruster map ($g(\text{PWM}) : \text{PWM} \rightarrow T$) given in Figure A.2. The true pose is obtained through the Qualisys

[3]`https://github.com/roarstavnes/ros-UUVsimulator`

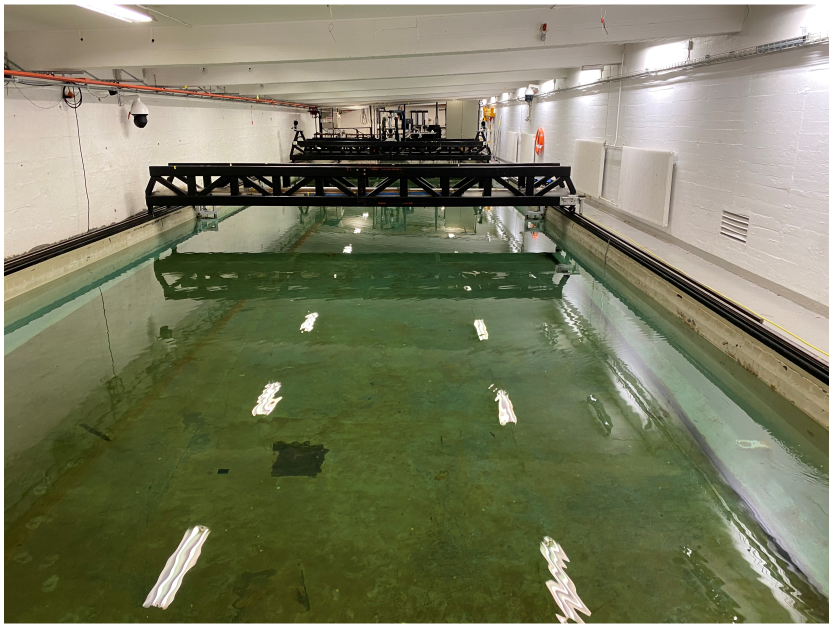


Figure 6.2: The Marine Cybernetics Laboratory at the Norwegian University of Science and Technology in Trondheim.

positioning system and published as a ROS `nav_msgs/Odometry` message using a computer dedicated to the Qualisys system at MC-lab.

Furthermore, an 18Ah Lithium-ion battery powers the ROV, yielding 14.8 volts. A distribution board powers the onboard components, where the engine motor controllers are powered directly from the battery with 14.8V. On the other hand, the RPi4 is powered through a transformer supplying the computer with 5V. For additional reading, a similar hardware platform was used in [22] for the derivation of a camera-assisted dynamic positioning system for ROVs.

A topside computer is connected to the BlueROV2 module through an ethernet umbilical, allowing for communication with the RPi4. For this experiment, the vision-inertial navigation system (VINS) runs on the topside computer due to the lack of computational power on the RPi4. However, the Gstreamer implementation provided a low latency video stream resulting in a negligible effect of delayed vision-based position aiding. The VINS implementations can be found at the author's git repository[4].

6.2.2 Control

Control design was not a part of the objectives in this thesis. However, a controller is necessary to test the navigation system. Furthermore, it is desirable to implement a human-in-the-loop controller rather than a feedback controller because feedback requires accurate state

[4]`https://github.com/roarstavnes/ros2-vision-aided-navigation`

estimation to avoid instabilities. Hence, a simple control allocation that maps the joystick input to thruster control input was derived. Figure 6.4 shows the controller logic, where the axis (blue arrows) are equivalent to producing a force in the specified degree of freedom. Moreover, the buttons on top create a fixed moment in yaw, where the right and left buttons are positive and negative moments, respectively. The ROV is self-stabilizable in roll and pitch; thus, the controller disregarded these two degrees to reduce the number of controllable states. In order to achieve safe operation and testing, one enable and disable button was assigned.

Let $R_x, R_y \in [-1, 1]$ denote the displacement of the right stick in right and up direction,

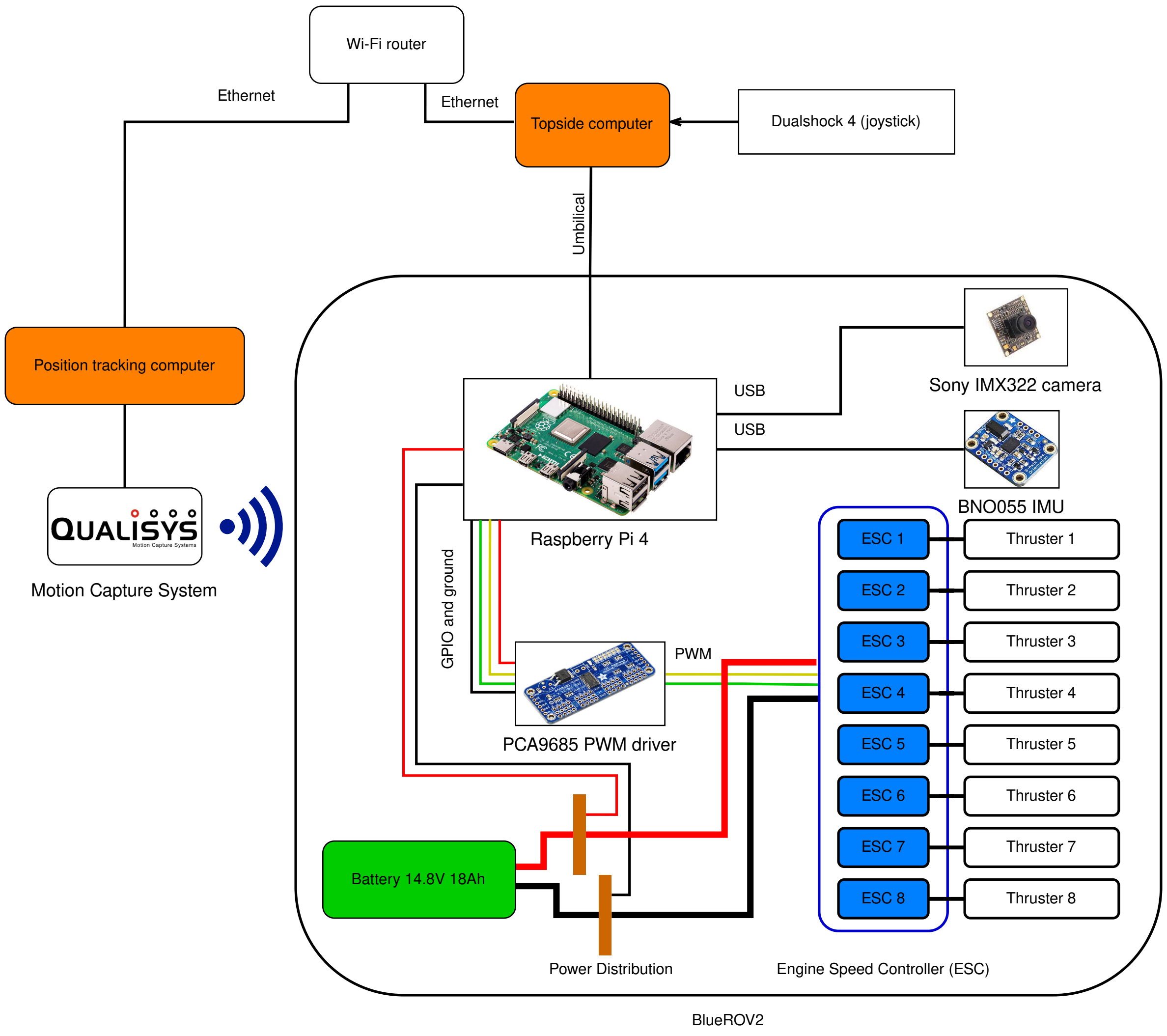


Figure 6.3: Illustration of the experimental hardware setup. This drawing should, by no means, be interpreted as a wiring/circuit diagram. Transformers are disregarded due to readability.

respectively. Similarly, let $L_y, L_x \in [-1, 1]$ represent the displacement of the left stick. Moreover, the buttons have two possible values, 1 for the pressed and 0 for the unpressed button. It is now possible to generate a map from the axes and buttons to desired body-force,

$$\boldsymbol{\tau} = \begin{bmatrix} K_{p,F}R_y \\ K_{p,F}R_x \\ K_{p,F}L_y \\ 0 \\ 0 \\ K_{p,M}(-L1 + R1) \end{bmatrix}, \tag{6.3}$$

where $K_{p,F}$ is the body-force gain, $K_{p,M}$ is the yaw gain, and $L1$ and $R1$ is the upper left and right button value, respectively. It is now straightforward to find the thruster control inputs using the thrust configuration matrix $T \in \mathbb{R}^{6\times 8}$, given in (A.2). The map is given by

$$\boldsymbol{\tau} = TK\boldsymbol{u}, \tag{6.4}$$

where $K \in \mathbb{R}^{8\times 8}$ and $u \in \mathbb{R}^8$ are a diagonal force matrix and the control input [15]. The inverse map, from body-force to control input, is found by taking the pseudo-inverse, i.e.,

$$\boldsymbol{u} = K^{-1}T^T(TT^T)^{-1}. \tag{6.5}$$

However, this is not sufficient in order to set the thruster instructions on the Raspberry Pi. The engine speed controller (ESC) takes PWM signals as input and regulates the thrusters to produce the desired thrust. Hence, it is necessary to find the PWM signal that corresponds to

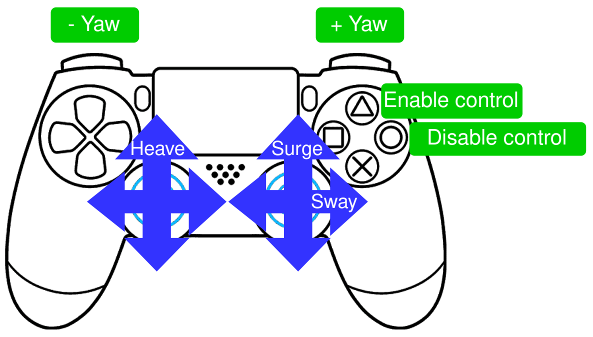


Figure 6.4: Logic of the Dualshock 4 controller. The blue arrow represents axes and the green boxes represents buttons. The plus sign denote positive moment, while the minus sign denote the opposite.

the desired control input calculated in (6.5). Figure A.2 shows the PWM-thrust map for the T200 thruster given by BlueRobotics. The RPi4 calculates the desired PWM signals using the inverse map, i.e.,

$$g(T_d) : T_d \to \text{PWM}, \tag{6.6}$$

where T_d represents the desired thrust. The signals are then sent to the thrusters through the Adafruit PWM driver.

6.2.3 Operating Condition

The experiment was performed without any external disturbances, such as wave and current loads. A yellow box was used to represent a subsea unit. Moreover, three rectangular ArUco markers with sides of 30 cm were attached as features for the vision-aided positioning (see Figure 6.5a). For convenience, a coordinate frame defining the position and orientation of the marker plane at the upper right corner was defined. This simplifies the process of determining the pose of the markers relative to $\mathcal{F}_n$. The experiment was initially performed in pitch-black conditions, using ROV LED as the light source (see Figure 6.5b). Consequently, the light source did interfere with the Qualisys positioning system resulting in occasional loss of motion tracking. Due to this fact, the experiment had to be conducted using external light sources to avoid the interaction between the Qualisys sytem.

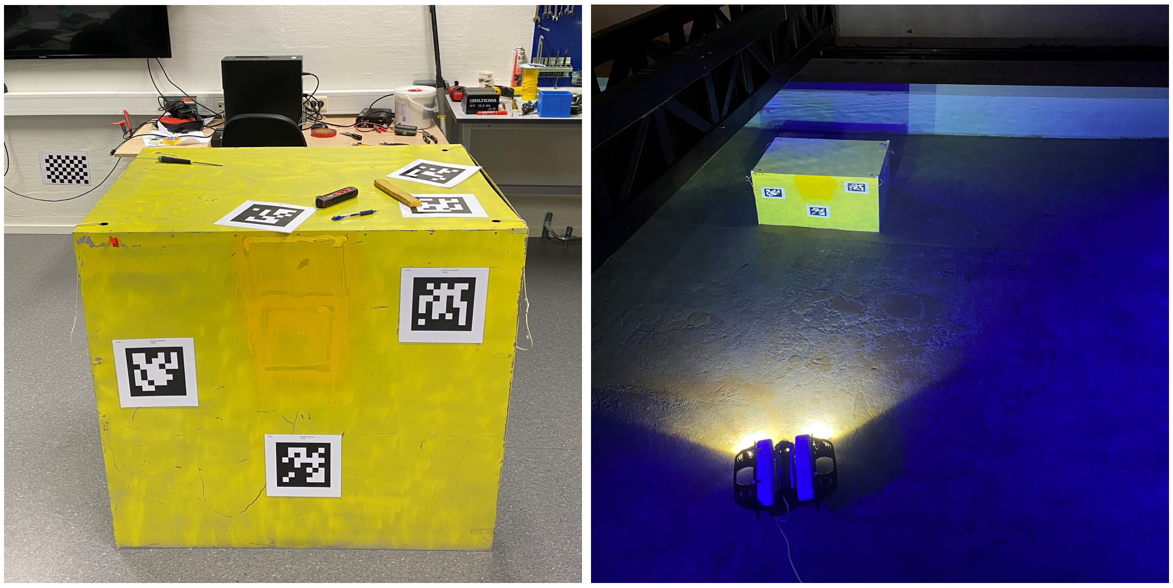

(a) Box with ArUco markers. (b) BlueROV2 operating in the MC-Lab.

Figure 6.5: The experimental setup.

Chapter 7

Results

This chapter presents the simulation results in addition to the experimental results from the Marine Cybernetics Laboratory (MC Lab) at the Norwegian University of Science and Technology (NTNU) in Trondheim. Note that the Euler angle representation is used exclusively to parameterize the attitude. This is because the Euler angle representation is intuitive and easy to interpret, simplifying presentation and debugging. Video of the simulation[1] and experiment[2] may be found by following the URL in the footnote or scanning the QR codes in Figure 7.1.

(a) Video of the simulation.

(b) Video of the experiment.

Figure 7.1: Video of the simulation and the experiment.

[1] `https://youtu.be/VIgVS2FLsy0`
[2] `https://youtu.be/xIjsFcuEM0c`

7.1 Simulation

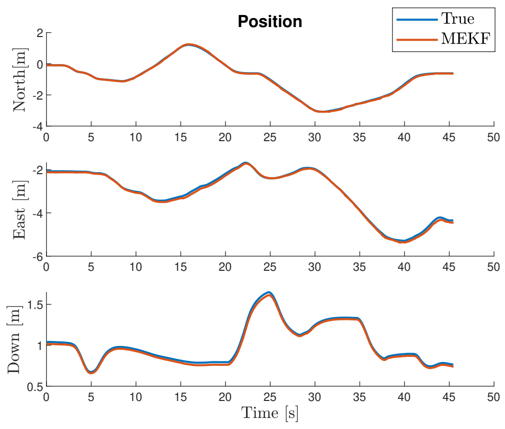
This section presents the result of the simulation study performed using the Gazebo-based UUV simulator [30]. The simulation setup and vehicle description are closer described in Chapter 6.

The simulation addresses the navigation system derived in Chapter 5, namely the multiplicative extended Kalman filter (MEKF). The MEKF is fed by the simulated magnetic field, linear body accelerations, angular body velocities, and vision-based position measurements and computes the motion estimate as output. The motion estimate includes position, orientation, and velocity expressed in $\mathcal{F}_n$, in addition to the accelerometer and gyroscope biases expressed in $\mathcal{F}_s$. The simulation was performed in the same operating conditions as described in Chapter 6, and a joystick controller was used to change the setpoints during simulation. Similar to the first simulation study in Chapter 4, the vehicle motion was primarily in the surge, sway, heave, and yaw direction, with minor changes in roll and pitch.

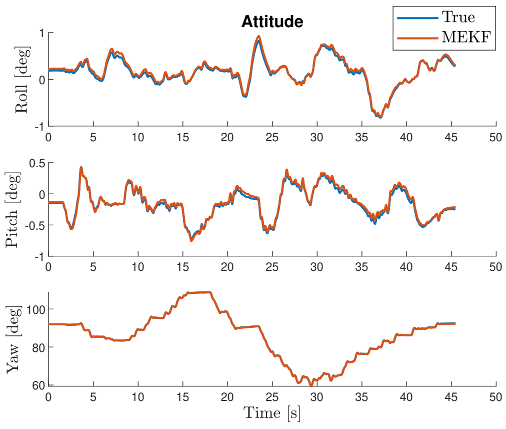
The simulated position and attitude are plotted against the estimated pose in Figure 7.2. Furthermore, the error between the simulated and estimated position is plotted in Figure 7.3 to get a clear overview of the order of magnitude. The corresponding mean, variance, and standard deviation of the error shown in Table 7.1 are used as performance measures of the navigation system. The same error data is represented in the histogram in Figure 7.4 to detect skewed data points and show the error distribution.

Table 7.1: Statistics of the MEKF in the simulated condition. The data is based on the error between the true and MEKF estimated value. Mean, Var, and Std denote the mean, variance, and standard deviation, respectively.

	North $\tilde{x}$	East $\tilde{y}$	Down $\tilde{z}$	Position $\tilde{\boldsymbol{p}}$	Roll $\tilde{\phi}$	Pitch $\tilde{\theta}$	Yaw $\tilde{\psi}$
Mean	0.0174 m	0.0516 m	0.0237 m	0.0690 m	-0.0351 deg	-0.0211 deg	0.0409 deg
Var	0.0007 m^2	0.0016 m^2	0.0001 m^2	0.0011 m^2	0.0007 deg^2	0.0006 deg^2	0.0115 deg^2
Std	0.0267 m	0.0399 m	0.0073 m	0.0335 m	0.0271 deg	0.0239 deg	0.1072 deg

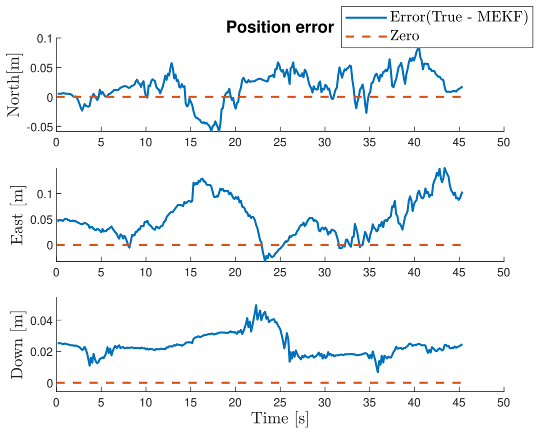


(a) Position estimate.

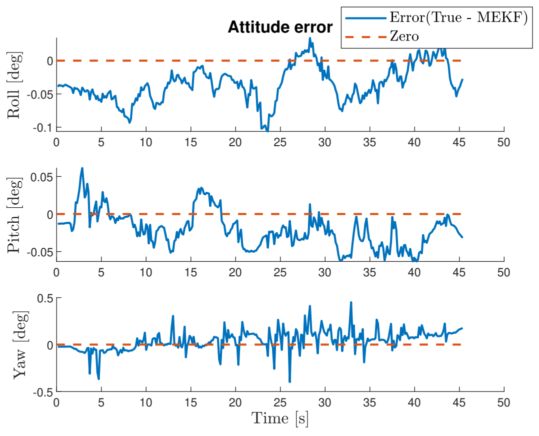


(b) Attitude estimate.

Figure 7.2: MEKF pose estimation in the simulated condition. The blue and orange line represents the true and MEKF estimated value, respectively. Each plot represents one of the six degrees of freedom where the attitude is parameterized using Euler angle representation.



(a) Position error.



(b) Attitude error.

Figure 7.3: MEKF estimation error in simulated condition. The blue represents the error between true and MEKF estimated value, while the dashed orange line represents the zero line.

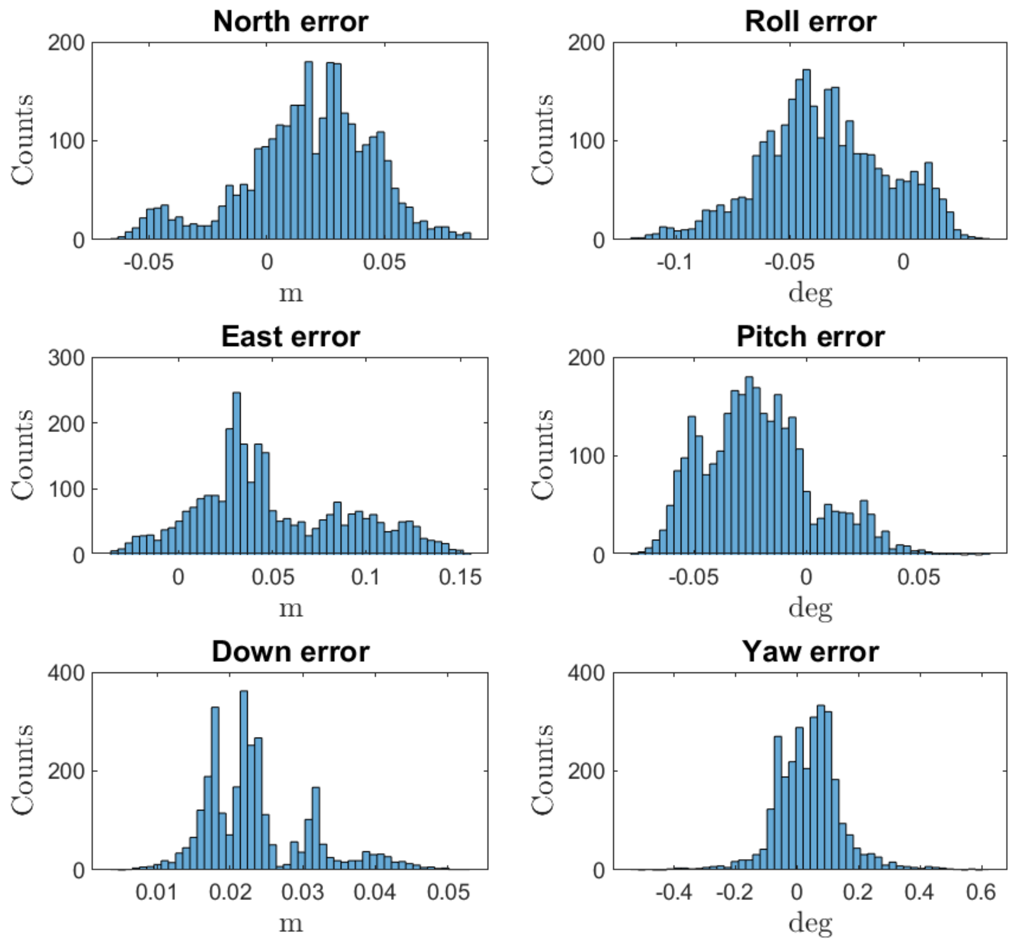
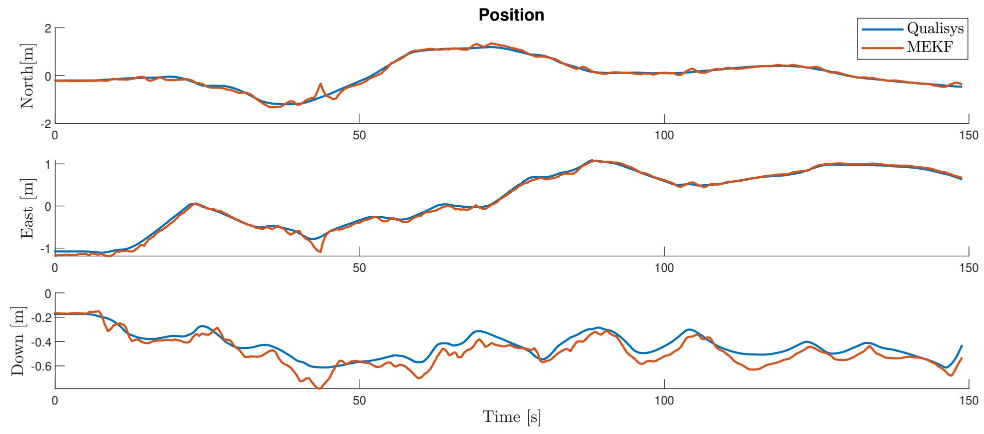


Figure 7.4: Histogram showing the distribution of error between the actual and the estimated vehicle pose in the simulated condition. Each histogram represents one of the six degrees of freedom.

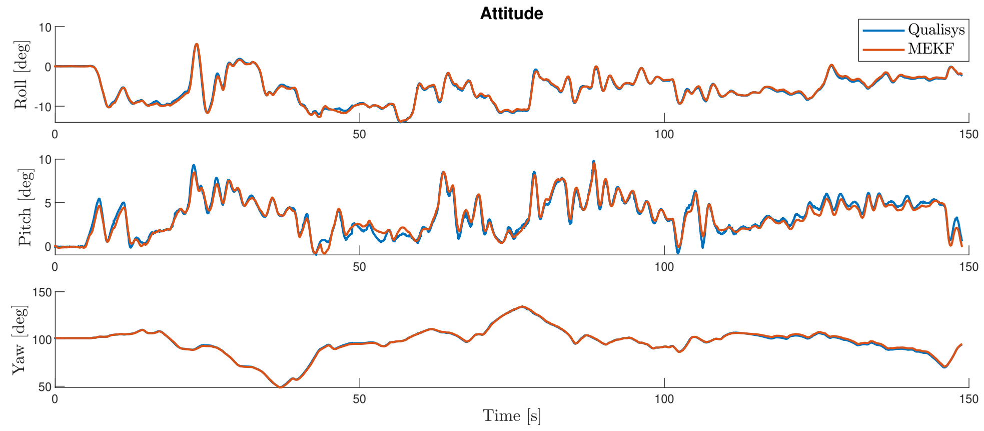
7.2 Experiment

Similar to the simulation, the experiment addresses the MEKF derived in Chapter 5. The inputs are two vector observations for attitude aiding, vision-based position measurement, linear body accelerations, and angular body velocities. However, the cheap MEMS-based magnetometer suffered from significant uncertainties. Consequently, a noisy heading reference vector was created using the Qualisys motion tracking system. The test facility and the experimental condition are described in Chapter 6.

The vehicle's actual position and attitude are plotted against the estimated pose in Figure 7.5. Equally to the performance measures in the simulation, the mean, variance, and



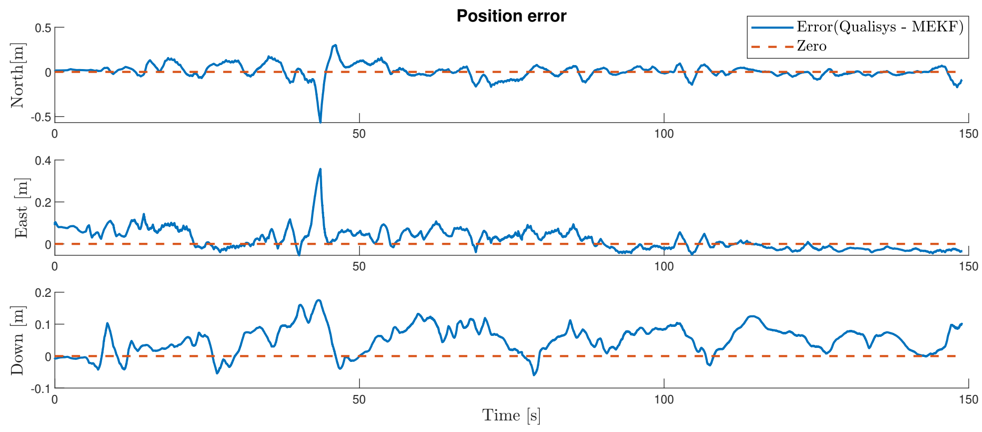
(a) Position estimate.



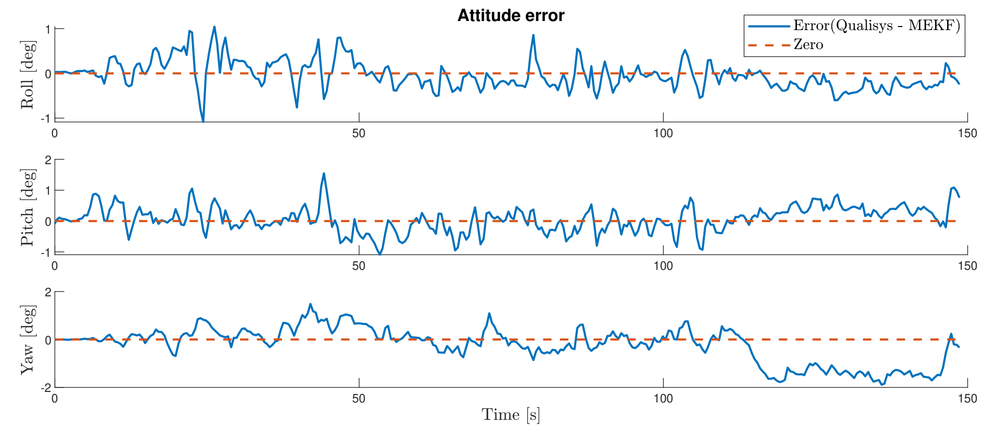
(b) Attitude estimate.

Figure 7.5: MEKF pose estimation in the experimental condition. The blue and orange line represents the true and MEKF estimated value, respectively. Each plot represents one of the six degrees of freedom where the attitude is parameterized using Euler angle representation.

standard deviation of the error between the true and estimated pose were calculated in Table 7.2. Furthermore, Figure 7.6 plots the error and the zero line, which is equivalent to perfect estimation, to easily visualize the order of magnitude of the error. The same data is represented in the histogram in Figure 7.7 which is helpful to observe any abnormal behavior in the distribution of error.



(a) Position error.



(b) Attitude error.

Figure 7.6: MEKF estimation error in the experimental condition. The blue represents the error between true and MEKF estimated value, while the dashed orange line represents the zero line.

Table 7.2: Statistics of the MEKF in the experimental condition. The data is based on the error between the true and MEKF estimated value. Mean, Var, and Std denote the mean, variance, and standard deviation, respectively.

	North $\tilde{x}$	East $\tilde{y}$	Down $\tilde{z}$	Position $\tilde{\boldsymbol{p}}$	Roll $\tilde{\phi}$	Pitch $\tilde{\theta}$	Yaw $\tilde{\psi}$
Mean	0.0058 m	0.0219 m	0.0514 m	0.1002 m	-0.0433 deg	0.0632 deg	-0.2228 deg
Var	0.0057 m^2	0.0025 m^2	0.0019 m^2	0.0031 m^2	0.1002 deg^2	0.1686 deg^2	0.5304 deg^2
Std	0.0756 m	0.0495 m	0.0433 m	0.0561 m	0.3166 deg	0.4107 deg	0.7283 deg

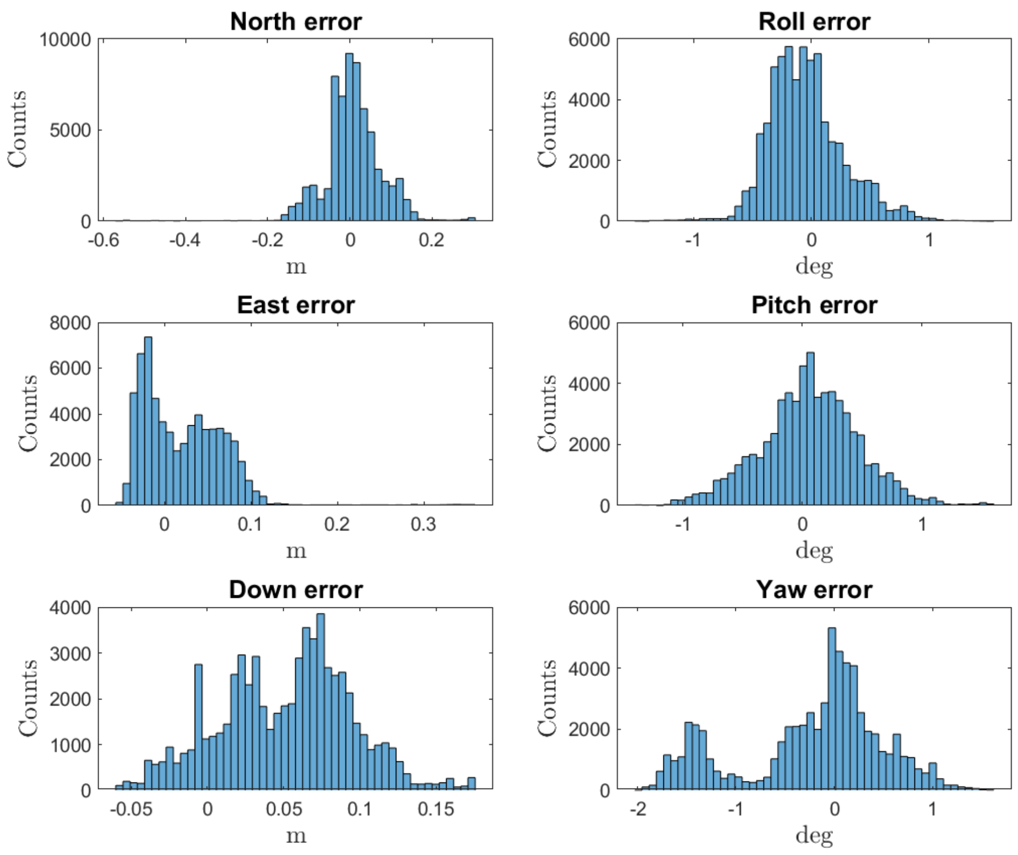


Figure 7.7: Histogram showing the distribution of error between the actual and the estimated vehicle pose in the experimental condition. Each histogram represents one of the six degrees of freedom.

Chapter 8

Discussion

The initial attempt, shown in Figures 4.4 and 4.5, was performed in the early phase of the project to determine the applicability of vision-based pose measurement. It involved computing the position and attitude based on the current observations directly through the forward kinematic equation (4.1). However, it turned out to be an inaccurate method for obtaining the pose. Figure 4.4b shows the significant noise in the measured rotation, primarily caused by the fact that each marker adds only four correspondences (each corner) to estimate the relative rotations. In addition, the small spacing between the corners resulting in significant deviation in the reprojection process. This problem is argued in some literature that addresses pose estimation by solving the Perspective-n-Point (PnP) problem, claiming a clear correlation between the number of correspondences and the rotation error. Nonetheless, as alluded in [26], less than ten correspondences result in considerably poor pose estimation for most PnP algorithms.

Furthermore, using the raw rotation measurement in computing the vehicle's global position, as shown in Figure 4.5a, suffers large deviations because of the direct relation to the measured rotation matrix in (4.1). Moreover, the raw measurements are the only information used for computing the pose; thus, it carries no memory of the previous states. In other words, the estimation does not account for the inertia of the physical system, i.e., the force needed to move the body from one pose to the next in one moment.

Instead of computing the vehicle's position with the raw pose measurements, the navigation system takes the measure translation as input. Moreover, the system computes the position based on the estimated rotation matrix according to (5.5). In this way, it accounts for the system's inertia through the rotation matrix estimated by the inertial navigation system. Furthermore, the imperfect rotation measurement is replaced by a magnetometer measurement. The Figures 7.2 and 7.5 show the advantage of replacing the rotation estimate and accounting for the inertia, resulting in the estimate following the actual pose closely.

According to Figures 7.3 and 7.6 and the error statistics in Tables 7.1 and 7.2, the simulation outperforms the experiment on all statistical performance criteria. However, this is to be expected as the experimental conditions contain complexities which simulation does not. For example, factors such as light and reflection complicate the marker pose estimation. Moreover, the simulated vessel dynamics and sensor models are simplified, disregarding some complexities.

Another factor influencing the performance of the experimental setup is the large motions in the roll and pitch. The BlueROV2 has a highly controllable design, meaning it should be accessible to induce large movements of all degrees of freedom. On the other hand, the simulated vehicle is a work-class ROV, which is highly self-stabilizable in the roll and pitch. As a result, the simulated vehicle and its camera hold a steady motion compared to the large movements of the BlueROV2. Moreover, the open-source controller design for the simulator is more robust and stable than the joystick controller implemented in Chapter 6. As the large vehicle motions induce both large accelerations and disturb the imaging, it presumable cause reduced performance. However, the large motions are a critical feature in evaluating the system; hence, the controller design was kept unchanged.

The histograms in Figures 7.4 and 7.7 shows that the estimation error in all 6 DOFs is similar to a normal distribution centered about zero. However, the experimental result has several outliers in the north and east error. Figure 7.6 shows the outliers between the interval of 40 to 50 seconds. Both spikes appear at the same time instant without significant errors in down, roll, pitch, and yaw. The same happens in the video from the experiment (see Figure 7.1b), where it loses track of the markers for three frames. As the marker detection node runs with a rate of 2 frames per second, the system is left unaided for more than 1.5 second. Simultaneously, the vehicle is exposed to large motions causing the estimate to drift off.

Comparing the result shown in Tables 7.1 and 7.2 with the existing navigation systems, one can see a clear advantage of the vision-aided navigation system (VINS). The acoustic USBL system in Table 1.1 has a position uncertainty of 0.12 m for an AUV depth of 50 m. On the other hand, the VINS solution had a position uncertainty of 0.0335 m and 0.0561 m for the simulation and experiment, respectively. It should be noted that the result in Table 1.1 has an attitude uncertainty of 0.01° in roll and pitch and 0.1° in the heading. However, it is not unlikely to have some misalignments of the transponder due to installation or environmental impacts. Hence, the VINS yield a significantly improved accuracy of the pose estimate when the vehicle is close to the markers.

Compared to existing SLAM methods, the feature positions do not contain uncertainty. This is because the actual marker position is known, in contrast to SLAM, where the feature position depends on the vehicle's position accuracy at the instant of observation. Consequently, the mean estimation error is approximately zero, as shown in Figures 7.3 and 7.5 and

Tables 7.1 and 7.2. Moreover, the ArUco markers possess a design making the detection and classification robust and fast, as seen in the simulation and experimental video (see Figure 7.1). In other words, the use of ArUco markers as features for vision-based position aiding eliminates one of the biggest challenges in SLAM navigation; misinterpretation of features.

Chapter 9

Conclusions and Further Work

9.1 Conclusions

Through the work of this thesis, the application of computer vision is found suitable and advantageous for increased underwater navigation accuracy. The computer vision application was designed for accurate navigation close to subsea intervention and maintenance locations, using ArUco markers as environmental features for localization.

The thesis presents a navigation system design based on the multiplicative extended Kalman filter, fusing the vision-based position estimation, attitude estimation, and inertial navigation. The principal contributions of this work are related to the extension upon an existing open-source simulation software capable of testing computer vision implementations merged with state estimation and control, with potential for further development for simulation of closed-loop intervention operations. Furthermore, it contributes to validating and testing the proposed navigation system using a simulated and experimental environment.

Seen in the light of the first research questions, the effect of the underwater environment on vision sensors was manageable. The dome-formed camera housing compensated for most of the air-glass-water refractions, and the sight was unproblematic for the experimental case. However, as the experimental results exclude field testing, the work can not conclude the environmental influence in the general case of subsea operations. Furthermore, distance and orientation between the vehicle and the observed objects were obtained using a single camera and ArUco markers, containing enough correspondences to estimate both position and orientation. The ArUco markers provided a robust detection and an easily distinguishable design. However, the small spacing between the correspondences resulted in a poor vision-based attitude estimate. Hence, the attitude estimate was replaced in favor of a magnetometer for attitude determination. Nevertheless, the markers provided a precise distance measurement for position aiding in the navigation system.

Compared to the existing underwater positioning techniques and SLAM methods, the proposed approach is a strong competitor with its precise pose estimation. Acoustic methods have higher position uncertainties than the VINS solution when the vehicle is in the vicinity of the markers. Furthermore, ArUco markers eliminate one of the biggest challenge in SLAM navigation, namely the misinterpretation of features. Hence, the proposed VINS may be a step towards autonomous subsea operations.

9.2 Further Work

The transition towards fully autonomous operations subsea is still quite a step away from today's solutions. However, the author hopes that the vision-aided inertial navigation system derived in this thesis would be relevant and applied further in the transition toward autonomous operations. Furthermore, the project work has inspired ideas and creative thinking beyond the presented material, among other things:

- Further development of the simulation software for simulation of intervention operations merged with guidance, navigation, and control.
- Improve the VINS attitude estimation by trigonometry when three or more ArUco markers with substantial distance are within sight.
- Extend the proposed VINS design to be suitable for the transition phase using, e.g., SLAM or acoustic transponders and modems techniques.

Bibliography

[1] G. Antonelli. *Underwater Robots, Third Edition.* Springer, 2014.

[2] A. Barrau and S. Bonnabel. “The invariant extended Kalman filter as a stable observer”. In: *IEEE Transactions on Automatic Control* 62.4 (2016), pp. 1797–1812.

[3] R. W. Beard and T. W. McLain. *Small unmanned aircraft: Theory and practice.* Princeton university press, 2012.

[4] V. Berg. “Development and Commissioning of a DP system for ROV SF 30k”. MA thesis. Norwegian University of Science and Technology, 2012.

[5] BlueRobotics. *BlueRobotics.* URL: `https://bluerobotics.com/`. (accessed: 05.06.2021).

[6] G. Bradski and A. Kaehler. *Learning OpenCV.* O’Reilly Media, Inc, 2008.

[7] D. C. Brown. “Close-range camera calibration”. In: *Photogrammetric Engineering* 37.8 (1971), pp. 855–866.

[8] D. C. Brown. “Decentring Distortion of Lenses”. In: *Photometric Engineering and Remote Sensing* 32 (1966), pp. 444–462.

[9] J. Canny. “A computational approach to edge detection”. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1986), pp. 679–698.

[10] M. De Agostino, A. M. Manzino, and M. Piras. “Performances comparison of different MEMS-based IMUs”. In: *IEEE/ION Position, Location and Navigation Symposium.* IEEE. 2010, pp. 187–201.

[11] D. H. Douglas and T. K. Peucker. “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature”. In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122.

[12] M. A. Engebretsen, K. S. Gjerden, Ø. B. Utbjoe, and A. Våge. “Autonomous Navigation, Mapping, and Exploration for Underwater Robots”. MA thesis. Norwegian University of Science and Technology, 2019.

[13] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. "On-Manifold Preintegration for Real-Time Visual–Inertial Odometry". In: *IEEE Transactions on Robotics* 33.1 (2016), pp. 1–21.

[14] T. I. Fossen. *Guidance and Control of Ocean Vehicles.* Wiley, Chichester, 1994.

[15] T. I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control (Draft Manuscript).* John Wiley & Sons, Ltd, 2021.

[16] P. J. From, J. T. Gravdahl, and K. Y. Pettersen. *Vehicle-Manipulator System: Modeling for Simulation, Analysis, and Control.* Springer-Verlag London, 2014.

[17] S. Garrido-Jurado and R. Muñoz-Salinas. *ArUco Library.* URL: `https://sourceforge.net/projects/aruco/`. (accessed: 28.04.2021).

[18] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez. "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47.6 (2014), pp. 2280–2292.

[19] B. C. Hall. *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction.* Springer, 2015.

[20] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* 2nd ed. Cambridge University Press, 2004.

[21] Ø. Hegrenæs, T. O. Sæbø, P. E. Hagen, and B. Jalving. "Horizontal mapping accuracy in hydrographic AUV surveys". In: *2010 IEEE/OES Autonomous Underwater Vehicles.* IEEE. 2010, pp. 1–13.

[22] A. V. Henriksen. "Camera-assisted Dynamic Positioning of ROVs". MA thesis. Norwegian University of Science and Technology, 2016.

[23] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering* 82.1 (1960), pp. 35–45.

[24] K. G. Kebkal and A. I. Mashoshin. "AUV acoustic positioning methods". In: *Gyroscopy and navigation* 8.1 (2017), pp. 80–89.

[25] E. J. Lefferts, F. L. Markley, and M. D. Shuster. "Kalman filtering for spacecraft attitude estimation". In: *Journal of Guidance, Control, and Dynamics* 5.5 (1982), pp. 417–429.

[26] V. Lepetit, F. Moreno-Noguer, and P. Fua. "Epnp: An accurate o(n) solution to the pnp problem". In: *International journal of computer vision* 81.2 (2009), p. 155.

[27] P. Liljebäck, K. Y. Pettersen, Ø. Stavdahl, and J. T. Gravdahl. *Snake robots: modelling, mechatronics, and control.* Springer Science & Business Media, 2012.

[28] T. Łuczyński, M. Pfingsthorn, and A. Birk. "The pinax-model for accurate and efficient refraction correction of underwater cameras in flat-pane housings". In: *Ocean Engineering* 133 (2017), pp. 9–22.

[29] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003.

[30] M. M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach. "UUV Simulator: A Gazebo-based package for underwater intervention and multi-robot simulation". In: *OCEANS 2016 MTS/IEEE Monterey*. 2016, pp. 1–8.

[31] D. W. Marquardt. "An algorithm for least-squares estimation of nonlinear parameters". In: *Journal of the society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441.

[32] NTNU. *Marine cybernetics teaching laboratory*. URL: `https://www.ntnu.edu/imt/lab/cybernetics`. (accessed: 28.05.2021).

[33] OpenCV. *Detection of ArUco Markers*. URL: `https://docs.opencv.org/master/d5/dae/tutorial_aruco_detection.html`. (accessed: 28.04.2021).

[34] N. Otsu. "A Threshold Selection Method from Gray-Level Histograms". In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), pp. 62–66.

[35] L. Paull, S. Saeedi, M. Seto, and H. Li. "AUV navigation and localization: A review". In: *IEEE Journal of oceanic engineering* 39.1 (2013), pp. 131–149.

[36] N. El-Sheimy, H. Hou, and X. Niu. "Analysis and modeling of inertial sensors using Allan variance". In: *IEEE Transactions on instrumentation and measurement* 57.1 (2007), pp. 140–149.

[37] M. D. Shuster and S. D. Oh. "Three-axis attitude determination from vector observations". In: *Journal of guidance and Control* 4.1 (1981), pp. 70–77.

[38] SNAME. "Nomenclature for treating the motion of a submerged body through a fluid". In: *The Society of Naval Architects and Marine Engineers, Technical and Research Bulletin* 1-5 (1950).

[39] SNL. *Nordsjødykkerne*. URL: `https://snl.no/nordsj%C3%B8dykkerne`. (accessed: 15.12.2020).

[40] J. Sola. "Quaternion kinematics for the error-state Kalman filter". In: *arXiv preprint arXiv:1711.02508* (2017).

[41] A. J. Sørensen. *Marine Cybernetics, Towards Autonomous Marine Operations and Systems, Lecture Notes*. Norwegian University of Science and Technology, 2018.

[42] M. W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control.* John Wiley & Sons, Inc, 2006.

[43] S. Suzuki and K. Abe. "Topological structural analysis of digitized binary images by border following". In: *Computer vision, graphics, and image processing* 30.1 (1985), pp. 32–46.

[44] J. Sverdrup-Thygeson, E. Kelasidi, K. Y. Pettersen, and J. T. Gravdahl. "The underwater swimming manipulator—A bioinspired solution for subsea operations". In: *IEEE Journal of Oceanic Engineering* 43.2 (2017), pp. 402–417.

[45] VISTA. *VISTA centre for development of collaborative robotic systems.* URL: `http://www.vista.no/nyheter/vis.html?tid=76227`. (accessed: 02.05.2021).

[46] G. Wahba. "A least squares estimate of satellite attitude". In: *SIAM review* 7.3 (1965), pp. 409–409.

[47] M. Wang and A. Tayebi. "Observers Design for Inertial Navigation Systems: A Brief Tutorial". In: *2020 59th IEEE Conference on Decision and Control (CDC).* IEEE. 2020, pp. 1320–1327.

[48] Z. Zhang. "A flexible new technique for camera calibration". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.11 (2000), pp. 1330–1334.

Appendix A

Specification BlueROV2

The specification of the ROV used as experimental platform is shown in Figure A.1. Moreover, the position and orientation of the thrusters are given in Table A.1, and the thruster-map from PWM signals to thrust are shown in Figure A.2.

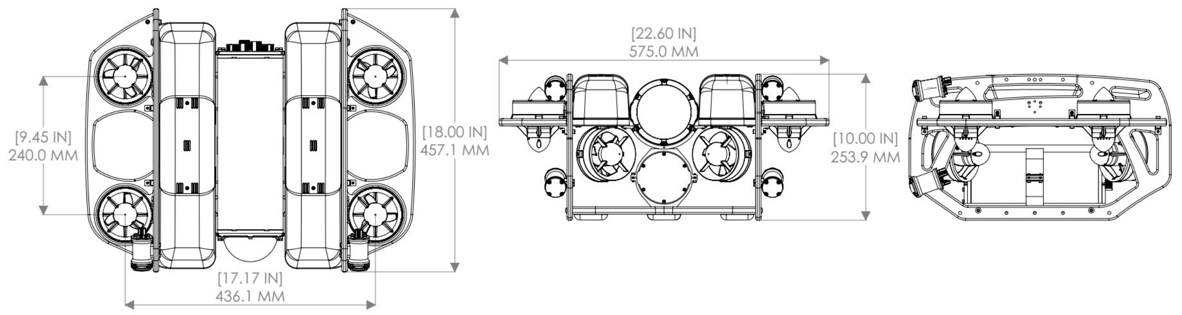


Figure A.1: Technical drawing of the BlueROV2 heavy configuration [5].

A.1 Control Allocation

A control law computes the generalized force vector, $\boldsymbol{\tau}$, which satisfies the control objectives. In order to produce the generalized body force, underwater vehicles are typically actuated by thrusters. Hence, it is necessary to derive a mapping between control input and body force. The problem of deriving the map and computing the control inputs is referred to as the control allocation problem. Depending on the number of actuators, r, the system might be *underactuated* if $r < n$, *fully actuated* if $r = n$, and *overactuated* if $r > n$, where n denotes the number of degrees of freedoms (DOFs). Underwater vehicles are typically modeled with $n = 4$ if the vehicle is strongly self-stabilizing in roll and pitch (the distance between the meta center and center of gravity, GM, is significantly greater than zero) or $n = 6$ if all DOFs

Table A.1: Position and orientation of the thrusters on BlueROV2. x, y, and z denote the position (in meters) of the thruster center relative to the body frame $\mathcal{F}_B$, and α, β are the rotation about the z- and y-axis, respectively.

Thruster r	x	y	z	α	β
Unit	$[m]$	$[m]$	$[m]$	$[°]$	$[°]$
1	0.156	0.111	0.085	0	-45
2	0.156	-0.111	0.085	0	45
3	-0.156	0.111	0.085	0	45
4	-0.156	-0.111	0.085	0	-45
5	0.12	0.218	0.0	90	0
6	0.12	-0.218	0.0	-90	0
7	-0.12	0.218	0.0	-90	0
8	-0.12	0.218	0.0	90	0

needs to be controlled.

For a multivariable system, the control force produces by actuator i is denoted $\boldsymbol{F}_i^b$. Hence, the forces and moments in 6 DOFs are expressed as [15]

$$\boldsymbol{\tau} = \sum_{i=1}^{r} \begin{bmatrix} \boldsymbol{F}_i^b \\ \boldsymbol{r}_{bp_i}^b \times \boldsymbol{F}_i^b \end{bmatrix} = \sum_{i=1}^{r} \begin{bmatrix} F_{x_i} \\ F_{y_i} \\ F_{z_i} \\ F_{z_i} l_{y_i} - F_{y_i} l_{z_i} \\ F_{x_i} l_{z_i} - F_{z_i} l_{x_i} \\ F_{y_i} l_{x_i} - F_{x_i} l_{y_i} \end{bmatrix}, \tag{A.1}$$

where $\boldsymbol{r}_{bp_i}^b = [l_{x_i}\ l_{y_i}\ l_{z_i}]^T$ are the lever arms, that is the perpendicular distances from the CO to the line of action of the force. Given the specifications in Table A.1, the following thrust configuration matrix is obtained

$$\boldsymbol{T} = \begin{bmatrix} 0.7071 & 0.7071 & 0.7071 & 0.7071 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.7071 & 0.7071 & 0.7071 & -0.7071 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1.0000 & 1.0000 & 1.0000 & -1.0000 \\ 0.0601 & -0.0601 & -0.0601 & 0.0601 & -0.2180 & -0.2180 & 0.2180 & 0.2180 \\ 0.0601 & 0.0601 & 0.0601 & 0.0601 & 0.1200 & -0.1200 & 0.1200 & -0.1200 \\ -0.1888 & 0.1888 & -0.1888 & 0.1888 & -0.0000 & 0.0000 & -0.0000 & 0.0000 \end{bmatrix}. \tag{A.2}$$

A.2 Thruster Map

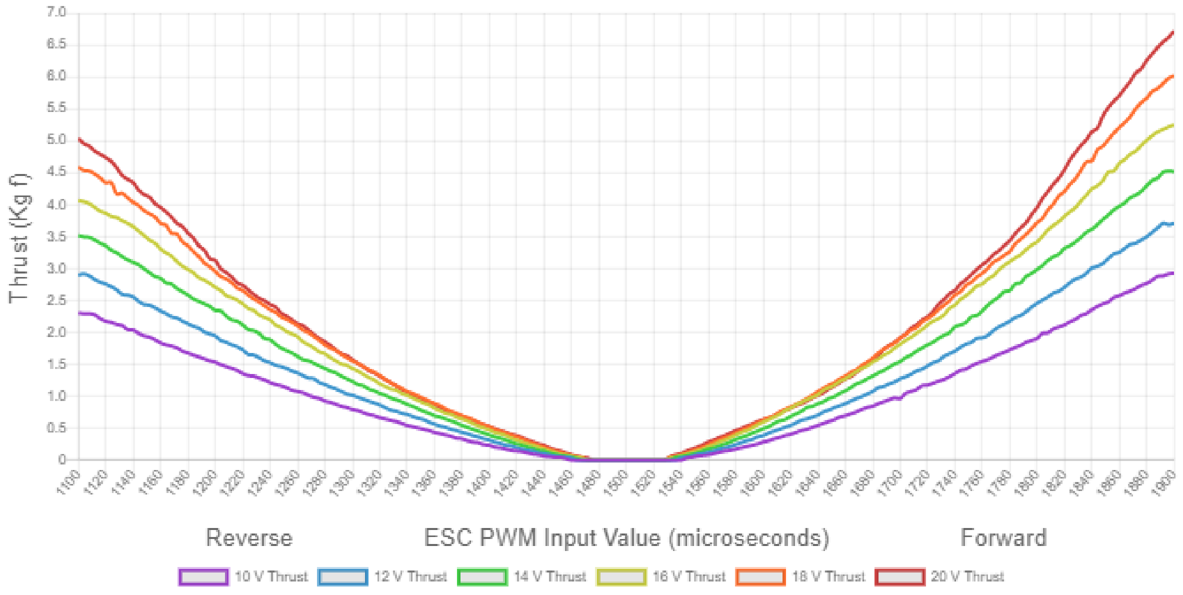


Figure A.2: The thruster map from PWM input value to thrust of the BlueROV2 [5].