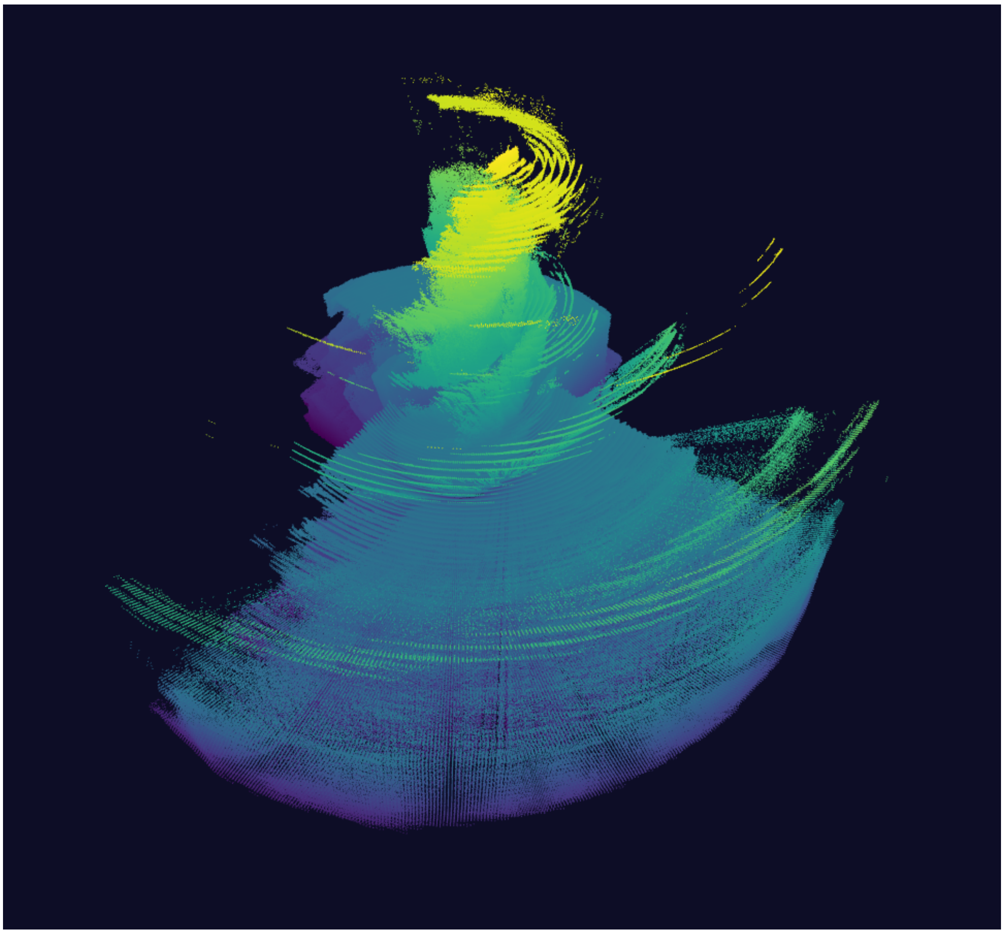

Frequency-domain Calibration of LiDARs

An online method for extrinsic calibration of LiDAR sensors on marine vessels using IMU data and frequency analysis

Master's thesis in Systems, Control and Mechatronics

Elsa Andersson
Thim Högberg

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Frequency-domain Calibration of LiDARs

An online method for extrinsic calibration of LiDAR sensors on marine vessels using IMU data and frequency analysis

ELSA ANDERSSON
THIM HÖGBERG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Vehicle Engineering and Autonomous Systems
Applied Artificial Intelligence Group
Chalmers University of Technology
Gothenburg, Sweden 2026

Frequency-domain Calibration of LiDARs
An online method for extrinsic calibration of LiDAR sensors on marine vessels using IMU data and frequency analysis
ELSA ANDERSSON, THIM HÖGBERG

Supervisor: Axel Måneskiöld, CPAC Systems AB
Examiner: Peter Forsberg, Department of Mechanics and Maritime Sciences

Master's Thesis 2026
Department of Mechanics and Maritime Sciences
Division of Vehicle Engineering and Autonomous Systems
Applied Artificial Intelligence Group
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Accumulated LiDAR point clouds during a roll motion on the test rig.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

Frequency-domain Calibration of LiDARs
An online method for extrinsic calibration of LiDAR sensors on marine vessels using IMU data and frequency analysis
ELSA ANDERSSON, THIM HÖGBERG
Department of Mechanics and Maritime Sciences
Division of Vehicle Engineering and Autonomous Systems
Applied Artificial Intelligence Group
Chalmers University of Technology

Abstract

LiDAR-based autonomous boat operations rely on accurate extrinsic calibration. Current practice requires manual measurement of each LiDAR's position and orientation (pose), which is time-consuming and must be repeated if a sensor is displaced during operation. This thesis presents an online and target-less method for automatic extrinsic calibration of LiDARs mounted at arbitrary positions on a marine vessel.

The proposed method uses the vessel's rigid-body dynamics through a four-step pipeline based on data from the LiDAR's internal IMU. First, the sensor's roll and pitch offsets are determined using gravity projection. Second, a straight-line acceleration maneuver determines the yaw offset. Third, periodic yaw oscillations are analyzed in the frequency domain to estimate the longitudinal and lateral positions. Fourth, roll oscillations are used to estimate the vertical position. The output is a six degrees of freedom pose estimate that serves as an initial guess for a fine calibration using the Generalized Iterative Closest Point (GICP) algorithm.

The method was tested in three environments: simulations with synthetic data, a controlled test rig and a 13 meter vessel. On the test rig, at a known sensor distance of 27.2 cm from the center of rotation (CoR), the longitudinal, lateral and vertical positions were estimated within 1 cm, 3 cm and 1.5 cm respectively. On the vessel, longitudinal position errors were 15-17% (1.04-1.19 m), lateral errors 3-10% (0.05-0.17 m) and vertical errors 23-29% (0.33-0.42 m), all relative to the distance from the CoR. The main limitations were gravity compensation, uncertainty in the estimated CoR and difficulty of producing sinusoidal maneuvers without dedicated maneuvering software.

The fine calibration using GICP was demonstrated on a sensor pair with overlapping field of view of a nearby building. A convergence region test showed that the algorithm converges for initial guess errors below approximately 1.5 m per axis, confirming that the pipeline provides a sufficiently accurate starting point for relative calibration between LiDARs.

Keywords: LiDAR, extrinsic calibration, IMU, frequency domain, marine vessel, autonomous systems, rigid-body dynamics, GICP.

Acknowledgements

We want to give a big thank you to our supervisor Axel Måneskiöld for being very supportive, providing clever ideas and for his help to collect boat data. We also want to thank our examiner Peter Forsberg for his great commitment and innovative tips, as well as for helping us with the test rig and letting us borrow all the tools for the testing. Lastly, we want to thank CPAC Systems for the opportunity to write our thesis here.

Elsa Andersson and Thim Högberg, Gothenburg, June 2026

Thesis advisor: Axel Måneskiöld, CPAC Systems AB
Thesis examiner: Peter Forsberg, Mechanics and Maritime Sciences

List of Acronyms

Below is the list of acronyms used throughout this thesis listed in alphabetical order:

CAN	Controller Area Network
CoR	Center of Rotation
DC	Direct Current
DIFOP	Device Information and Operation Parameters
DoF	Degrees of Freedom
FFT	Fast Fourier Transform
FOG	Fiber Optic Gyroscope
FOV	Field of View
GA	Genetic Algorithm
GICP	Generalized Iterative Closest Point
GNSS	Global Navigation Satellite System
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
LiDAR	Light Detection and Ranging
MAE	Mean Absolute Error
NIR	Near-Infrared
NN	Neural Network
OLS	Ordinary Least Squares
PSO	Particle Swarm Optimization
Radar	Radio Detection and Ranging
RMSE	Root Mean Square Error
SNR	Signal-to-Noise Ratio
ToF	Time of Flight

Nomenclature

Below is the nomenclature of symbols and variables used throughout this thesis.

Greek letters

α	Angular acceleration [rad/s^2]
ω	Angular velocity [rad/s]
φ	Roll angle [rad]
θ	Pitch angle [rad]
ψ	Yaw angle [rad]

Latin letters

a	Acceleration [m/s^2]
b	Lateral offset from CoR [m]
$C(t)$	Common-mode acceleration [m/s^2]
f	Frequency [Hz]
g	Gravitational acceleration (≈ 9.82 m/s^2)
h	Vertical offset from CoR [m]
l	Longitudinal offset from CoR [m]
L	Known distance between two sensors [m]
r	Distance from CoR [m]
R	Rotation matrix
t	Translation vector [m] / Time [s]
T	Homogeneous transformation matrix / Recording duration [s]
v	Velocity [m/s]

Subscripts and superscripts

$(\cdot)_B$	Expressed in the vessel (body) frame
$(\cdot)_S$	Expressed in the sensor frame
$(\cdot)_{\text{tan}}$	Tangential component
$(\cdot)_{\text{corr}}$	Corrected value
$(\cdot)_{\text{raw}}$	Uncorrected value

Contents

List of Acronyms **ix**

Nomenclature **xi**

List of Figures **xvii**

List of Tables **xxi**

1 Introduction **1**

1.1 Previous work 1

1.1.1 Fundamentals of the iterative closest point algorithm 2

1.1.2 Generalized iterative closest point 3

1.1.3 Fundamentals of target-based calibration 3

1.1.4 Biologically inspired optimization methods 4

1.1.5 Optimization in extrinsic LiDAR calibration 5

1.1.6 3D LiDAR and particle swarm optimization 5

1.1.7 Machine learning for online IMU calibration 6

1.1.8 Challenges with deep learning in marine environments 6

1.1.9 IMU-assisted point cloud alignment 7

1.1.10 Using marine dynamics for online calibration 7

1.1.11 Frequency-domain performance evaluation 7

1.1.12 Frequency analysis in IMU calibration 8

1.1.13 Lever arm estimation via frequency analysis 8

1.1.14 Conclusion and research gap 9

1.2 Research questions 9

1.3 Limitations 9

2 Theory **11**

2.1 Vessel kinematics 11

2.2 Sensor calibration 12

2.2.1 Intrinsic and extrinsic parameters 12

2.2.2 Mathematical formulation 13

2.2.3 Calibration approaches 13

2.2.3.1 Offline vs online calibration 14

2.2.3.2 Target-based vs target-less calibration 14

2.2.3.3 Feature-based vs motion-based calibration 14

2.3 LiDAR properties 14

2.4 Different types of LiDARs 15
2.4.1 Mechanical LiDAR 16
2.4.2 Solid-state LiDAR 16
2.5 IMU 16
2.6 Standard ICP 17
2.7 Generalized ICP 17
2.8 Point cloud preprocessing 18
2.8.1 Voxel downsampling 18
2.8.2 KD-tree nearest-neighbor search 18
2.9 Fourier analysis 18
2.9.1 Integration in the frequency domain 19
2.10 Rigid-body dynamics 19
2.10.1 Yaw rotation 19
2.10.2 Roll rotation 20
2.11 Ordinary least squares 20

3 Methods **21**
3.1 Calibration pipeline overview 21
3.2 Step 1: Orientation calibration (roll and pitch) 23
3.2.1 Setup and physics 24
3.2.2 Pitch offset estimation 25
3.2.3 Roll offset estimation 26
3.2.4 Derivation and application of rotation matrices 27
3.2.4.1 Derivation step 1: Deriving a standard 2D rotation via basis vectors 27
3.2.4.2 Derivation step 2: Applying the inverse rotation . . . 28
3.2.4.3 Derivation step 3: Expanding to 3D 29
3.2.4.4 Derivation step 4: Applying the corrections 29
3.3 Step 2: Orientation calibration (yaw) 30
3.3.0.1 Applying the yaw correction 31
3.4 Step 3: Horizontal position calibration 32
3.4.1 Kinematic derivation 33
3.5 Step 4: Vertical position calibration 34
3.6 Center of rotation estimation 35
3.6.1 Yaw CoR: two-accelerometer differential method 35
3.6.2 Roll CoR: two-sensor common-mode correction 36
3.7 Sensitivity analysis of pipeline with simulated data 36
3.7.1 Influence of sensor noise and placement on estimation precision 37
3.7.2 Influence of recording duration on estimation precision 37
3.7.3 Error propagation between calibration steps 37
3.7.4 Effect of maneuver intensity on signal-to-noise ratio 38
3.8 Adaptations for physical sensor data 38
3.8.1 IMU axis identification and DIFOP transform 38
3.8.2 Rodrigues rotation in Step 1 38
3.8.3 Gravity compensation in Step 4 39
3.8.4 Whole-period trimming 39

3.9 Test rig . . . 40
3.9.1 Hardware and data acquisition . . . 40
3.9.2 Step 1: Roll and pitch estimation . . . 41
3.9.3 Step 2: Yaw estimation . . . 41
3.9.3.1 Manual yaw angle for test rig . . . 42
3.9.4 Step 3: Lateral and longitudinal estimation . . . 42
3.9.5 Step 4: Vertical estimation . . . 43
3.9.6 Fine calibration verification . . . 43
3.10 Marine testing . . . 44
3.10.1 Hardware and data acquisition . . . 44
3.10.2 Step 1: Roll and pitch estimation . . . 44
3.10.3 Step 2: Yaw estimation . . . 45
3.10.4 Step 3: Lateral and longitudinal estimation . . . 45
3.10.5 Step 4: Vertical estimation . . . 45
3.10.6 Fine calibration . . . 46
3.10.6.1 Overlap detection and cropping . . . 47
3.10.6.2 Registration procedure . . . 47
3.10.6.3 Convergence region verification . . . 47
3.10.7 Center of rotation estimation . . . 48

4 Results **49**
4.1 Sensitivity analysis of the pipeline with simulated data . . . 49
4.1.1 Influence of sensor noise and placement on estimation precision 49
4.1.2 Influence of recording duration on estimation precision . . . 51
4.1.3 Error propagation between calibration steps . . . 55
4.1.4 Effect of maneuver intensity on signal-to-noise ratio . . . 55
4.2 Test rig . . . 57
4.2.1 Step 1: Roll and pitch calibration . . . 57
4.2.2 Step 2: Yaw calibration . . . 58
4.2.3 Step 3: Horizontal position estimation . . . 59
4.2.3.1 Time-domain signals . . . 59
4.2.3.2 Frequency-domain analysis . . . 60
4.2.3.3 Estimation results . . . 61
4.2.4 Step 4: Vertical position estimation . . . 61
4.2.4.1 Time-domain signals . . . 62
4.2.4.2 Frequency-domain analysis . . . 62
4.2.4.3 Estimation results . . . 63
4.2.5 Fine calibration sensitivity . . . 63
4.3 Marine testing . . . 64
4.3.1 Center of rotation estimation . . . 64
4.3.2 Four-step pipeline results . . . 64
4.3.3 Steps 1 and 2: Orientation estimation . . . 65
4.3.4 Step 3: Horizontal position estimation . . . 65
4.3.4.1 Time-domain signals . . . 65
4.3.4.2 Frequency-domain analysis . . . 66
4.3.5 Step 4: Vertical position estimation . . . 67

4.3.5.1 Time-domain signals . . . 67
4.3.5.2 Frequency-domain analysis . . . 68
4.3.6 Position estimation summary . . . 68
4.3.7 Fine calibration . . . 69
4.3.7.1 Sensor pair selection . . . 69
4.3.7.2 Point cloud overview . . . 69
4.3.7.3 Overlap detection . . . 70
4.3.7.4 Registration result . . . 71
4.3.7.5 Convergence region analysis . . . 72

5 Discussion **75**
5.1 Sensitivity analysis of the pipeline with simulated data . . . 75
5.1.1 Sensitivity and robustness to sensor noise . . . 75
5.1.2 Sequential calibration and orientation accuracy . . . 76
5.1.3 Lever-arm effect and distance-dependent uncertainties . . . 76
5.1.4 The effect of distinct maneuvers . . . 76
5.1.5 Balancing calibration time and frequency-domain resolution . . . 77
5.1.6 Sensor noise vs process noise . . . 77
5.2 Test rig . . . 77
5.2.1 Step 1: Office setup . . . 77
5.2.2 Step 2: Yaw calibration . . . 77
5.2.3 Step 3: Lateral estimation error . . . 78
5.2.4 Step 4: Vertical error . . . 78
5.2.5 Choice of sensor for test rig measurements . . . 78
5.2.6 Fine calibration sensitivity . . . 79
5.3 Marine testing . . . 79
5.3.1 Center of rotation estimation . . . 79
5.3.2 Four-step pipeline . . . 80
5.3.2.1 Position estimation accuracy . . . 80
5.3.2.2 Signal quality and data selection . . . 81
5.3.2.3 Environmental conditions . . . 81
5.3.3 Fine calibration . . . 81
5.3.3.1 FOV overlap as a prerequisite . . . 81
5.3.3.2 Convergence region and pipeline errors . . . 82
5.3.3.3 Environmental requirements . . . 82
5.3.3.4 Absolute vs relative calibration . . . 82
5.4 Future work . . . 83

6 Conclusion **85**

Bibliography **87**

List of Figures

1.1 A 2D illustration of the ICP algorithm. The algorithm finds the nearest points (indicated by arrows) to iteratively align a "reading" shape (red) with a "reference" shape (blue). The right side shows the final alignment. *Image: "Idea behind the closest point algorithm" via Wikimedia Commons (CC0 1.0 / Public Domain).* 2
1.2 Illustration of the setup, including a camera and a 2D laser. Both sensors observe a planar checkerboard target (left). The right side shows the corresponding sensor data: the camera captures a 2D image of the pattern (top right), while the laser records a 2D scan profile of the board's surface (bottom right). Figure adapted from [20] 4

2.1 Illustration of a marine vessel's body-fixed coordinate system, defining its six degrees of freedom. The figure displays the translational motions (surge, sway, heave) along the x, y and z axes, alongside the rotational motions (roll, pitch, yaw) around them. 11
2.2 Schematic overview of a LiDAR system calculating the measured distance to a target to generate a 3D point cloud. 15
2.3 Schematic representation of an IMU showing the three-axis accelerometer and three-axis gyroscope components and how these are integrated to estimate orientation and position. 16

3.1 Illustration of the four-step calibration pipeline. Inputs are accelerometer and gyroscope data and the output is the 6-DOF pose relative to the CoR. 21
3.2 Conceptual illustration of multiple LiDAR positions and their relative distances to the vessel's CoR. 22
3.3 Definition of the roll (φ) and pitch (θ) offset angles between the boat's reference frame (black) and the sensor's reference frame (red). 24
3.4 Vessel seen from the starboard side. The coordinate axes of the boat (B) and the sensor (S) are misaligned around the Y-axis, illustrating a pitch offset. 25
3.5 View from behind the vessel. The coordinate axes of the boat (B) and the sensor (S) are misaligned around the X-axis, illustrating a roll offset. 26
3.6 Derivation of the rotation matrix in the YZ-plane. The original basis vectors $\mathbf{e}_y, \mathbf{e}_z$ (blue) are rotated by α to $\mathbf{e}'_y, \mathbf{e}'_z$ (red), forming the columns of the rotation matrix. 28

3.7 Top-down view of the yaw misalignment ψ. The coordinate axes of the vessel (B) and the sensor (S) are misaligned in the horizontal plane. 30
3.8 Top-down view illustrating the sensor's horizontal position relative to the CoR, showing the longitudinal offset (l) and the lateral offset (b). 32
3.9 Rear view of the vessel illustrating the sensor's vertical position relative to the CoR, showing the vertical offset (h), the lateral offset (b) and the intermediate distance (h'). 34
3.10 Test rig used for Steps 3 and 4, here oriented for Step 3 in its horizontal position. 40
3.11 Setup for Step 2. The LiDAR sensor is secured to a rolling cart using hook-and-loop fasteners and powered through a long cable. 41
3.12 Close-up of the LiDAR sensor mounted on the cart for Step 2. The dome faces the front of the cart and the cable points upward, giving an expected yaw offset of $0° \pm 3°$. 42
3.13 Test rig configured for Step 4. The rig is placed vertically, i.e. rotated 90 degrees compared to Step 3, so the motor creates roll oscillations. . 43
3.14 Illustration of the alternating 10° and 30° yaw rotations performed by the vessel. 45
3.15 Illustration of the roll motion induced by alternating the side thrusters. 46

4.1 MAE of the roll, pitch and yaw estimation as a function of sensor noise. 50
4.2 MAE of the position estimation at a 2 m range as a function of sensor noise. 50
4.3 MAE of the position estimation at a 15 m range as a function of sensor noise. 51
4.4 MAE of the position estimation as a function of sensor distance to the CoR. Note that the lateral error (blue line) overlaps with the longitudinal error (green line). 52
4.5 MAE of the roll and pitch estimation (Step 1) as a function of recording duration. 52
4.6 MAE of the yaw estimation (Step 2) as a function of recording duration. 53
4.7 MAE of the horizontal position estimation (Step 3) as a function of recording duration. 54
4.8 MAE of the vertical position estimation (Step 4) as a function of recording duration. 54
4.9 MAE of the horizontal position estimation (Step 3) as a function of yaw misalignment from Step 2. 55
4.10 MAE of the yaw estimation (Step 2) as a function of forward acceleration magnitude. 56
4.11 MAE of the horizontal position estimation (Step 3) as a function of yaw oscillation amplitude. 56
4.12 MAE of the vertical position estimation (Step 4) as a function of roll oscillation amplitude. 57
4.13 Measured acceleration a_x (blue), a_y (orange) and a_z (green) over time when the sensor was lying flat on a surface. 58

4.14 Measured horizontal acceleration (a_x and a_y) over time during a single cart test trial. Light traces show the raw sensor signal and bold traces show smoothed values. 59
4.15 First 20 seconds of the measured time-domain yaw rate, surge acceleration and sway acceleration during Step 3 rig oscillations. The oscillation period is approximately 2.4 s. 60
4.16 FFT magnitude spectrum of the Step 3 signals. The green-shaded region indicates the bandpass range [0.1, 1.0] Hz. 61
4.17 First 20 seconds of the measured time-domain roll rate and sway acceleration during Step 4 rig oscillations. The oscillation period is approximately 2.0 s. 62
4.18 FFT magnitude spectrum of the Step 4 signals. The green-shaded region indicates the bandpass range [0.1, 1.0] Hz. 63
4.19 120 seconds of the measured signals during yaw oscillations on the boat (Step 3). Light traces show the raw sensor signal and bold traces show smoothed values. 66
4.20 FFT magnitude spectrum of the yaw oscillation signals on the boat (Step 3). The green-shaded region indicates the bandpass range [0.01, 0.7] Hz. 67
4.21 First 20 seconds of the measured signals during roll oscillations on the boat (Step 4). Light traces show the raw sensor signal and bold traces show smoothed values. 68
4.22 FFT magnitude spectrum of the roll oscillation signals on the boat (Step 4). The green-shaded region indicates the bandpass range [0.1, 0.7] Hz. 68
4.23 Both LiDAR point clouds in the vessel coordinate system. Red: bow-mounted Airy. Blue: roof-mounted Helios. The building and the wooden dock (below the building) on the pier are visible to both sensors. 70
4.24 Overlap detection result. Red: Airy points in the overlap region. Blue: Helios points in the overlap region. Gray: discarded non-overlapping points. The overlap radius threshold was 1.2 m. 71

List of Tables

4.1 Step 3 results: estimated sensor position relative to the CoR. 61
4.2 Step 4 results: estimated sensor position relative to the CoR. 63
4.3 GICP convergence at varying displacements. 63
4.4 CoR estimation results from the marine tests. 64
4.5 Pipeline estimation results for starboard and port bow LiDARs. All positions are relative to the CoR (positive lateral values indicate starboard). The CoR is 4.55 m from the stern and 0.13 m below the lower deck floor. 69
4.6 GICP correction from the measured mounting poses. The correction represents the difference between the manually measured relative pose and the GICP-refined alignment. 71
4.7 Per-axis convergence region. Each row shows the convergence rate (out of 20 trials) at a given displacement magnitude along a single axis. 72
4.8 Combined convergence scenarios with simultaneous displacement along all three axes. The pipeline error magnitudes are taken from Table 4.5. Signs are randomized across 20 trials. 72

1

Introduction

Safe autonomous operations rely on robust perception systems. While traditional marine vessels depend on navigational sensors like Radio Detection and Ranging (radar) and Global Navigation Satellite Systems (GNSS), the transition toward increased autonomy requires the integration of high-resolution sensors, primarily camera and Light Detection and Ranging (LiDAR). The precise 3D spatial data provided by LiDAR is useful for maneuvers such as autonomous docking as well as object detection, especially in low-light conditions. To fuse the data from these sensors into a unified and reliable representation of the environment, they must be extrinsically calibrated. Correct calibration is a prerequisite for all downstream tasks, including perception, trajectory prediction and control [8].

In the automotive industry, sensors are typically mounted in standardized positions, which simplifies automatic calibration. In contrast, marine vessels, especially larger ones, are manufactured in much smaller volumes and come in many different shapes, leading to a wide variety of sensor placements. To achieve a 360-degree view around the vessel, multiple LiDARs are needed. If these are not accurately extrinsically calibrated against each other, overlapping point clouds can result in artifacts, such as the duplication or "ghosting" of objects. Consequently, current practices rely on manual calibration, which is time-consuming, as the exact position and orientation of each LiDAR must be physically measured. Moreover, if a sensor is displaced during operation it must be recalibrated. Offline recalibration requires taking the vessel out of operation, leading to downtime. An ideal solution would allow for the arbitrary placement of LiDARs anywhere on the vessel, with a system subsequently performing the calibration automatically while the vessel is at sea.
Furthermore, calibrating LiDARs on water introduces unique challenges compared to land vehicles. Boats move constantly due to waves and the open water lacks stationary objects, such as buildings or signs, that the LiDARs can use as reference points. Another issue is that LiDAR struggles to detect the water surface. Since these sensors operate in the near-infrared spectrum, the laser pulses are often absorbed by the water or lost to specular reflection, which can create data gaps in the point clouds [11].

1.1 Previous work

This section provides an overview of previous research relevant to this thesis. First, it covers the fundamentals of point cloud registration and traditional calibration methods, including both target-based and feature-based approaches. Second, it covers global optimization techniques, such as biologically inspired algorithms and their application in LiDAR calibration. Third, it covers the limitations of machine learning in marine environments. Finally, it covers recent advancements in using marine dynamics and frequency-domain analysis for sensor calibration, which forms the basis for the proposed method in this thesis.

1.1.1 Fundamentals of the iterative closest point algorithm

The Iterative Closest Point (ICP) algorithm, originally introduced by Besl and McKay [21], remains a fundamental and widely used mathematical framework for targetless point cloud registration. Provided a good initial guess, the core principle of ICP is straightforward and effective. It iteratively aligns a "reading" point cloud with a "reference" point cloud [22].

The standard algorithm iterates between two primary steps, as illustrated in Figure 1.1. First, it pairs every point in the "reading" cloud with its nearest neighbor in the "reference" cloud. Second, it calculates the optimal translation and rotation to minimize the distance between these paired points [22]. The reading cloud is then transformed and the process is repeated until the alignment converges.

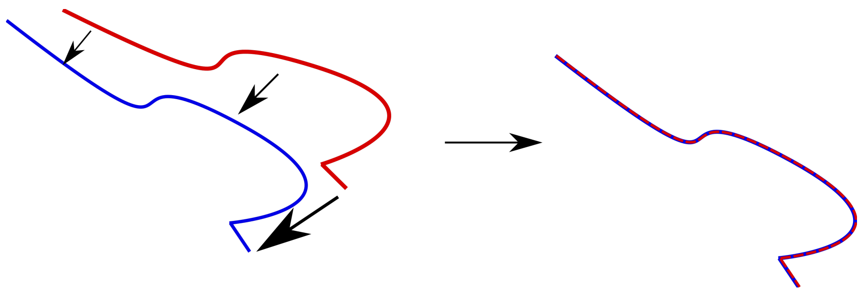

Figure 1.1: A 2D illustration of the ICP algorithm. The algorithm finds the nearest points (indicated by arrows) to iteratively align a "reading" shape (red) with a "reference" shape (blue). The right side shows the final alignment. *Image: "Idea behind the closest point algorithm" via Wikimedia Commons (CC0 1.0 / Public Domain).*

While standard ICP is computationally fast and mathematically well-established, it relies entirely on finding the nearest points. This makes it dependent on an initial starting position. If the initial guess is poor, the algorithm will pair the wrong points together, causing the optimization to fail and converge to an incorrect local minimum [22]. This limitation drives the need for alternative and more robust optimization methods.

1.1.2 Generalized iterative closest point

Segal et al. introduced the Generalized Iterative Closest Point (GICP) algorithm as a more robust alternative to the standard ICP [28]. GICP introduces a probabilistic model that generalizes the ICP algorithm, meaning a single mathematical equation can handle both direct point matching and matching points to flat surfaces.

Instead of viewing a point cloud as just a collection of isolated dots, GICP analyzes the physical shape of the scanned environment. For every single point, the algorithm looks at its closest neighboring points. By calculating how these neighbors are distributed in space using a covariance matrix, GICP can determine the local surface structure [28]. In practice, this means each point is transformed from a zero-dimensional dot into a representation of a small, localized plane.

In the final alignment step, GICP compares the locally modeled planes from both point clouds. Instead of forcing strict point-to-point distance calculations, it allows points to slide freely along matching flat surfaces. It only penalizes errors when points deviate perpendicularly from these surfaces [28]. Because it focuses on aligning the underlying shapes rather than isolated dots, GICP handles noisy data better and is less prone to getting stuck in incorrect local minima compared to standard ICP.

1.1.3 Fundamentals of target-based calibration

Extrinsic calibration simply means finding the position and orientation (also known as pose) of a sensor relative to a reference point or another sensor. In 2004, Zhang and Pless [20] introduced what is widely considered the first published method for calibrating a camera and a 2D laser using a physical target. In their setup, the camera cannot see the laser, making it difficult to calculate how the sensors are mounted relative to each other.

To solve this, the authors used a planar checkerboard as a calibration target, illustrated in Figure 1.2. Their method relies on a simple geometric constraint: when both sensors observe the checkerboard, the laser points must mathematically align with the plane estimated by the camera. By observing the checkerboard in multiple poses, the system calculates the exact extrinsic parameters. Additionally, optimizing these constraints also improves the camera's internal calibration, such as its focal length.

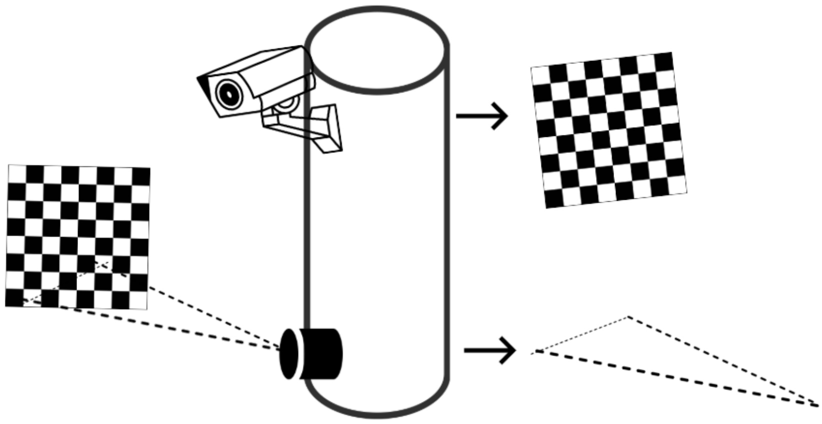

Figure 1.2: Illustration of the setup, including a camera and a 2D laser. Both sensors observe a planar checkerboard target (left). The right side shows the corresponding sensor data: the camera captures a 2D image of the pattern (top right), while the laser records a 2D scan profile of the board's surface (bottom right). Figure adapted from [20]

However, the paper also demonstrates the limitations of target-based methods. The algorithm requires a physical calibration board to be manually placed in specific orientations [20]. Because both sensors must simultaneously observe this physical object, the calibration relies entirely on a controlled offline setup. Consequently, it cannot run continuously using the natural environment.

To overcome the need for artificial targets, modern systems use feature-based calibration. Instead of a checkerboard, these methods look for natural shapes in the environment, such as corners, edges or flat surfaces. For example, Hänsch et al. [23] showed how 3D point clouds can be aligned by automatically detecting and matching these natural features. Moving away from artificial targets is necessary for creating calibration systems that can run continuously in the real world.

1.1.4 Biologically inspired optimization methods

As stated, ICP often fails if the initial alignment is poor. Therefore, global optimization methods are needed. Among the most robust are biologically inspired optimization methods, which mimic processes found in nature to efficiently search large and non-linear search spaces [24].

Two of the most common algorithms in this field are Genetic Algorithms (GA) and Particle Swarm Optimization (PSO). GA is inspired by Darwinian evolution and the principle of natural selection [24]. It works by generating a "population" of possible calibration solutions. Through processes that imitate mating (crossover) and random genetic changes (mutation), the best-performing solutions survive and pass their traits to the next generation. Over several iterations, the population gradually evolves toward the optimal sensor alignment.

Similarly, PSO is based on the social behavior of animals, such as a flock of birds or a school of fish [24]. Instead of evolving, a swarm of "particles" flies through the search space, where each particle represents a potential mathematical solution. Every particle continuously adjusts its flight path based on its own past success and the best position found by the entire swarm. This collaborative behavior allows the swarm to quickly converge on the best overall solution.

Because both GA and PSO evaluate many potential solutions simultaneously, they are effective at finding the correct extrinsic calibration from scratch, completely removing the reliance on a good initial guess.

1.1.5 Optimization in extrinsic LiDAR calibration

Choosing the right optimization algorithm is important, especially when dealing with large search spaces and non-linear optimization problems. A previous master's thesis [15] investigated the offline extrinsic calibration of four 2D LiDARs mounted on a moving truck. The goal was to estimate the x, y and yaw coordinates for all four sensors simultaneously, resulting in a 12-parameter optimization problem.

The authors in [15] compared the ICP algorithm with a stochastic optimization approach, namely GA. The results showed that while ICP is mathematically well-established, it is dependent on a good initial guess and often fails if the overlap between LiDAR scans is too small. In contrast, the GA method proved to be significantly more robust, successfully finding the correct calibration in over 95% of the test cases without relying on feature extraction. The GA evaluated the calibration quality using a circular occupancy grid, where a higher alignment of point clouds resulted in a higher fitness score.

Furthermore, the study highlighted that calibrating on a moving vehicle introduces distortions like scan skewing and timing inconsistencies. While GA effectively overcomes ICP's sensitivity to poor initial guesses in dynamic environments, it is too computationally expensive for real-time use [15]. This emphasizes the need for lighter optimization algorithms, such as PSO, to achieve continuous online calibration.

1.1.6 3D LiDAR and particle swarm optimization

Building upon previous 2D calibration methods [15], a subsequent master's thesis [16] explored the extrinsic calibration of multiple 3D solid-state LiDARs using PSO. The authors used a voxel-grid-based fitness function, where the optimization algorithm aims to maximize point overlap within voxels. Here, a higher number of overlapping points indicates a better calibration alignment.

A major finding of this study was that PSO outperformed both GA and the standard ICP algorithm in terms of robustness and computational efficiency [16]. The authors noted that standard ICP frequently diverged because it assumes point clouds are identical, whereas sensors mounted at different positions observe different features and occlusions.

However, the study had several limitations. First, it simulated an unrealistically large 270-degree field of view (FOV), creating artificially high data overlap. Second, using cubical voxels for spherical LiDAR data caused matching errors that sometimes rewarded incorrect calibrations. Third, their online tracking failed by ignoring the vehicle's movement between scans. Finally, locking certain sensor positions to save time removed the system's flexibility to globally balance errors, ultimately worsening the calibration.

1.1.7 Machine learning for online IMU calibration

While traditional optimization and geometric approaches remain the standard for sensor calibration, some studies have explored the use of machine learning as an alternative. A previous master's thesis [17] investigated the use of Neural Networks (NNs) for the intrinsic calibration of IMUs. The authors hypothesized that a nonlinear NN approximation would model the sensor characteristics better than a standard linear model. During offline training, the NN successfully lowered the calibration error compared to traditional methods.

However, the study found that continuously retraining an NN online is computationally heavy and performs worse than a simple linear model. Moreover, the method failed in dynamic environments because it unrealistically required the sensor to remain completely stationary between data episodes.

Although [17] focuses on intrinsic calibration, it shows that heavy machine learning models are over-engineered for real-time tracking. Because their method requires the sensor to be stationary, it cannot be used on a moving boat. Therefore, lightweight methods that can handle constant motion are necessary.

1.1.8 Challenges with deep learning in marine environments

A master's thesis by Dahlin and Jonsson [25] investigated how to process LiDAR data on boats in real time. To reduce the heavy computational load, they flattened the 3D point clouds into 2D images before analyzing them with a NN.

Although their network successfully segmented the data, the method faced two major bottlenecks for real-time marine use. First, the lack of varied marine training data limited the model's reliability. Second, running heavy deep learning models in

Python was simply too slow for real-time performance on edge devices.

Even though the work by [25] focuses on segmentation, it highlights important challenges for marine LiDAR processing using deep learning. The lack of training data and the delay of heavy AI models indicate that relying on deep learning for real-time tasks can be impractical.

1.1.9 IMU-assisted point cloud alignment

A recent study by Das et al. [18] proposed an online extrinsic calibration method for multiple LiDARs that specifically handles the initialization problem, i.e., the difficulty of establishing a reliable starting pose without prior estimates. Their approach uses IMUs placed at the same location as each LiDAR to generate an initial extrinsic guess by matching raw inertial signals, such as angular velocity and linear acceleration.

After establishing the initial estimate, the system fine-tunes the calibration using GICP. Additionally, the method improves efficiency by filtering the IMU data to only use the most informative measurements. This work is relevant as it validates a two-step pipeline where the LiDAR's built-in IMU provides the initial guess and GICP handles the fine-tuning.

1.1.10 Using marine dynamics for online calibration

While dynamic motion can be viewed as a challenge for sensor calibration, some methods actively use it. Chang et al. [19] proposed an online calibration method specifically designed for a ship-based Fiber Optic Gyroscope (FOG) IMU. Rather than requiring the vessel to be stationary, their approach makes full use of the natural swing motion caused by sea waves to continuously calibrate the inertial parameters in real time. The authors used a Kalman filter to continuously estimate the IMU's internal errors during sailing. This study shows that wave-induced motion can be used for online calibration and that lightweight Kalman filters work well in real-time marine settings.

1.1.11 Frequency-domain performance evaluation

While most research evaluates IMU filters in the time domain, recent studies use frequency analysis to better understand performance under dynamic conditions. A study by Sultan and Greiser [13] evaluated different IMU orientation algorithms (such as Kalman, Madgwick and Complementary filters) in both the time and frequency domains.

To test how well the filters handled fast and dynamic movements, the researchers in [13] used a very accurate robotic arm as a reference ground truth. They applied fast-changing noise signals to the IMU and analyzed the results using frequency response metrics. The results showed that the Madgwick and Complementary filters provided the most stable performance across both domains.

While [13] evaluates intrinsic IMU performance, it shows that frequency-domain analysis is better for capturing noise and dynamic behaviors during complex motion. This supports using frequency analysis in online calibration to identify sensor characteristics that are difficult to detect in the time domain.

1.1.12 Frequency analysis in IMU calibration

Estimating sensor bias and offset in dynamic marine environments is challenging due to external disturbances like engine vibrations and waves. A recent study by Fukuda et al. [14] evaluated several algorithms, including Fast Fourier Transform (FFT), to estimate IMU bias using data from real ships. By looking at the 0 Hz frequency (Direct Current (DC) component), the authors found that FFT worked perfectly in computer simulations and was good enough to calibrate gyroscopes.

However, when the method was used on accelerometer data in a real marine environment, the frequency analysis became less accurate. Mathematically, it did not perform any better than just taking a simple average. The authors in [14] concluded that uncorrected lever-arm errors (errors caused by the physical offset between the sensors) and unfiltered ship movements caused the estimations to become unstable. To fix this in a real-world scenario, specific low-pass filters (like a Butterworth filter) were needed to actively reduce noise frequencies before the bias could be found.

These findings from [14] show the gap between theoretical computer simulations and real-world use. For online calibration of LiDAR and IMU systems at sea, this means that frequency-based methods must be combined with strong pre-filtering and proper compensation for lever-arm offsets to achieve reliable results.

1.1.13 Lever arm estimation via frequency analysis

A recent patent by Lund [26] introduces a method for determining the position of a translational movement sensor on a marine vessel. The method receives a time-varying rotational movement signal (e.g. from a gyroscope) and a time-varying translational movement signal (e.g. from an accelerometer) and transfers both to the frequency domain. At the reference frequency where the rotational movement amplitude is largest, the algorithm extracts the corresponding translational movement amplitude. The ratio between these amplitudes gives information about the sensor's distance from the center of rotation (CoR).

By performing this procedure for different rotational axes (yaw and roll), both the horizontal position (longitudinal and lateral offsets) and the vertical position of the sensor can be determined. The method can use velocity, acceleration or displacement signals. An important advantage is that the vessel's own periodic motions (steering maneuvers or wave-induced roll) provide the excitation signal, so no external calibration equipment is needed. The patent does not address the orientation (roll, pitch, yaw offsets) of the sensor.

1.1.14 Conclusion and research gap

The literature review reveals that while traditional methods like ICP are efficient, they remain sensitive to initial estimates. Biologically inspired algorithms like PSO and GA offer a solution to the initialization problem but are often computationally heavy. Furthermore, while machine learning shows potential, its reliance on training data and computational power makes it less suitable for real-time calibration on marine vessels.

A research gap exists in the use of frequency-domain analysis for marine sensor calibration. While recent studies and the patent by Lund [26] suggest that frequency analysis is a promising tool for estimating lever-arm positions, it has not yet been integrated into a complete LiDAR calibration method that also determines orientation.

Therefore, the goal of this project is to develop a complete method that determines the sensor orientation, uses frequency analysis and vessel dynamics to estimate the sensor position and refines the result using point cloud registration.

1.2 Research questions

This thesis aims to answer the following research questions by developing a method for automatic, online extrinsic calibration of LiDAR sensors and their integrated IMUs:

- How can the position and orientation (pose) of a LiDAR sensor be determined automatically when mounted at an arbitrary position on a marine vessel?
- To what extent can frequency-domain analysis and vessel dynamics be used to accurately estimate the pose of a LiDAR using IMU data?
- How does the proposed method perform in terms of accuracy when transitioning from simulated environments to real-world dynamic marine conditions?

1.3 Limitations

This thesis is subject to the following limitations:

- Only extrinsic calibration is considered. Intrinsic calibration of the sensors is assumed to have been performed correctly by the respective manufacturers.
- The LiDAR sensor and its integrated IMU are treated as a single rigid body. The vessel is also assumed to be a rigid body.
- The project scope is limited to the development and validation of the calibration pipeline. Integration into a production system is not addressed.

- Marine testing is limited to calm conditions. The performance under difficult conditions such as large waves is outside the scope of this work.
- During the marine tests, the placement and orientation of the LiDARs were restricted to pre-existing mounting locations.

2

Theory

This chapter introduces the theoretical foundations required for the proposed calibration method. It covers vessel kinematics, sensor calibration principles, LiDAR and IMU sensor properties, point cloud registration algorithms, Fourier analysis and the rigid-body dynamics that form the basis of the position estimation.

2.1 Vessel kinematics

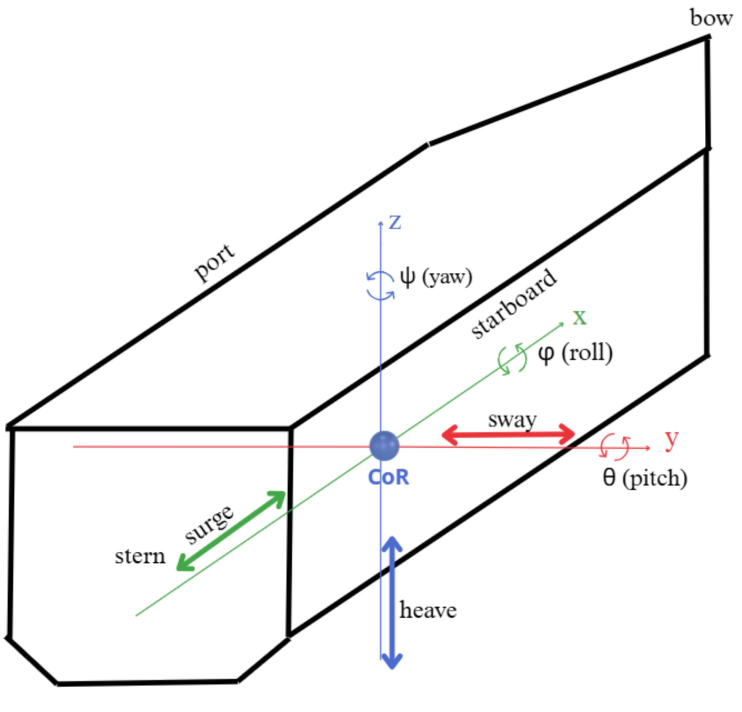

Figure 2.1: Illustration of a marine vessel's body-fixed coordinate system, defining its six degrees of freedom. The figure displays the translational motions (surge, sway, heave) along the x, y and z axes, alongside the rotational motions (roll, pitch, yaw) around them.

A marine vessel is a "hollow structure made to float upon water for purposes of transportation and navigation; especially one that is larger than a rowboat" [7]. It is characterized by its primary parts: the bow (front), stern (rear), port (left) and starboard (right). Its motion is defined by six degrees of freedom relative to a body-fixed coordinate frame, with its origin at the vessel's center of rotation. The frame consists of three orthogonal axes, the longitudinal, lateral (transverse) and vertical, as illustrated in Figure 2.1. A boat can rotate around these axes as well as translate along them. Together, these three rotational and three translational movements define the boat's motion in the water.

The coordinate system used in this project is defined according to the patent [26]. For translational movements, the X-axis points forward towards the bow of the vessel, representing surge. The Y-axis points to the right towards the starboard side, representing sway. Finally, the Z-axis points vertically upwards towards the sky, representing heave. This axis setup creates a left-handed coordinate system for translation. This is different from the standard right-handed North-East-Down system normally used in marine navigation.

However, all rotational movements around these axes strictly follow the right-hand rule. Rotation around the X-axis is called roll (φ) [7], where a positive angle indicates a clockwise rotation when looking forward, causing the starboard side to tilt downwards. Rotation around the Y-axis is known as pitch (θ) and a positive angle means the bow rises while the stern sinks. Lastly, rotation around the Z-axis is referred to as yaw (ψ), where a positive angle results in a counter-clockwise turn when viewed from above, steering the vessel to the left.

2.2 Sensor calibration

Sensor calibration is the process of determining an accurate relationship between a sensor's raw measurements and the physical world. It is a requirement for autonomous systems to perceive their environment correctly. Calibration can be divided into two main categories: intrinsic and extrinsic [8].

2.2.1 Intrinsic and extrinsic parameters

Intrinsic calibration addresses the internal parameters and hardware imperfections of a specific sensor. For a LiDAR, intrinsic parameters typically include distance measurement offsets and individual beam angles. These parameters are typically calibrated by the manufacturer before the sensor is sold [12].

Extrinsic calibration determines, from a physical point of view, the position (x, y, z) and orientation (roll, pitch, yaw), i.e. the pose, of the sensor relative to a local reference point, such as a vessel's CoR, or another sensor [8]. This is important for sensor fusion, as it allows multiple sensors to be represented in a single, common coordinate frame. Without accurate extrinsic calibration, the system may suffer from a form of "double vision". This occurs when a single physical object is detected by multiple sensors but is registered at different coordinates in the system's frame.

This thesis is concerned with finding the pose of a sensor, i.e. only extrinsic calibration is considered. Calibration thus refers to extrinsic calibration throughout this thesis.

2.2.2 Mathematical formulation

From a mathematical point of view, extrinsic calibration is the process of finding the rotational and translational matrices needed to merge spatial data from a local sensor into a common coordinate system. This mathematical mapping ensures that a physical object observed by the sensor is registered at its true location relative to the vessel.

The transformation relies on a 3×3 rotation matrix R, which aligns the orientation, and a 3×1 translation vector t, which applies the positional distances:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \quad t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{2.1}$$

The rotation matrix R is defined by Euler angles to describe the orientation as a sequence of three rotations. When the transformation from a reference frame to a target frame is defined using the ZYX sequence, the individual rotation matrices are multiplied in the following order:

$$R = R_x(\varphi) \cdot R_y(\theta) \cdot R_z(\psi) \tag{2.2}$$

where the rotations are applied in the order of yaw (ψ), pitch (θ) and roll (φ).

To perform both rotation and translation in a single step, these components are combined into a 4×4 homogeneous transformation matrix T:

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

Using this matrix, a 3D point P_{source} recorded in the sensor's local coordinate frame is mapped to P_{target}, which is the exact same physical point expressed in the vessel's target coordinate frame:

$$P_{target} = T \cdot P_{source} \implies \begin{bmatrix} x_{target} \\ y_{target} \\ z_{target} \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{source} \\ y_{source} \\ z_{source} \\ 1 \end{bmatrix} \tag{2.4}$$

2.2.3 Calibration approaches

The procedure to find the transformation matrix T can be categorized by when the calibration occurs, whether it requires specific objects and what data it analyzes.

2.2.3.1 Offline vs online calibration

Calibration can be conducted offline (when the, in this case, vessel is inactive) or online (when it is active). It is not uncommon that marine LiDARs today are calibrated offline by manually measuring distances and rotations, based on the sensors' approximate positions and orientations on the vessel. This manual process is prone to human error due to the physical placement of the sensors. Furthermore, if a sensor is accidentally moved or twisted during operation, for example by a mooring line, it must be recalibrated. Offline recalibration requires taking the vessel out of operation, leading to costly downtime.

2.2.3.2 Target-based vs target-less calibration

Calibration methods can be divided into target-based, which require specific artificial calibration objects, and target-less, which rely entirely on the natural environment [8]. Traditional offline calibration is typically target-based, whereas achieving continuous online calibration requires target-less approaches.

2.2.3.3 Feature-based vs motion-based calibration

Within target-less online calibration, algorithms typically analyze either the environment or the sensor's movement [8]:

Feature-based calibration relies on finding geometric structures in the environment, such as corners or flat walls, and matching them between different sensors in their overlapping FOV. While this method can be effective on land, it can struggle in marine environments. Water surfaces are reflective, absorb LiDAR pulses and constantly change shape, resulting in a lack of stable geometric features [11].

Motion-based calibration focuses on the movement paths of the sensors rather than matching specific objects between them. While the trajectories themselves are often created using visual features (visual odometry), the calibration process only requires the calculated motion data, making it possible to calibrate sensors that do not have overlapping FOV.

Additionally, the concept of self-calibration is preferred within these methods. Self-calibration refers to an approach that uses the correspondence between data captured during motion to estimate a sensor's intrinsic and extrinsic parameters [8]. This allows the system to calculate its own pose using only internal data, e.g. from an IMU.

2.3 LiDAR properties

LiDAR is a sensing method that uses light to measure distances to objects. LiDAR systems mostly work with infrared light, especially near-infrared (NIR) wavelengths [4]. LiDAR can create accurate 3D models of the environment, making it used in areas such as autonomous vehicles, aerial mapping and industrial safety [6].

A method called Time-of-Flight (ToF) [3] measures the time it takes for a light pulse to reach an object and return, allowing the distance to be calculated using the following equation:

$$d = \frac{c \cdot \Delta t}{2} \tag{2.5}$$

where c is the speed of light and Δt is the total time it takes for the light to be emitted, reflected and detected by the sensor.

Figure 2.2 illustrates how this works in practice. The LiDAR's laser sends out a beam (reference light) towards a target object. When the light hits the object, it bounces back (reflected light) and is detected by the scanner. By calculating the measured distance for millions of these light beams in different directions, the system creates a 3D point cloud, which is a digital map of the environment.

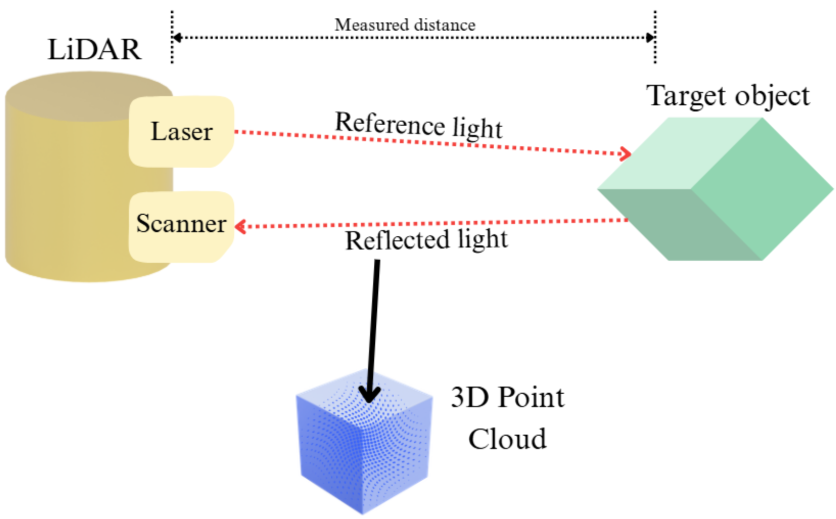

Figure 2.2: Schematic overview of a LiDAR system calculating the measured distance to a target to generate a 3D point cloud.

2.4 Different types of LiDARs

LiDARs come in 2D and 3D versions. A 2D LiDAR generates a single line because it scans in only one plane to measure distance, whereas a 3D LiDAR generates a full point cloud of the environment. In this thesis, only 3D LiDARs are used. There are various kinds of 3D LiDARs, with the two main types on the market being mechanical and solid-state. The LiDARs differ in how the scanner directs the beam. The way the laser beam is directed affects the precision, speed, FOV and object detection, which in turn affects the final resolution [5].

2.4.1 Mechanical LiDAR

Mechanical LiDARs can create a wide, 360-degree FOV. Their optical design and high laser power can achieve a high signal-to-noise ratio, enabling reliable long-range detection. Historically, these systems have been limited by high cost, large physical dimensions, high power consumption and sensitivity to vibrations [5]. While recent models have reduced the cost and size of rotating sensors, their reliance on moving parts still presents long-term reliability and maintenance considerations compared to non-mechanical options.

2.4.2 Solid-state LiDAR

Solid-state LiDARs do not have any rotating components, and therefore their individual FOV is reduced compared to mechanical LiDARs. To overcome this, multiple sensors can be placed around the vehicle or vessel to combine their data into a wider FOV. Overall, solid-state LiDARs are robust due to not having moving parts, while offering high resolution and fast scanning speeds [5].

2.5 IMU

An Inertial Measurement Unit (IMU) is a sensor that consists of a three-axis accelerometer and a three-axis gyroscope, providing a total of six degrees of freedom (DOF) [9]. The gyroscope measures the angular velocity of the sensor, which is the rate of change of its orientation. The accelerometer measures the external specific force along three axes, also known as g-force. This specific force is defined as the sum of the sensor's acceleration and the Earth's gravity:

$$a = a_g + a_s \tag{2.6}$$

When the object is at rest, the sensor measures only the gravity vector, $a = a_g$, with a magnitude of approximately $9.82 \text{ m/s}^2 = 1 \text{ g}$.

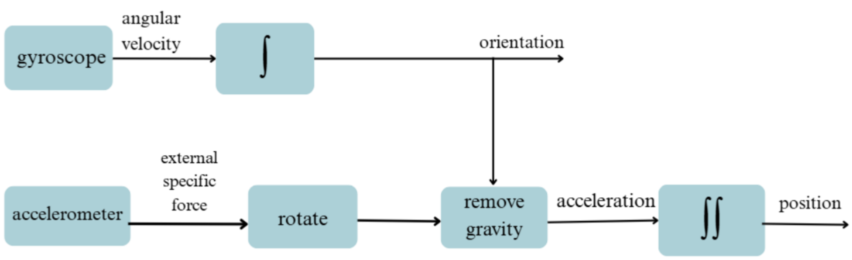

Figure 2.3: Schematic representation of an IMU showing the three-axis accelerometer and three-axis gyroscope components and how these are integrated to estimate orientation and position.

An IMU that is rigidly attached to an object can provide information about the pose, meaning the position and orientation, of that object. The process of calculating the position and orientation is illustrated in Figure 2.3 and is known as dead reckoning. The orientation is determined by integrating the angular velocity. The position is found by first removing the gravity component and then double-integrating the acceleration data. It is important to note that the orientation must be known before gravity can be correctly removed from the measurements.

In practice, the pose estimate is not perfect because inertial measurements contain noise and bias. Due to these factors, the integration steps lead to integration drift. This error accumulates over time and impacts the quality of the estimated pose. Consequently, both the orientation and position will drift away from their true values over time [9].

2.6 Standard ICP

The standard Iterative Closest Point algorithm, as proposed by Besl and McKay [21] views each point simply as an isolated coordinate, without considering the physical shape it belongs to. Given a source point cloud $\mathcal{A} = \{a_i\}$ and a target point cloud $\mathcal{B} = \{b_i\}$, the goal is to find the rigid transformation T (consisting of a rotation R and translation t) that minimizes the sum of squared Euclidean distances:

$$E_{ICP} = \sum_i \|Ta_i - b_i\|^2 \tag{2.7}$$

This means the algorithm penalizes distance errors equally in every direction. It assumes the uncertainty around each point is perfectly spherical, rather than adapting to the actual shape of the surface.

2.7 Generalized ICP

The Generalized ICP (GICP) algorithm [28] improves upon this by attaching a covariance matrix to each point, effectively modeling the local surface geometry. Each point a_i and b_i is assumed to be a sample from a Gaussian distribution:

$$a_i \sim \mathcal{N}(\hat{a}_i, C_i^A), \quad b_i \sim \mathcal{N}(\hat{b}_i, C_i^B) \tag{2.8}$$

where C_i^A and C_i^B are the covariance matrices representing the local surface orientation at each point. The transformation T is then found by minimizing the following objective function:

$$E_{GICP} = \sum_i d_i^T \left(C_i^B + TC_i^A T^T\right)^{-1} d_i \tag{2.9}$$

where $d_i = Ta_i - b_i$ represents the residual vector between the paired points.

The strength of GICP lies in the inverse weighting matrix $(C_i^B + TC_i^A T^T)^{-1}$. This matrix assigns a low penalty to errors along a flat surface, allowing points to "slide" freely along matching planes. In contrast, any deviation perpendicular to the surface is penalized. This makes sure that the algorithm aligns the underlying geometric structures rather than forcing strict point-to-point matches.

2.8 Point cloud preprocessing

Before running registration algorithms such as ICP, the raw point clouds can be preprocessed to reduce computational cost and improve alignment quality.

2.8.1 Voxel downsampling

Processing raw LiDAR scans can be computationally expensive. Voxel downsampling divides a 3D space into a regular grid of cubic cells (voxels) with a specified side length. Within each voxel, all points are replaced by a single representative point, typically the centroid. This reduces the point count while preserving the overall shape of the scene. A smaller voxel size retains more detail but keeps more points, while a larger voxel size gives a coarser representation with fewer points.

2.8.2 KD-tree nearest-neighbor search

Both ICP algorithms and overlap detection rely on finding the closest point in a source cloud (with M points) to each point in a target cloud (with N points). A brute-force search has a complexity of $\mathcal{O}(M \cdot N)$, which is slow for large datasets. To solve this, a KD-tree divides the 3D space into smaller regions. This rules out distant regions during the search, reducing the total search complexity to an average of $\mathcal{O}(M \log N)$.

2.9 Fourier analysis

The Fourier transform decomposes a time-domain signal into its frequency components. For a continuous signal $x(t)$, the Fourier transform is defined as:

$$X(f) = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi ft}\, dt \tag{2.10}$$

where f is the frequency in Hz and j is the imaginary unit. The magnitude $|X(f)|$ gives the amplitude at frequency f, and the phase $\angle X(f)$ gives the time shift of that component.

In practice, signals are sampled at discrete time steps. The FFT is an efficient algorithm that computes the discrete version of this transform. For a signal recorded over a duration T seconds, the frequency resolution of the FFT is:

$$\Delta f = \frac{1}{T} \tag{2.11}$$

A longer recording allows the FFT to distinguish between frequencies that are closer together.

2.9.1 Integration in the frequency domain

A useful property of the Fourier transform is that integration in the time domain corresponds to division by the angular frequency in the frequency domain. If $x(t)$ has Fourier transform $X(f)$, then the integral $\int x(t)\,dt$ has the transform:

$$\frac{X(f)}{j \cdot 2\pi f} \tag{2.12}$$

This means that to convert an acceleration signal to velocity, one can divide each frequency component by $j \cdot 2\pi f$ instead of performing numerical integration in the time domain. This avoids the accumulation of drift errors that time-domain integration suffers from.

2.10 Rigid-body dynamics

This section describes how an accelerometer on a rotating vessel can be used to determine its distance from the CoR. These relationships form the theoretical basis for both the patent method [26] and the CoR estimation described in Section 3.6.

When a rigid body rotates, every point on it traces a circular arc around the CoR. A point at a perpendicular distance r from the CoR has a tangential velocity

$$v_{\text{tan}} = r \cdot \omega \tag{2.13}$$

where ω is the angular velocity. The further a point is from the CoR, the faster it moves. Differentiating with respect to time gives the tangential acceleration:

$$a_{\text{tan}} = r \cdot \alpha \tag{2.14}$$

where $\alpha = \dot{\omega}$ is the angular acceleration. This linear relationship between distance and tangential acceleration is an important principle used throughout this thesis: by measuring a_{tan} and α, the distance r from the CoR can be estimated.

2.10.1 Yaw rotation

During yaw, the vessel rotates about the vertical z-axis. The yaw-induced tangential acceleration lies in the horizontal plane. With the coordinate system used in this thesis, a positive yaw motion turns the vessel to port. A sensor located forward of the CoR therefore accelerates toward port, while a sensor located to starboard accelerates forward. The tangential acceleration components are:

$$a_x = b \cdot \alpha_z \tag{2.15}$$

$$a_y = -l \cdot \alpha_z \tag{2.16}$$

where l is the longitudinal distance from the CoR, b is the lateral distance from the CoR and α_z is the yaw angular acceleration. For a sensor on the vessel centerline, $b = 0$, so the yaw-induced tangential acceleration appears only on the lateral axis.

Gravity acts vertically and therefore does not project onto the horizontal measurement axes.

2.10.2 Roll rotation

When the vessel tilts during roll motion, gravity projects onto the lateral axis. Consider a sensor located at a vertical height h above the roll CoR. When the vessel rolls by an angle $\varphi(t)$, the sensor traces a circular arc of radius h. The accelerometer measures the specific force, and the sway-axis reading consists of two terms:

$$f_y(t) = h\,\ddot{\varphi}(t) - g\sin\Big(\varphi(t)\Big) \tag{2.17}$$

where $h\,\ddot{\varphi}(t)$ is the tangential acceleration from circular motion (the signal that carries the height information) and $g\sin(\varphi(t))$ is the projection of gravity onto the tilted sway axis (a disturbance that does not depend on h).

2.11 Ordinary least squares

When estimating physical parameters from real-world sensor data, the measurements contain noise. To find a linear relationship between two measured variables, such as tangential acceleration and angular acceleration, Ordinary Least Squares (OLS) regression can be used.

OLS is a statistical method that estimates the unknown parameters in a linear model. If the theoretical relationship between an independent variable x and a dependent variable y is modeled as a straight line, OLS finds the line of best fit by minimizing the sum of the squared residuals. A residual is the vertical difference between an actual measured data point and the corresponding predicted value on the line.

In the context of this thesis, OLS is applied to calculate the slope of the linear relationship between different acceleration signals. As shown in Section 2.10, this slope corresponds directly to the physical distance to the CoR.

3

Methods

This chapter describes the proposed calibration method and its validation. The chapter first presents the calibration pipeline and its four steps, the sensitivity analysis setup, implementation details and the experimental setups for both the test rig and marine environments. It also covers the estimation of the CoR of the vessel, as well as the fine calibration. To validate the pipeline, the method was evaluated in three progressively more challenging environments: (1) simulated data consisting of synthetic signals with known ground truth, first under ideal conditions and then with added sensor and process noise, (2) a motorized test rig that produces known yaw and roll oscillations at a measured distance from the CoR and (3) a 13 meter test vessel at sea with natural environmental disturbances.

3.1 Calibration pipeline overview

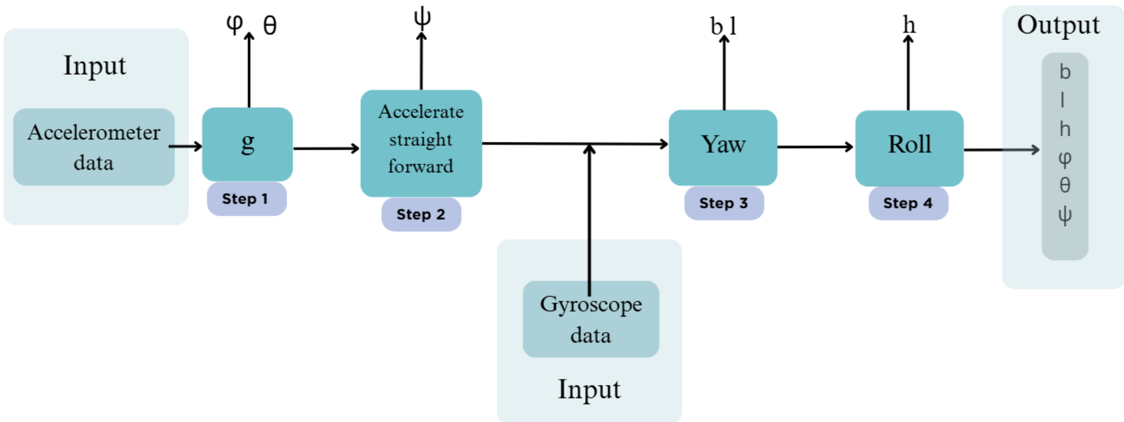

Figure 3.1: Illustration of the four-step calibration pipeline. Inputs are accelerometer and gyroscope data and the output is the 6-DOF pose relative to the CoR.

Figure 3.1 illustrates the calibration pipeline, which operates sequentially in four steps. The coordinate system used in this project is defined according to Figure 2.1 and described in Section 2.1. The input consists of time-series data from the gyroscope and accelerometer of a LiDAR's internal IMU. The output is the 6-DOF pose relative to the CoR. The CoR could, in principle, be provided by the vessel manufacturer. In this project, the CoR was not known beforehand and was therefore estimated using two IMUs, as described in Section 3.6.

The pipeline estimates the pose of each LiDAR relative to the vessel's CoR, as conceptually illustrated in Figure 3.2, using data recorded during the same vessel maneuvers. By linking all sensors to this common origin, the relative poses between the individual units are consequently determined.

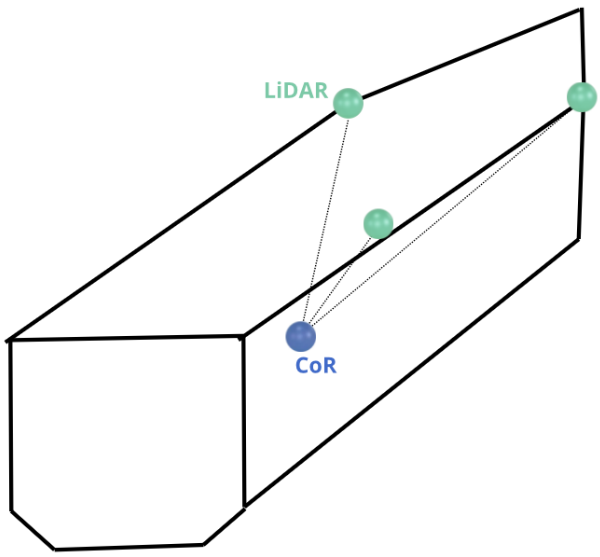

Figure 3.2: Conceptual illustration of multiple LiDAR positions and their relative distances to the vessel's CoR.

The four calibration steps are summarized in Algorithm 1 and described in detail in Sections 3.2–3.5. The final output is the 6-DOF pose $(b, l, h, \varphi, \theta, \psi)$ of the IMU inside the LiDAR relative to the vessel's CoR.

Algorithm 1 Four-step calibration pipeline

System Input: Accelerometer data, gyroscope data
System Output: 6-DOF pose relative the CoR $(b, l, h, \varphi, \theta, \psi)$

Step 1: Roll and pitch estimation
Input: Accelerometer data
Estimate the roll (φ) and pitch (θ) offsets using the measured gravity vector
Output: φ, θ

Step 2: Yaw estimation
Input: Accelerometer data, φ, θ
Accelerate the vessel in a straight line to estimate the yaw offset (ψ)
Output: ψ

Step 3: Horizontal position estimation
Input: Accelerometer data, gyroscope data, φ, θ, ψ
Execute a yaw-induced maneuver to estimate the lateral (b) and longitudinal (l) offsets
Output: b, l

Step 4: Vertical position estimation
Input: Accelerometer data, gyroscope data, φ, θ, ψ, b
Execute a roll-induced maneuver to estimate the vertical height offset (h)
Output: h

After completing the four pipeline steps, the output serves as an initial estimate of the 6-DOF parameters. This estimate is then refined in a final fine calibration step to reduce residual errors. The fine calibration is performed using the GICP algorithm described in Section 2.7. Because GICP relies on a sufficiently accurate initial guess to avoid converging into local minima, the estimation provided by the pipeline is essential for accurate calibration. This final step uses LiDAR point cloud data, which requires an overlapping FoV with at least one other LiDAR.

3.2 Step 1: Orientation calibration (roll and pitch)

The purpose of this step is to determine the roll and pitch offset angles required to align the sensor's coordinate system with the boat's coordinate system. The gravity vector is used to accomplish this.

An accelerometer measures the normal force acting upon it. When the vessel is stationary, the accelerometer measures the gravity vector. Therefore, the accelerometer measures an upward acceleration vector of $1g$ along the Z-axis of the global frame:

$$g_W = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{3.1}$$

When the sensor is physically rotated by a pitch angle (θ) and a roll angle (φ) (see Figure 3.3), the gravity vector is projected onto the sensor's own axes (X_S, Y_S, Z_S). Yaw rotation does not affect this projection, since the yaw axis (Z-axis) is parallel to the gravity vector. The rotation from the vessel's reference frame to the sensor frame is defined using Euler angles in the ZYX sequence (yaw, pitch, roll). We compute the offset angles by projecting the average measured acceleration vector, $\bar{a} = [\bar{a}_x, \bar{a}_y, \bar{a}_z]$, onto the boat's horizontal plane. The angles are then derived by setting the condition that the average horizontal acceleration must be zero when the boat is not actively accelerating.

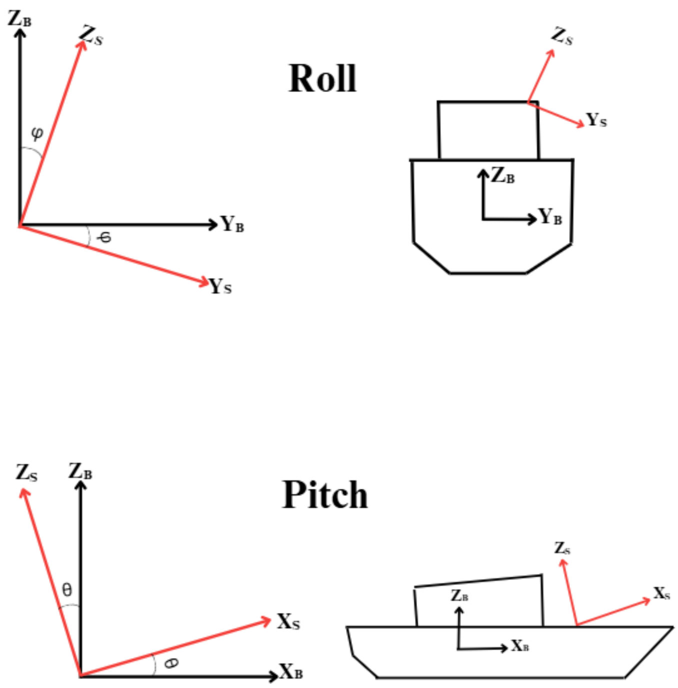

Figure 3.3: Definition of the roll (φ) and pitch (θ) offset angles between the boat's reference frame (black) and the sensor's reference frame (red).

3.2.1 Setup and physics

For this calibration step, the vessel must be either stationary or moving at a constant speed without turning. Under these conditions, the vessel experiences bounded motion, meaning it may oscillate due to environmental factors such as waves, but it does not continuously accelerate in any specific direction.

As noted by Vinande et al., pitch and roll mounting angles can be accurately estimated when a vehicle is stationary or moving without acceleration on a horizontal surface [2].

For bounded vessel motion, the net change in velocity over a sufficiently long time interval T is assumed to be negligible. Mathematically, the mean acceleration related to the vessel's physical motion is defined as:

$$\bar{\mathbf{a}}_{motion} = \frac{1}{T}\int_0^T \mathbf{a}_{motion}(t)\,dt = \frac{\mathbf{v}(T) - \mathbf{v}(0)}{T} \tag{3.2}$$

As the integration time T approaches infinity, the term regarding the change in velocity over time tends toward zero because the velocity $\mathbf{v}(t)$ is bounded. Therefore, the expected value of the kinematic acceleration becomes zero:

$$E[\mathbf{a}_{motion}] = \lim_{T\to\infty} \frac{1}{T}\int_0^T \mathbf{a}_{motion}(t)\,dt = \mathbf{0} \tag{3.3}$$

This implies that the average acceleration caused by the vessel's actual motion is effectively zero. Consequently, the only non-zero component remaining in the long-term average is the measured acceleration due to gravity:

$$\bar{\mathbf{a}}_{measured} \approx \mathbf{g}_{projected} \tag{3.4}$$

The vessel essentially acts as a dynamic system oscillating around a stable equilibrium. Even if the vessel experiences roll, pitch or yaw disturbances from waves during the data collection, the mean acceleration value will converge to the static gravity vector, provided the recording time T is sufficiently long.

From a signal processing perspective, the constant gravity vector acts as a direct current (DC) signal with a frequency of 0 Hz. Calculating the mean value of the accelerations is therefore analogous to applying an ideal low-pass filter, which extracts the static DC offset representing gravity while filtering out all higher-frequency motions.

3.2.2 Pitch offset estimation

This subsection addresses the calculation of the pitch offset angle (θ_{offset}), which represents the sensor's rotation around the lateral Y-axis. As illustrated in Figure 3.4, a positive pitch angle corresponds to a nose-up orientation of the vessel.

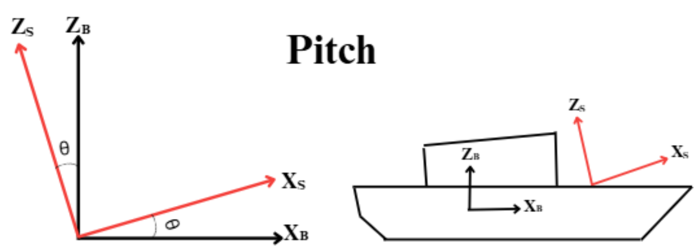

Figure 3.4: Vessel seen from the starboard side. The coordinate axes of the boat (B) and the sensor (S) are misaligned around the Y-axis, illustrating a pitch offset.

Assuming temporarily that the sensor experiences no roll ($\varphi = 0$), the measured acceleration can be projected onto the vessel's longitudinal axis (X_B) as follows:

$$a_{X_B} = \bar{a}_x \cos\theta - \bar{a}_z \sin\theta \tag{3.5}$$

During this calibration step, the true longitudinal acceleration of the vessel is zero, as the boat is either stationary or traveling at a constant speed. Setting a_{X_B} to zero gives:

$$\bar{a}_x \cos\theta - \bar{a}_z \sin\theta = 0 \implies \tan\theta = \frac{\bar{a}_x}{\bar{a}_z} \tag{3.6}$$

However, this simplified model only holds true if the sensor is perfectly level laterally. If the IMU is simultaneously subjected to roll ($\varphi \neq 0$), gravity projects a force along both the sensor's Y and Z axes. Relying only on $\bar{a}_z$ would underestimate the total vertical gravitational force. Because this term acts as the denominator in the pitch calculation, an underestimation would lead to an exaggerated pitch angle.

To ensure an accurate pitch calculation independent of the roll angle, the total vertical gravity vector must be determined. This is achieved by combining the Y and Z axis measurements using the Pythagorean theorem ($\sqrt{\bar{a}_y^2 + \bar{a}_z^2}$). Furthermore, using the four-quadrant arctangent function (arctan2) ensures the correct sign of the angle and prevents division-by-zero errors.

Consequently, the formula for the pitch offset is as follows:

$$\theta_{\text{offset}} = \text{arctan2}\left(\bar{a}_x, \sqrt{\bar{a}_y^2 + \bar{a}_z^2}\right) \tag{3.7}$$

3.2.3 Roll offset estimation

This subsection addresses the calculation of the roll offset angle (φ_{offset}), which represents the sensor's misalignment around the longitudinal X-axis. As illustrated in Figure 3.5, a positive roll angle is defined as a rotation where the starboard side of the vessel tilts downwards.

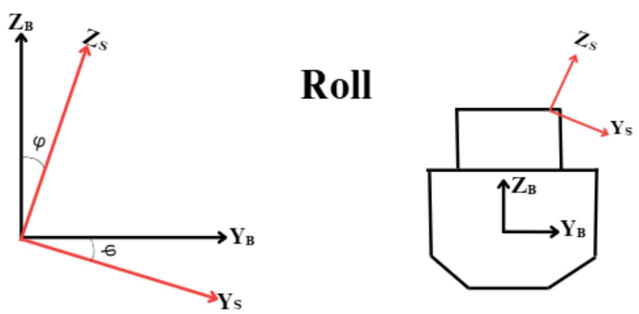

Figure 3.5: View from behind the vessel. The coordinate axes of the boat (B) and the sensor (S) are misaligned around the X-axis, illustrating a roll offset.

To determine this offset, the sensor's acceleration measurements must be projected onto the vessel's lateral axis (Y_B). Based on the geometric relationship between the misaligned frames, the lateral acceleration is expressed as:

$$a_{Y_B} = \bar{a}_y \cos\varphi + \bar{a}_z \sin\varphi \tag{3.8}$$

During this specific calibration step, the vessel is required to be stationary or moving with a constant velocity, without turning. Consequently, the true lateral acceleration of the vessel is zero ($a_{Y_B} = 0$). Substituting this into the equation gives:

$$0 = \bar{a}_y \cos\varphi + \bar{a}_z \sin\varphi \tag{3.9}$$

By rearranging the terms, the relationship between the measured accelerations and the roll angle can be isolated:

$$-\bar{a}_z \sin\varphi = \bar{a}_y \cos\varphi \implies \tan\varphi = -\frac{\bar{a}_y}{\bar{a}_z} \tag{3.10}$$

The final formula for the roll offset is therefore:

$$\varphi_{\text{offset}} = \text{arctan2}\left(-\bar{a}_y, \bar{a}_z\right) \tag{3.11}$$

3.2.4 Derivation and application of rotation matrices

Once the pitch (θ_{offset}) and roll (φ_{offset}) mounting offsets have been determined, they must be applied to the raw 3-axis accelerometer and gyroscope data. This is done by constructing 3D rotation matrices that project the measurement vectors from the misaligned sensor frame into an intermediate "level" frame. This intermediate frame is horizontally aligned with the vessel's plane, although it remains uncalibrated in yaw.

Because the sensor is physically mounted with an angular offset (for example, tilted by φ_{offset}), the data it produces is systematically offset by that same angle. To correct this measurement and align the data with the true coordinate system of the vessel, the measured vectors must be mathematically rotated in the opposite direction, corresponding to a rotation of $-\varphi_{\text{offset}}$.

To understand the mechanics of these correction matrices, the rotation matrix for the roll offset is sequentially derived below.

3.2.4.1 Derivation step 1: Deriving a standard 2D rotation via basis vectors

To construct a standard mathematical rotation matrix in a 2D plane (e.g., the YZ-plane), the derivation relies on a fundamental property of linear algebra.

Consider the unit vectors aligned with the original Y and Z axes:

$$\mathbf{e}_y = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{e}_z = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.12}$$

When rotating this coordinate system counter-clockwise by an arbitrary angle α, the new coordinates of the basis vectors must be determined. This transformation is illustrated in Figure 3.6 and are as follows:

The unit vector $\mathbf{e}_y$ rotates to the new position $\mathbf{e}'_y$, resulting in an extension of $\cos(\alpha)$ along the Y-axis and $\sin(\alpha)$ along the Z-axis:

$$\mathbf{e}'_y = \begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix} \tag{3.13}$$

The unit vector $\mathbf{e}_z$ rotates to the new position $\mathbf{e}'_z$. Because its starting position is perpendicular to the Y-axis, its new angle is $(90° + \alpha)$. Applying trigonometric identities $(\cos(90° + \alpha) = -\sin(\alpha)$ and $\sin(90° + \alpha) = \cos(\alpha))$ gives:

$$\mathbf{e}'_z = \begin{bmatrix} -\sin(\alpha) \\ \cos(\alpha) \end{bmatrix} \tag{3.14}$$

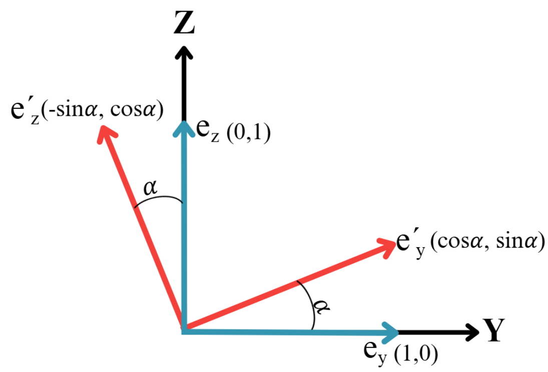

Figure 3.6: Derivation of the rotation matrix in the YZ-plane. The original basis vectors $\mathbf{e}_y, \mathbf{e}_z$ (blue) are rotated by α to $\mathbf{e}'_y, \mathbf{e}'_z$ (red), forming the columns of the rotation matrix.

By assembling these newly rotated basis vectors as the columns of a matrix, the standard 2D rotation matrix for the angle α is:

$$R_{standard} = \begin{bmatrix} \mathbf{e}'_y & \mathbf{e}'_z \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{3.15}$$

Applying this matrix to an arbitrary measurement vector $\mathbf{v} = [y, z]^T$ gives its new rotated coordinates in the specified plane.

3.2.4.2 Derivation step 2: Applying the inverse rotation

As previously shown, to correct for a physical mounting offset of φ_{offset}, the measured data must be mathematically rotated by the inverse angle, $-\varphi_{\text{offset}}$.
By substituting the rotation angle in the standard matrix with $-\varphi_{\text{offset}}$, and applying the trigonometric identities $\cos(-\varphi) = \cos(\varphi)$ and $\sin(-\varphi) = -\sin(\varphi)$, the negative signs are shifted. This results in the 2D correction matrix:

$$R_{correction} = \begin{bmatrix} \cos(-\varphi_{\text{offset}}) & -\sin(-\varphi_{\text{offset}}) \\ \sin(-\varphi_{\text{offset}}) & \cos(-\varphi_{\text{offset}}) \end{bmatrix} = \begin{bmatrix} \cos(\varphi_{\text{offset}}) & \sin(\varphi_{\text{offset}}) \\ -\sin(\varphi_{\text{offset}}) & \cos(\varphi_{\text{offset}}) \end{bmatrix} \tag{3.16}$$

3.2.4.3 Derivation step 3: Expanding to 3D

Roll is defined as a rotation around the X-axis. Therefore, the physical X-coordinate is completely unaffected by roll, meaning the equation for the new X-axis is simply $x_{new} = x$. Combining this invariant X-axis with the 2D correction matrix for the YZ-plane gives the final 3×3 roll rotation matrix (R_{roll}):

$$R_{roll} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi_{\text{offset}}) & \sin(\varphi_{\text{offset}}) \\ 0 & -\sin(\varphi_{\text{offset}}) & \cos(\varphi_{\text{offset}}) \end{bmatrix} \tag{3.17}$$

Similarly, pitch is a rotation around the Y-axis by θ_{offset}, meaning the Y-coordinate behaves invariantly. Applying the exact same inverse rotation logic to the XZ-plane gives the pitch rotation matrix (R_{pitch}):

$$R_{pitch} = \begin{bmatrix} \cos(\theta_{\text{offset}}) & 0 & -\sin(\theta_{\text{offset}}) \\ 0 & 1 & 0 \\ \sin(\theta_{\text{offset}}) & 0 & \cos(\theta_{\text{offset}}) \end{bmatrix} \tag{3.18}$$

It should be noted that a yaw rotation matrix (R_{yaw}) is not derived or applied at this stage. Yaw represents a rotation around the Z-axis, which is parallel to the gravity vector. Consequently, yaw misalignment cannot be observed using static gravity measurements. The yaw offset is instead calculated and corrected in Step 2 of the calibration pipeline.

3.2.4.4 Derivation step 4: Applying the corrections

To correct the raw sensor data, the 3D measurement vectors for acceleration ($\mathbf{a}_{raw}$) and angular rate ($\boldsymbol{\omega}_{raw}$) are multiplied by these newly derived rotation matrices. Assuming the sequence of rotations is roll followed by pitch, the full correction step for every time sample is formulated as:

$$\begin{bmatrix} a_{x,\text{level}} \\ a_{y,\text{level}} \\ a_{z,\text{level}} \end{bmatrix} = R_{pitch} R_{roll} \begin{bmatrix} a_{x,\text{raw}} \\ a_{y,\text{raw}} \\ a_{z,\text{raw}} \end{bmatrix} \tag{3.19}$$

$$\begin{bmatrix} \omega_{x,\text{level}} \\ \omega_{y,\text{level}} \\ \omega_{z,\text{level}} \end{bmatrix} = R_{pitch} R_{roll} \begin{bmatrix} \omega_{x,\text{raw}} \\ \omega_{y,\text{raw}} \\ \omega_{z,\text{raw}} \end{bmatrix} \tag{3.20}$$

These leveled signals ($\mathbf{a}_{level}$ and $\boldsymbol{\omega}_{level}$) are now correctly aligned with the vessel's horizontal plane. They are subsequently used as the necessary inputs for all calculations in Steps 2, 3 and 4.

3.3 Step 2: Orientation calibration (yaw)

This step calculates the yaw offset, which is the misalignment between the sensor's forward axis and the vessel's actual direction of travel. Because gravity acts in the vertical direction, it cannot be used to determine heading misalignments. Instead, dynamic kinematic constraints must be used. As established in sensor calibration literature by Vinande et al. [2], once pitch and roll mounting angles are determined, they can be used to estimate the yaw mounting angle by relying on a known forward acceleration. However, with the geometric method presented below, the exact magnitude of the forward acceleration does not need to be known. It strictly needs to be non-zero, as the magnitude ultimately cancels out during the derivation.

To achieve this, the vessel must accelerate, or decelerate, in a straight line. If the vessel were to travel at a constant velocity, the kinematic acceleration in the horizontal plane would be zero. Under such conditions, the accelerometer would only register gravity and noise, leaving the system with no reference signal to distinguish the forward direction from the lateral direction.

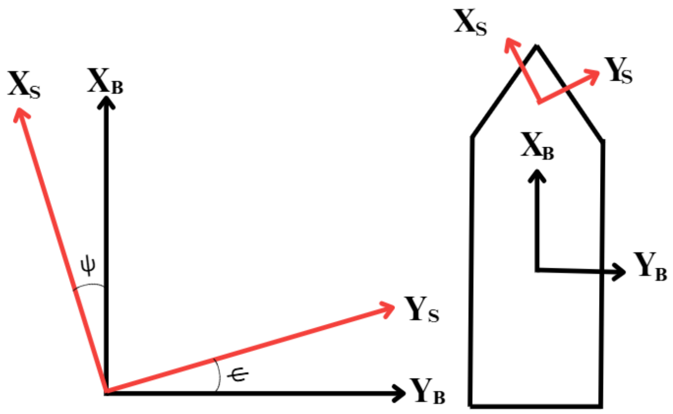

Figure 3.7: Top-down view of the yaw misalignment ψ. The coordinate axes of the vessel (B) and the sensor (S) are misaligned in the horizontal plane.

The relationship between the vessel's coordinate axes (surge X_B, sway Y_B), the sensor's axes (X_S, Y_S), and the unknown yaw offset angle ψ is defined by a 2D rotation, and the vectors are illustrated in Figure 3.7. When projecting the measured accelerations from the sensor's frame (a_x, a_y) onto the vessel's frame (a_{X_B}, a_{Y_B}), the relationship is expressed as:

$$a_{X_B} = a_x \cos(\psi) + a_y \sin(\psi) \tag{3.21}$$

$$a_{Y_B} = -a_x \sin(\psi) + a_y \cos(\psi) \tag{3.22}$$

During this specific calibration step, the vessel is accelerated in a strictly straight line. Therefore, the true acceleration in the lateral sway direction of the vessel is zero ($a_{Y_B} = 0$). Substituting this constraint into Equation 3.22 gives:

$$0 = -a_x \sin(\psi) + a_y \cos(\psi) \tag{3.23}$$

Rearranging the terms:

$$a_x \sin(\psi) = a_y \cos(\psi) \tag{3.24}$$

By dividing both sides by $a_x \cos(\psi)$, the tangent of the offset angle can be isolated:

$$\tan(\psi) = \frac{a_y}{a_x} \implies \psi_{\text{offset}} = \arctan\left(\frac{a_y}{a_x}\right) \tag{3.25}$$

To ensure the angle is correctly resolved across all four quadrants and to prevent division-by-zero errors arctan2 is used. The final formula for the yaw offset is therefore:

$$\psi_{\text{offset}} = \text{arctan2}(a_y, a_x) \tag{3.26}$$

This formulation computes the yaw offset angle using only the ratio of the measured lateral and longitudinal accelerations, rendering the calculation independent of the actual acceleration magnitude.

While it is common for a vessel to pitch upwards (raise its bow) during forward acceleration, this dynamic behavior does not affect our 2D yaw calculation. The pitching motion introduces a gravity component along the vessel's longitudinal axis (X_B) and vertical axis (Z_B). As long as the vessel maintains a straight trajectory, no lateral forces are introduced ($a_{Y_B} = 0$). Consequently, the pitch only alters the total magnitude of the acceleration measured along the forward axis. When this longitudinal vector is projected onto the sensor's misaligned axes, both measured horizontal components (a_x and a_y) scale proportionally. Because arctan2 calculates the angle based strictly on the ratio between these two components, the total acceleration magnitude cancels out. Therefore, the yaw calculation works despite the vessel tilting upwards.

3.3.0.1 Applying the yaw correction

Once the offset angle ψ_{offset} has been determined, it must be applied to correct the sensor's measurements to ensure that subsequent steps operate within the correct coordinate frame. Because yaw is a rotation around the Z-axis, the Z-coordinate remains unaffected. Expanding the 2D projection into a 3D rotation matrix (R_{yaw}) results in:

$$R_{yaw} = \begin{bmatrix} \cos(\psi_{\text{offset}}) & \sin(\psi_{\text{offset}}) & 0 \\ -\sin(\psi_{\text{offset}}) & \cos(\psi_{\text{offset}}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.27}$$

To align the sensor's kinematic measurements, the leveled 3D vectors from Step 1 are multiplied by this yaw rotation matrix:

$$\begin{bmatrix} a_{x,corrected} \\ a_{y,corrected} \\ a_{z,corrected} \end{bmatrix} = R_{yaw} \begin{bmatrix} a_{x,level} \\ a_{y,level} \\ a_{z,level} \end{bmatrix} \tag{3.28}$$

These corrected signals are now aligned with the vessel's longitudinal, lateral and vertical axes, which enables the position estimations in Steps 3 and 4.

3.4 Step 3: Horizontal position calibration

This calibration step, along with Step 4, is based on the frequency-domain positioning method introduced by Lund [26] and detailed in Section 1.1.13. In this thesis, the same principle is implemented by estimating the lateral and longitudinal offsets directly from the surge and sway velocity components at the reference frequency. This step is conducted by actively maneuvering the vessel in a periodic yaw motion to determine the horizontal position of the sensor relative to the vessel's CoR. This position is defined by the lateral offset (b) and the longitudinal offset (l), as illustrated in Figure 3.8. This step assumes that the IMU signals have already been rotated into the vessel frame using the orientation corrections from Steps 1 and 2.

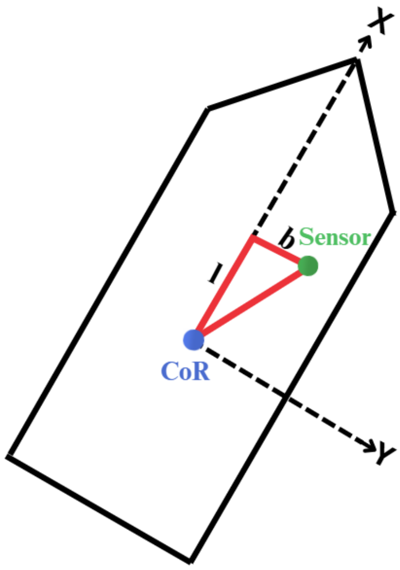

Figure 3.8: Top-down view illustrating the sensor's horizontal position relative to the CoR, showing the longitudinal offset (l) and the lateral offset (b).

By maintaining a consistent periodic yaw rate, the kinematic energy of the maneuver is concentrated into a distinct, narrow frequency band. This periodicity maximizes the signal-to-noise ratio (SNR), making sure that the resulting motion stands out against noise when transformed into the frequency domain.

To isolate the relevant dynamics and determine these parameters, the time-series data from the accelerometer and gyroscope are analyzed in the frequency domain. The raw signals are first passed through a bandpass filter. This filtering step eliminates high-frequency noise as well as low-frequency drifts. The specific low-cut and high-cut frequencies were determined based on the period of the executed maneuver to ensure that the fundamental frequency of the motion was captured within the passband. Consequently, a maneuver with a longer period corresponds to lower cutoff frequencies, and vice versa.

Subsequently, an FFT is applied to the filtered data to convert the signals into the frequency domain. By identifying the reference frequency (f_{ref}) where the yaw motion shows the highest spectral power, the dominant yaw rate amplitude (A_{ω_z}) is extracted.

Since the IMU measures acceleration rather than velocity, the surge and sway signals are converted to velocity directly in the frequency domain. At the reference frequency, each acceleration amplitude is divided by $2\pi f_{ref}$ and each phase is shifted by $-\frac{\pi}{2}$. This gives the surge and sway velocity components without drift from time-domain integration.

3.4.1 Kinematic derivation

By integrating the yaw-induced tangential acceleration components in Section 2.10.1, the corresponding velocity components are obtained. In the implemented pipeline, the corrected signals are expressed in the vessel frame, so the horizontal motion in Step 3 is described by:

$$v_x = \omega_z b \tag{3.29}$$

$$v_y = -\omega_z l \tag{3.30}$$

where v_x is the surge velocity, v_y is the sway velocity, ω_z is the yaw rate about the vessel's vertical axis, b is the lateral offset and l is the longitudinal offset.

In the frequency domain, the magnitude of each offset is obtained from the amplitude ratio between the corresponding velocity component and the yaw rate amplitude at f_{ref}:

$$|b| = \frac{A_{v_x}}{A_{\omega_z}} \tag{3.31}$$

$$|l| = \frac{A_{v_y}}{A_{\omega_z}} \tag{3.32}$$

The signs are then determined from the phase differences between each velocity component and the yaw rate. Let $\Delta\phi_x = \phi_{v_x} - \phi_{\omega_z}$ and $\Delta\phi_y = \phi_{v_y} - \phi_{\omega_z}$. The final offsets are:

$$b = \frac{A_{v_x}}{A_{\omega_z}} \cos(\Delta\phi_x) \tag{3.33}$$

$$l = -\frac{A_{v_y}}{A_{\omega_z}} \cos(\Delta\phi_y) \tag{3.34}$$

3.5 Step 4: Vertical position calibration

The final step of the calibration pipeline is determining the vertical offset (h) of the sensor relative to the vessel's CoR. This calibration step is conducted by isolating the rolling motion of the vessel, which is achieved either by actively inducing roll or by natural roll caused by waves.

Every vessel has a natural roll frequency, which is the specific rate at which it naturally oscillates side-to-side. When the vessel is induced to roll at this exact frequency, either by wave action or active steering maneuvers, the motion is amplified due to resonance. This generates a clear signal that is distinguishable from background noise in the frequency domain.

To isolate this motion, the time-domain IMU data is first passed through a bandpass filter, similar to the procedure in Step 3, to filter out high-frequency noise and slow drifts. A FFT is then computed for these bandpass-filtered signals and the reference frequency (f_{ref}) is identified at the peak amplitude of the roll signal spectrum. This step assumes that the IMU signals have already been rotated into the vessel frame using the orientation corrections from Steps 1 and 2.

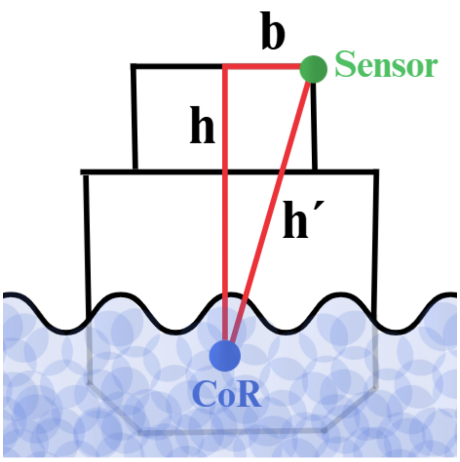

Figure 3.9: Rear view of the vessel illustrating the sensor's vertical position relative to the CoR, showing the vertical offset (h), the lateral offset (b) and the intermediate distance (h').

The method described in 1.1.13 calculates the vertical offset based on position amplitudes, namely the sway displacement (A_y) and the roll angle (A_φ). The intermediate lever arm h' (seen in Figure 3.9) is determined by this equation:

$$h' = \sqrt{\frac{4{A_y}^2 \cdot \sin^2(90^\circ - A_\varphi)}{\sin^2(2A_\varphi)} + b^2} \tag{3.35}$$

where b is the lateral position offset determined in Step 3.

This equation can be simplified using standard trigonometric identities: $\sin(90° - A_\varphi) = \cos(A_\varphi)$ and $\sin(2A_\varphi) = 2\sin(A_\varphi)\cos(A_\varphi)$. Substituting these into the formula gives:

$$h' = \sqrt{\frac{4{A_y}^2 \cdot \cos^2(A_\varphi)}{4\sin^2(A_\varphi)\cos^2(A_\varphi)} + b^2} = \sqrt{\frac{{A_y}^2}{\sin^2(A_\varphi)} + b^2} \tag{3.36}$$

The geometric relation between the intermediate distance h', the vertical offset h and the lateral offset b is

$$h = \sqrt{(h')^2 - b^2} \tag{3.37}$$

Substituting the reduced form of h' into this expression makes the b term cancel, which gives

$$h = \frac{A_y}{\sin(A_\varphi)} \tag{3.38}$$

This displacement-based formulation was used in the simulated pipeline. When the method was later applied to physical IMU data, it was replaced by a velocity-based formulation with gravity compensation, as described in Section 3.8.

3.6 Center of rotation estimation

The calibration pipeline requires the CoR as a reference point. If this point is not known beforehand it can be estimated experimentally using two IMUs mounted along the centerline at different longitudinal and vertical positions. This section covers this method.

3.6.1 Yaw CoR: two-accelerometer differential method

During yaw rotation, tangential acceleration appears on the lateral axis of each sensor. If two sensors are mounted at distances r_1 and r_2 from the same rotation center separated by a known distance $L = |r_1 - r_2|$, their tangential accelerations are:

$$a_1 = r_1 \cdot \alpha, \quad a_2 = r_2 \cdot \alpha \tag{3.39}$$

By subtracting them a differential signal is formed:

$$a_1 - a_2 = L \cdot \alpha \tag{3.40}$$

This differential cancels any common acceleration such as translational accelerations from drift or sway motion and isolates the rotational component. If the yaw CoR lies between the two sensors their lateral signals will be in anti-phase. The distance from sensor 1 to the CoR can then be found by regressing a_1 against $(a_1 - a_2)$:

$$a_1 = \frac{r_1}{L} \cdot (a_1 - a_2) \tag{3.41}$$

The slope of this regression gives a coefficient $k = r_1/L$ meaning $r_1 = k \cdot L$ can be calculated without needing to explicitly derive the angular acceleration α.

3.6.2 Roll CoR: two-sensor common-mode correction

For roll motion the two-sensor ratio approach is insufficient if both sensors are located on the same side of the CoR. In this configuration their tangential accelerations are in phase, making the differential small and sensitive to a common-mode bias.

A sensor on a rolling vessel does not only experience rotational acceleration. It also experiences a shared lateral acceleration $C(t)$ that affects all points on the vessel equally:

$$a_i(t) = r_i \cdot \alpha(t) + C(t) \tag{3.42}$$

If $C(t)$ is correlated with the angular acceleration $\alpha(t)$, an OLS regression of a_i on α becomes biased:

$$r_{i,\text{raw}} = r_{i,\text{true}} + \beta, \quad \beta = \frac{\sum \alpha(t)\, C(t)}{\sum \alpha(t)^2} \tag{3.43}$$

Because the two sensors share the same $C(t)$ their mean acceleration is:

$$\frac{a_1(t) + a_2(t)}{2} = \frac{r_1 + r_2}{2}\, \alpha(t) + C(t) \tag{3.44}$$

To apply this correction the angular acceleration α is first obtained by differentiating the gyroscope roll rate $(d\omega/dt)$. Individual OLS fits of a_i on α are computed for both sensors. The common-mode bias $\hat{\beta}$ is then estimated by regressing the sensor mean $(a_1 + a_2)/2$ on α and subtracted to find the true distance:

$$r_{i,\text{corr}} = r_{i,\text{raw}} - \hat{\beta} \tag{3.45}$$

3.7 Sensitivity analysis of pipeline with simulated data

To systematically evaluate the robustness and performance of the proposed 4-step calibration pipeline in sections 3.1-3.5, a number of sensitivity analyses were conducted using simulated data. Both the calibration pipeline and the data generation were implemented in Python. A Monte Carlo approach with 100 independent trials for all simulations was used, where varying levels of different parameters, including noise, recording duration, misalignment and maneuver intensity, were introduced to quantify their impact on the estimation error. For each configuration, the mean absolute error (MAE) and the standard deviation across the 100 trials were computed to evaluate performance. The setup for each analysis is detailed below.

3.7.1 Influence of sensor noise and placement on estimation precision

To evaluate the robustness of the calibration pipeline against sensor noise, Gaussian white noise with varying standard deviations was introduced to the simulated accelerometer and gyroscope signals. For the position estimation, the sensor distance parameters (b, l and h) relative to the vessel's CoR were configured to 2 meters and 15 meters. For each noise increment, the MAE and standard deviation for the lateral, longitudinal and vertical offset estimates were calculated. It should be noted that errors from the estimated orientation angles are not propagated into the positional errors of b, l and h.

In addition to the positional offsets, the robustness of the initial orientation estimation (φ, θ and ψ) was evaluated across the varying noise levels. Finally, to isolate the lever-arm effect, a separate simulation was performed with a fixed noise standard deviation of 0.01. In this test, the absolute estimation error was calculated and evaluated as a continuous function of the distance from the sensor to the CoR.

3.7.2 Influence of recording duration on estimation precision

Since the calibration steps rely on averaging over time and frequency-domain analysis, an important parameter is the duration of the data recording. To evaluate how the four pipeline steps perform with different recording times, a duration sensitivity analysis was conducted. To make the simulations more realistic, wave motion and engine vibration were added as process noise alongside the sensor noise.

The same simulation settings were used for all four calibration steps. The noise model included white sensor noise (0.01 m/s^2 and 0.01 rad/s), wave-induced process noise (0.5 m/s^2 at 0.3 Hz with corresponding 1° rotational amplitudes in steps 3 and 4) and engine vibrations (0.1 m/s^2 at 50 Hz). In reality, there are different noise levels for the accelerometer and gyroscope. Given that the effect of sensor noise is minor compared to process noise and that the sensor noise was chosen conservatively, the same noise level was used for both. The process noise parameters were selected within within reasonable limits. 45° angular offsets, and 1.0 m position offsets were used.

3.7.3 Error propagation between calibration steps

The proposed 4-Step calibration pipeline is sequential, meaning the position estimations in Steps 3 and 4 rely on accurate orientation estimations from Steps 1 and 2. To evaluate how angular errors propagate through the pipeline, an error propagation analysis was conducted.

Specifically, this analysis isolated the effect of an incorrect yaw rotation (resulting from Step 2) on the horizontal position calibration (Step 3). Synthetic data was generated with a true horizontal offset of 1.0 m laterally and 1.0 m longitudinally ($b_{\text{true}} = 1.0$ m, $l_{\text{true}} = 1.0$ m). Here, no process noise was added in order to isolate the projection error. The assumed orientation from Step 2 was then intentionally skewed with a yaw misalignment (ψ_{offset}) ranging from $-90°$ to $+90°$. The position estimation algorithm for Step 3 was subsequently executed for each misalignment angle to quantify the resulting positional error.

3.7.4 Effect of maneuver intensity on signal-to-noise ratio

In Steps 2, 3 and 4 a maneuver is performed in order to calibrate the sensor, where the intensity affects the SNR. In Step 1, the vessel is stationary or at constant speed and no physical maneuver intensity is of relevance. For Step 2 (yaw offset estimation), the magnitude of the forward acceleration was varied from 0.2 to 5.0 m/s^2. For Step 3 (horizontal positioning), the yaw amplitude was varied between 0.5 and 10 degrees. Finally, for Step 4 (vertical positioning), the roll amplitude was evaluated from 1 to 15 degrees.

In all tests, the maneuver duration was fixed at 60 seconds, and the true offset parameters were set to 45° for yaw and 1.0 m for the respective position offsets. The simulations included a noise model identical to the duration analysis. The mean estimation error was recorded for each intensity level.

3.8 Adaptations for physical sensor data

The calibration pipeline described in the previous section was developed and validated using simulated data. When applying it to real IMU data, several adaptations were required. This section describes these, which were integrated into the Python codebase and subsequently used to process the data from the physical testing in the following sections.

3.8.1 IMU axis identification and DIFOP transform

The raw IMU data from the RoboSense Airy is expressed in the chip's internal coordinate frame, which does not align with the coordinate system shown in the sensor's user manual. The correct transformation from chip axes to the LiDAR body frame is provided as a quaternion in the DIFOP data packet. This quaternion rotation is applied to all IMU samples before they enter the calibration pipeline.

3.8.2 Rodrigues rotation in Step 1

In the simulations, roll and pitch were computed sequentially using arctan2 expressions. However, this method causes a gimbal lock at a $\pm 90°$ pitch, where the roll and yaw axes become indistinguishable. Since the sensor was mounted nearly vertically on the test rig, this became a practical issue.

To handle any orientation, gravity alignment was instead computed using a Rodrigues rotation. This avoids Euler angles entirely by calculating the shortest rotation to align the measured gravity vector with the vertical axis. The result is a single 3×3 rotation matrix that works for any mounting angle without singularities.

3.8.3 Gravity compensation in Step 4

In the simulations, Step 4 used displacement amplitudes to estimate the vertical offset, following the patent equations directly. However, when using real accelerometer data, an extra issue had to be addressed. Gravity affects the sway measurements when the object tilts.

When the vessel rolls by an angle φ, the accelerometer measures a gravity component $g\sin(\varphi)$ along the sway axis. This component is not caused by translational motion and must be removed before estimating the vertical offset. For small roll angles, the gravity contribution is small relative to the kinematic signal. For larger angles or longer oscillation periods, the gravity term can become very large.

To compensate for this gravity component, the roll angle $\varphi(t)$ is first obtained by integrating the gyroscope roll rate. The gravity term $g\sin(\varphi(t))$ is then added back to the measured sway acceleration to recover the kinematic sway acceleration. After compensation, the body-frame relation becomes

$$a_y(t) = h\ddot{\varphi}(t) \tag{3.46}$$

Integrating once in the frequency domain gives

$$h = \frac{A_{v_y}}{A_{\dot{\varphi}}} \tag{3.47}$$

where A_{v_y} is the sway velocity amplitude and $A_{\dot{\varphi}}$ is the roll rate amplitude at f_{ref}. This velocity-based formulation avoids the extra integration required by the displacement-based method and is therefore less sensitive to low-frequency drift.

3.8.4 Whole-period trimming

The FFT assumes that the input signal contains whole periods. If the recording is cut in the middle of an oscillation cycle, the resulting discontinuity causes spectral leakage, which spreads energy across neighboring frequency bins and reduces the accuracy of the amplitude estimate.

To avoid this, the pipeline automatically detects the dominant oscillation frequency from the yaw rate (Step 3) or roll rate (Step 4) and then adjusts the end of the data window so that it contains an integer number of complete periods.

3.9 Test rig

To evaluate the method with real-world data, tests were conducted in two different environments. The first environment was an office setting, where a test rig was built to collect data and conduct the estimations in a controlled environment. The second was a real-world marine setting where data was collected onboard a 13 m vessel with natural environmental noise. The test rig is covered in this section, and the marine setting is covered in the following section.

The purpose of the office setup was to validate the proposed method in a controlled environment. To perform the yaw and roll motions required for Steps 3 and 4, a dedicated test rig was constructed, as shown in Figure 3.10. The rig consists of a wooden base equipped with a windshield wiper motor that drives an extended aluminum arm. A mounting plate is attached to the outer end of the arm, where the LiDAR, or IMU, is secured using hook-and-loop fasteners. The data collection procedure was divided into the following steps, with each step performed independently.

Figure 3.10: Test rig used for Steps 3 and 4, here oriented for Step 3 in its horizontal position.

3.9.1 Hardware and data acquisition

The testing was performed using a RoboSense Airy LiDAR with its internal IMU. A separate CPAC IMU was also tested for comparison. The RoboSense Airy transmits data over Ethernet at a sampling rate of 200 Hz. Recordings were captured using Wireshark and then parsed for the calibration pipeline. The CPAC IMU transmits data over a CAN bus. A PEAK USB dongle was used to access the CAN data, and recordings were made using PCAN-View. The CAN data was then parsed for the pipeline.

3.9.2 Step 1: Roll and pitch estimation

The setup was kept entirely stationary with the motor powered off. The LiDAR was placed flat on the plate and data was recorded for approximately two minutes. Since the rig is rotated 90° between Step 3 and Step 4, this step was done for both positions to get correct orientation offsets.

Although an inclinometer was intended to measure the tilt directly on the rig, the setup was too sensitive to movement to get consistent readings. Instead, the Step 1 code was validated on a stable office desk. By comparing the Step 1 estimation using the LiDAR's IMU data with the inclinometer reference measurements on this stable surface, the orientation calibration could be accurately verified. The inclinometer features a display resolution of 0.05° and a manufacturer-specified accuracy of ±0.1° when operating near 0°.

3.9.3 Step 2: Yaw estimation

The LiDAR was secured to a cart using hook-and-loop fasteners (see Figure 3.11). The cart was accelerated in a straight line for a few seconds, with a 25-meter cable supplying power to the sensor. Figures 3.11 and 3.12 show one configuration where the LiDAR dome faces the front of the cart and its cable points upwards. In this orientation, the expected yaw value is $0° \pm 3°$, as it is difficult to achieve perfect manual yaw alignment of the sensor. This was repeated with the LiDAR mounted at various rotated angles.

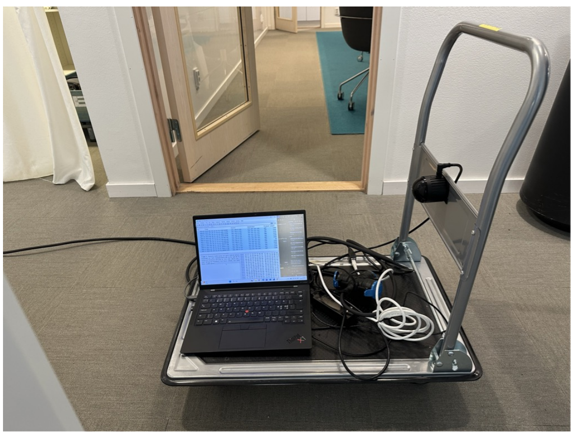

Figure 3.11: Setup for Step 2. The LiDAR sensor is secured to a rolling cart using hook-and-loop fasteners and powered through a long cable.

Figure 3.12: Close-up of the LiDAR sensor mounted on the cart for Step 2. The dome faces the front of the cart and the cable points upward, giving an expected yaw offset of $0° \pm 3°$.

3.9.3.1 Manual yaw angle for test rig

Because the test rig could not be accelerated in a straight line, Step 2 could not be performed as intended. Instead, the yaw angle was set manually. To simplify this process, the sensor was always mounted with one of its axes aligned with gravity. Consequently, the yaw offset could only take values in multiples of 90°. A helper script calculated the exact yaw angle based on the physical mounting orientation, using the forward-facing and upward-facing sides of the LiDAR as inputs. For example, if the bottom of the LiDAR faced forward and the left side faced upward, the manual yaw angle was set to $-90°$.

3.9.4 Step 3: Lateral and longitudinal estimation

The rig was placed horizontally on an even surface and secured using a screw clamp (see Figure 3.10). Step 1 was re-run for this rig orientation to calibrate roll and pitch because the sensor orientation on the rig differed from the flat-surface measurement. The sensor was oriented such that one of its axes was aligned with the lever arm. Following the procedure described in Section 3.9.3.1, the yaw angle was manually set to $-90°$.

A power supply unit was used to control the motor voltage, allowing the oscillation period to be adjusted. The motor was activated to rotate the arm at a specific frequency, creating yaw movements over a range of 83°. Data was recorded for approximately two minutes. The sensor was intentionally mounted to have a lateral offset of 0 cm. To verify the estimation, the physical longitudinal lever arm distance from the sensor to the center of rotation (the motor shaft) was measured using a ruler, with an estimated measurement accuracy of ± 0.5 cm. Multiple trials were conducted for Step 3 and Step 4, and representative datasets with stable oscillations were selected for the final analysis.

3.9.5 Step 4: Vertical estimation

The rig was secured vertically using a clamp and rotated 90 degrees compared to Step 3 so the motor created roll oscillations, see Figure 3.13. Step 1 was re-run for this rig orientation to calibrate roll and pitch. In this orientation one of the sensor axes was aligned with the gravity vector. Following the procedure described in Section 3.9.3.1 the yaw angle was manually set to $-90°$.

The motor voltage was regulated using a power supply unit to generate roll movements over an 83° range, and data was recorded for approximately two minutes. As in the previous step, a ruler was used to measure the true vertical distance to the center of rotation, with an estimated measurement accuracy of ± 0.5 cm.

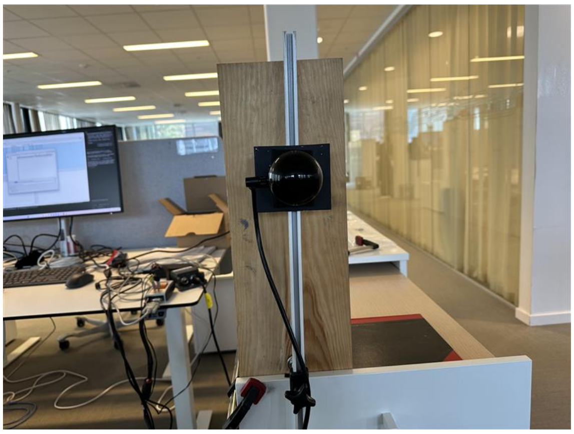

Figure 3.13: Test rig configured for Step 4. The rig is placed vertically, i.e. rotated 90 degrees compared to Step 3, so the motor creates roll oscillations.

3.9.6 Fine calibration verification

The output from steps 1 to 4 serves as the initial guess for the fine calibration using the GICP algorithm. The objective of this step is to determine whether this initial guess is sufficiently accurate for GICP to successfully converge and reduce the residual error relative to the true value.

For example, if the pipeline estimates a longitudinal length of 30 cm, but the true measured distance is 32 cm, it is necessary to verify whether GICP can correct this error. The goal is to find the tolerance limit of the initial guess and determine how inaccurate it can be before the algorithm gets trapped in a local minimum.

Here, the LiDAR point cloud data was used. The LiDAR was placed at an initial position to capture a reference scan and was then physically translated by known distances (ranging from 0.3 m to 5.0 m) to capture subsequent scans. For each distance, the GICP algorithm was tested by feeding it initial guesses with artificial errors ranging from 0% up to 80% relative to the true displacement. The evaluation procedure is summarized in Algorithm 2.

Algorithm 2 GICP Sensitivity to initial guess

Input: Source cloud $\mathcal{S}$ (moved), Target cloud $\mathcal{T}$ (reference)
Input: Ground truth displacement d_{GT}
Define Test Errors: $E = [0\%, 5\%, 10\%, \ldots, 80\%]$

Downsample $\mathcal{S}$ and $\mathcal{T}$ using a voxel grid and compute normals
for each error percentage $e \in E$ **do**
    Simulate inaccurate initial guess: $d_{init} \leftarrow d_{GT} \cdot (1 + e/100)$
    Run GICP registration using d_{init} as the starting point
    Extract the calculated displacement d_{result}
    Compute residual error: $\varepsilon \leftarrow \|d_{result} - d_{GT}\|$
    Record convergence status and RMSE
end for

3.10 Marine testing

Tests were conducted at sea using a 13 m test vessel provided by CPAC to evaluate the method under real-world conditions. Due to confidentiality, no pictures of the vessel are included. The calibration procedure followed the same four steps as the office setup, but the practical execution differed at sea.

3.10.1 Hardware and data acquisition

Three mechanical LiDAR units were used in this thesis, mounted on the test vessel in different positions and orientations. The setup included two RoboSense Airy sensors mounted at the bow (one on the starboard side and one on the port side) and one RoboSense Helios 32 (H32F70) positioned on the starboard cabin roof. The Airy models have a built-in IMU, while the Helios sensors do not. Since Steps 1-4 require IMU data, the Airy model was used in these steps. In the fine calibration step, both Airy and Helios sensors are used. The vessel's CoR was determined using two CPAC IMUs. The data acquisition process was similar to the office setup described in Section 3.9.1.

To evaluate the position estimates, the physical mounting positions of the sensors were measured, relative to the aft, lower deck floor and centerline, using a tape measure, with an estimated accuracy of ±10 cm. However, the true mounting orientation (roll, pitch and yaw) could not be accurately measured on the vessel. Because of this, the pipeline relies directly on the orientation estimates from Steps 1 and 2, without a separate ground truth validation. The marine testing for each step is detailed below.

3.10.2 Step 1: Roll and pitch estimation

The vessel was kept as stationary as possible with the engines turned off, although it was still affected by natural motions, like waves. Data was recorded for 120 seconds.

3.10.3 Step 2: Yaw estimation

The vessel performed an acceleration in a straight line in open water. The objective was to generate pure forward acceleration, ensuring that the mean acceleration in the surge direction was significantly greater than in the sway direction. Data was recorded for approximately 20-30 seconds, until the vessel reached its maximum speed.

3.10.4 Step 3: Lateral and longitudinal estimation

To generate yaw movements, the vessel's joystick was used. The onboard compass on the vessel was used as a reference to monitor the extent of the yaw rotations. Data was first recorded for 120 seconds while the vessel performed continuous alternating 30-degree rotations to the left and right. This exact procedure was then repeated with smaller alternating rotations of 10 degrees, see Figure 3.14.

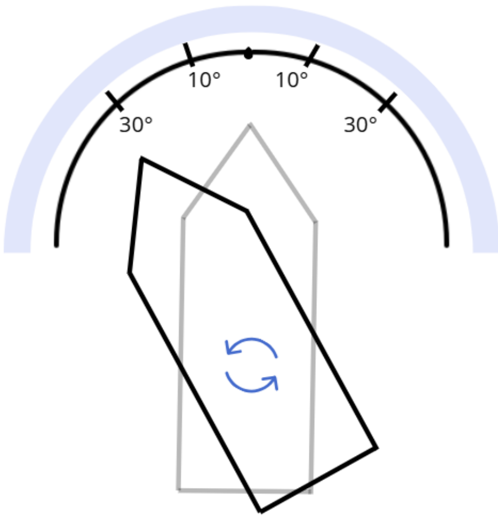

Figure 3.14: Illustration of the alternating 10° and 30° yaw rotations performed by the vessel.

3.10.5 Step 4: Vertical estimation

To induce continuous roll movements, the side thrusters were manually controlled by the joystick to find and match the vessel's natural roll frequency. By alternating thrust to the port and starboard sides, a resonant rolling motion was created and recorded for 120 seconds, see Figure 3.15.

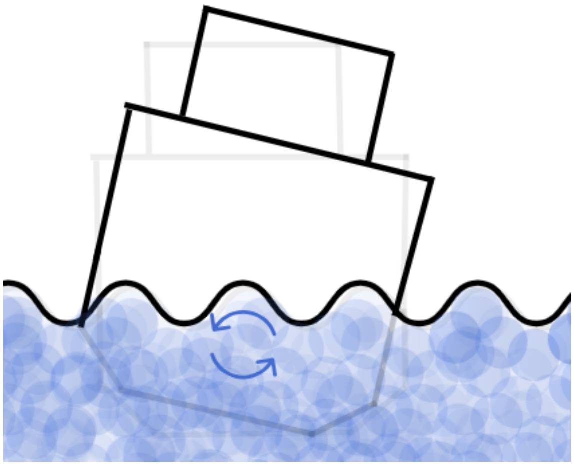

Figure 3.15: Illustration of the roll motion induced by alternating the side thrusters.

3.10.6 Fine calibration

While the office setup in Section 3.9.6 tested GICP convergence by displacing a single sensor, the marine fine calibration registers the point clouds of two separate sensors against each other. For this alignment to work two conditions must be met: the sensors must have an overlapping FOV and the overlapping region must contain geometric features.

As described in Section 3.10.1, the vessel was equipped with three LiDAR units. Recorded point cloud data from all sensors was available. A recording taken near a building on a pier was selected because it offered geometric features (walls, corners, windows and wooden docks) that both sensors could observe simultaneously. The vessel was positioned approximately 10 m from the building during the recording.

The raw pcap data was parsed into 3D point clouds using a Python parser based on the RoboSense rs-driver protocol. Individual sensors were distinguished by their UDP destination ports, which were identified beforehand using Wireshark.

The manually measured mounting poses are used to transform both clouds into the vessel coordinate system. The relative transform between the two sensors computed from these measured poses serves as the initial guess for GICP. Running GICP from this starting point produces a refined relative transform that represents the best achievable alignment given the point cloud data. This refined result is then used as the ground truth when evaluating whether GICP can also converge from a less accurate initial guess, such as one produced by the pipeline (Steps 1-4).

3.10.6.1 Overlap detection and cropping

Because the sensors point in different directions, most of their point clouds do not overlap. To avoid including points that have no counterpart in the other sensor's view, an overlap detection step is performed before running GICP.

Both point clouds are first transformed into the vessel coordinate system using the measured mounting poses. A KD-tree nearest-neighbor search is then performed in both directions: for each point in cloud A, the distance to the closest point in cloud B is computed and vice versa. Points with a nearest-neighbor distance below a threshold r_{overlap} are considered to belong to the overlapping region. Only these points are kept for registration. The cropped points are stored in their original sensor coordinate frames, which is what GICP requires as input.

3.10.6.2 Registration procedure

The cropped overlap regions are voxel-downsampled and surface normals are estimated. GICP is then run using a multiscale approach, starting with a large correspondence distance (1.5 m) and progressively reducing it (1.0, 0.5, 0.3, 0.15 m). This coarse-to-fine approach helps the algorithm escape false local minima in the early iterations while achieving precise alignment in the final iterations.

The output of GICP is a refined relative transform T_{refined} between the two sensors. Although the GICP optimization internally models each point as a sample from a local surface patch (using covariance matrices derived from the estimated surface normals), the alignment quality is reported using two point-based metrics: the fitness score (fraction of source points with a nearest neighbor within the maximum correspondence distance) and the RMSE (root mean square of the Euclidean distances between each source point and its nearest neighbor in the target cloud).

3.10.6.3 Convergence region verification

To determine whether the pipeline's estimation errors are small enough for GICP to converge, a convergence region test was performed. First, a ground truth relative transform T_{gt} is established by running the registration procedure (overlap detection, cropping and GICP) using the measured mounting poses as the initial guess. Because these poses are close to the true values GICP converges to its best possible alignment. This result serves as the baseline.

The measured pose of one sensor is then displaced by adding a known translation error in the vessel coordinate system. The displacement is applied along each axis separately (longitudinal, lateral and vertical) and in combined scenarios designed to match the relative errors generated by the calibration pipeline. For each displacement level the entire registration pipeline is repeated. This includes overlap detection for the displaced pose, cropping, preprocessing and GICP registration.

At each error level multiple trials are run where the sign of the displacement is chosen randomly (positive or negative along the axis). This tests whether the algorithm converges regardless of the direction of the error. Convergence is defined as the result being within 100 mm translation and 1° rotation of the ground truth T_{gt}. The convergence rate (fraction of trials that converge) is recorded as a function of the displacement magnitude. This answers whether GICP can consistently converge, given the residual errors remaining from the pipeline.

3.10.7 Center of rotation estimation

The CoR was determined using two IMUs, applying the method described in Section 3.6. The gravity-free lateral acceleration signals from the CPAC IMUs were used, which avoids the need for manual gravity compensation during roll. The sensors were mounted at different locations in the vessel. One was placed aft near the cabin ceiling and the other forward in the stairs leading to the lower deck. The longitudinal separation between the sensors was 2.41 m and the vertical separation was 2.05 m, both measured using a tape measure. The sensors' positions were also measured relative to the stern and lower deck floor. Specifically, the aft sensor was positioned 3.55 m from the stern, and the forward sensor was located 1.06 m above the lower deck floor. The sensor data used to estimate the CoR was collected simultaneously during the periodic yaw and roll maneuvers performed for Step 3 and Step 4.

For the data processing of the yaw CoR a rolling-window approach with 5-second windows was used, since the regression is performed in the time-domain. Only windows with sufficient signal energy and $R^2 > 0.5$ were kept to isolate segments with active maneuvers. For the roll CoR data the gyroscope signal was passed through a 1 Hz low-pass filter to suppress noise prior to differentiation of the angular velocity.

4

Results

This chapter presents the results of the calibration method. The evaluation is divided into three main stages. First, the sensitivity analysis using simulated data examines how sensitive the pipeline is against different parameters. Next, the pipeline and the fine calibration are validated through controlled experiments in an office environment with known ground truth parameters. Finally, the full system, pipeline and fine calibration, is validated through tests on a 13 m vessel.

4.1 Sensitivity analysis of the pipeline with simulated data

This section presents the results from the proposed 4-step calibration pipeline using simulated data. The method and algorithms were tested by varying parameters such as sensor noise, process noise (waves and engine vibrations), recording duration and the intensity of physical maneuvers (acceleration and amplitudes). Throughout this section, solid lines in the plots represent the MAE, while shaded regions indicate the standard deviation across the simulation trials.

4.1.1 Influence of sensor noise and placement on estimation precision

Figure 4.1 illustrates the MAE and standard deviation for the estimated roll, pitch and yaw angles across varying noise levels. As shown in the figure, the roll (blue) and pitch (green) estimates remain stable. The mean absolute error stays near 0° across the tested noise range and the standard deviation remain narrow. The yaw error (red) increases with noise, indicating a loss of precision. The widening shaded region reflects increased variance across the simulation trials. However, the maximum error is still relatively low, at less than 0.1 degrees.

Additionally, simulations were performed using different true initial angles, such as 15° and 45°. The recorded errors from these tests were almost identical regardless of the starting angle. Because the initial angle did not change the resulting error, Figure 4.1 is presented as a representative result for all evaluated orientations.

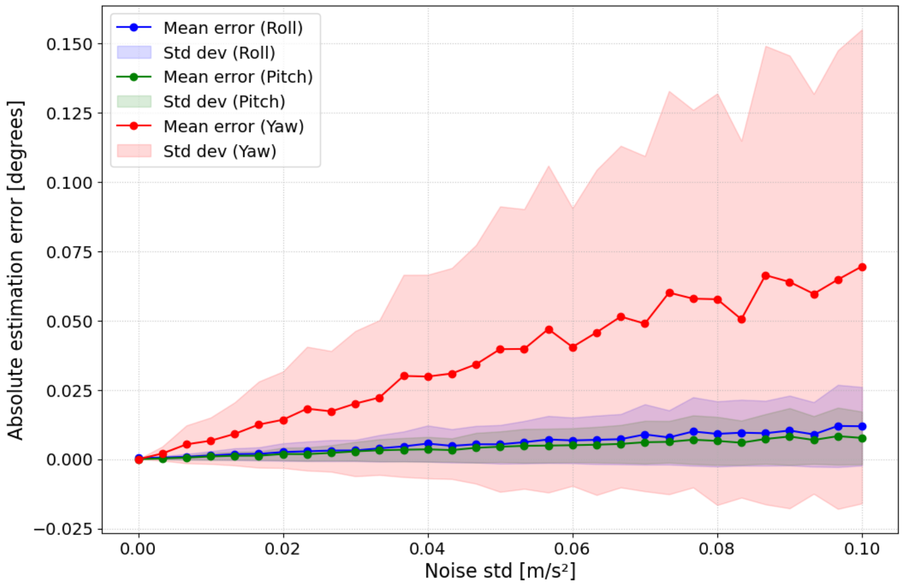

Figure 4.1: MAE of the roll, pitch and yaw estimation as a function of sensor noise.

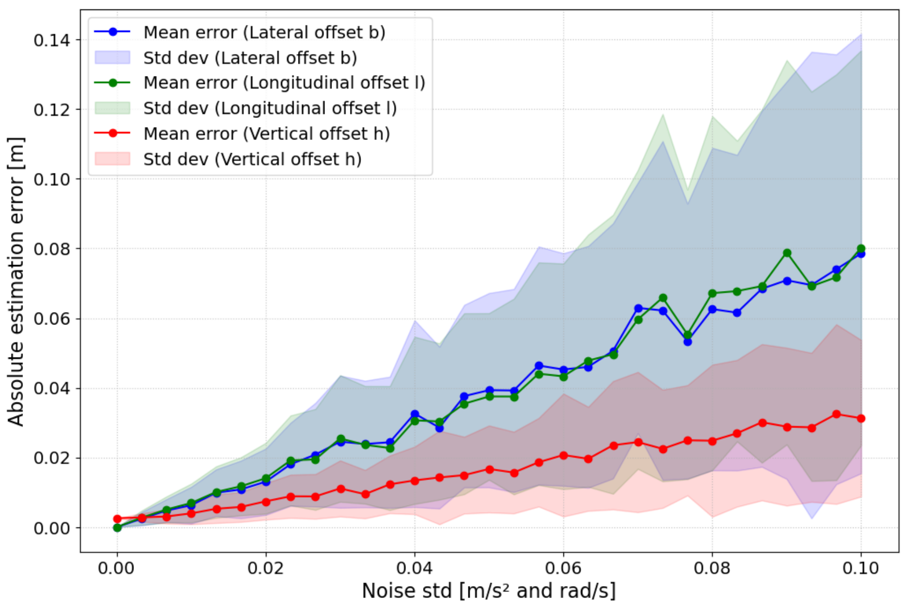

Figure 4.2: MAE of the position estimation at a 2 m range as a function of sensor noise.

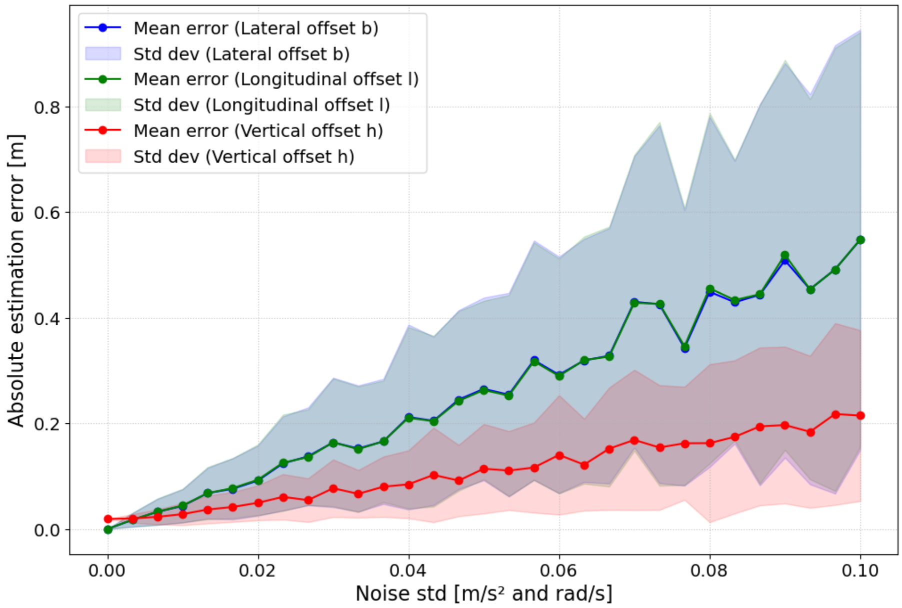

Figure 4.3: MAE of the position estimation at a 15 m range as a function of sensor noise.

The robustness of the estimated positions under varying levels of Gaussian white noise is displayed in Figures 4.2 and 4.3. Both figures show a largely linear correlation between the added noise standard deviation and the resulting estimation error. Across both tested distances, the MAE for the vertical height estimate (h) is consistently lower than the errors for the lateral (b) and longitudinal (l) offsets. In the 2 meter configuration (Figure 4.2), the MAE for h remains below 4 cm at the maximum noise level, while the errors for the horizontal components reach approximately 8 cm. The results indicate that the estimation error increases along with the distance parameters. At a distance of 15 meters (Figure 4.3), the maximum horizontal MAE reaches approximately 60 cm.

Figure 4.4 plots the MAE of the position estimation as a function of the distance to the sensor, from the CoR, in meters. The figure show that the estimation error grows proportionally as the distance increases.

4.1.2 Influence of recording duration on estimation precision

The following analysis evaluates how the four pipeline steps perform with different recording times.

Figure 4.5 displays the duration sensitivity for Step 1, where the accelerometer data is averaged over time to estimate the roll and pitch offset. The mean estimation error is even for very short recordings below 0.0025 degrees. As the recording duration increases the estimation error and the variance is decreased further. The most substantial improvements occur within the first 100 seconds, after which the curve

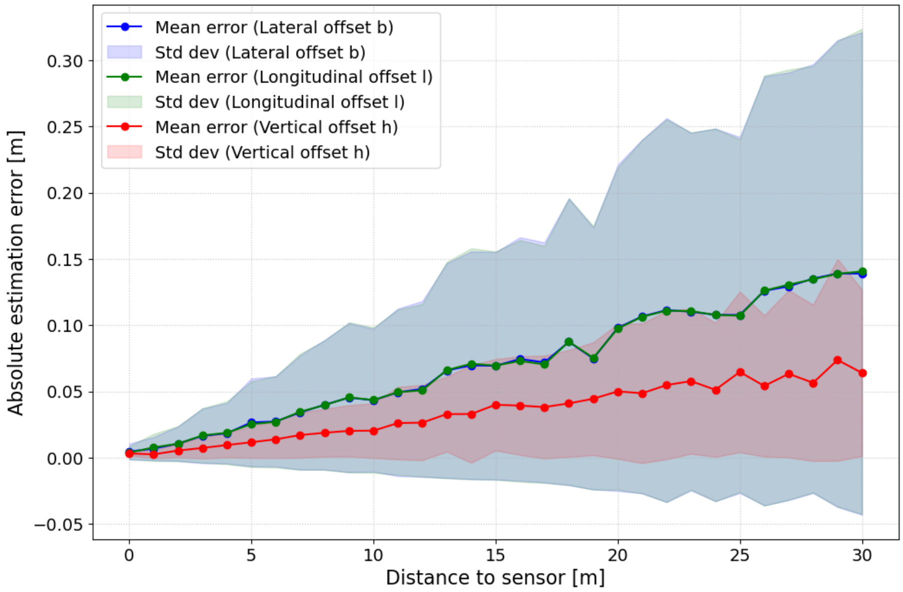

Figure 4.4: MAE of the position estimation as a function of sensor distance to the CoR. Note that the lateral error (blue line) overlaps with the longitudinal error (green line).

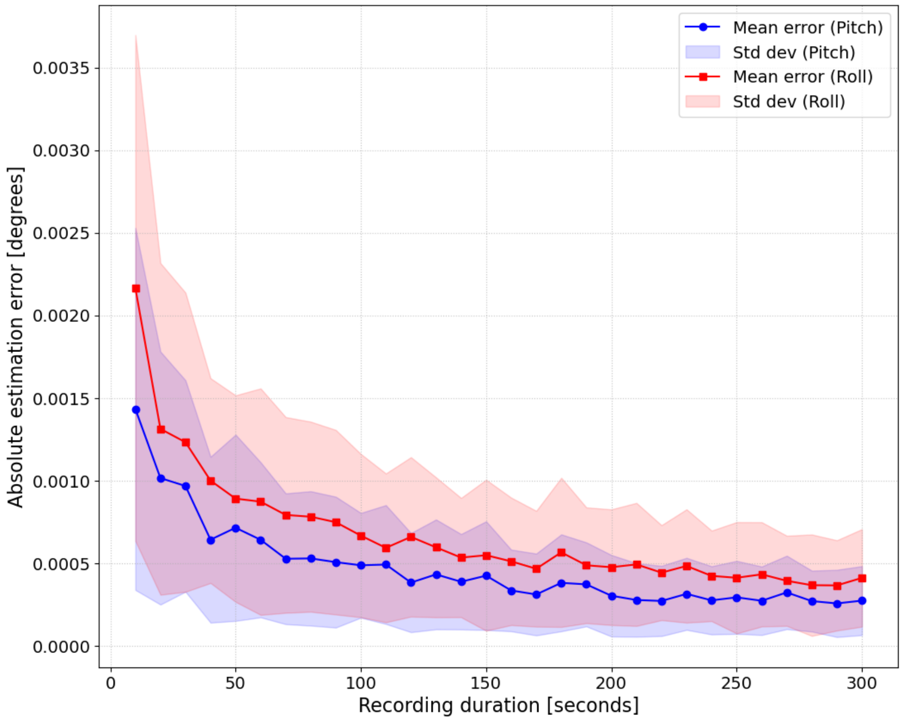

Figure 4.5: MAE of the roll and pitch estimation (Step 1) as a function of recording duration.

begins to flatten out across the remaining durations. This indicates diminishing returns for extending the recording duration beyond 150-200 seconds.

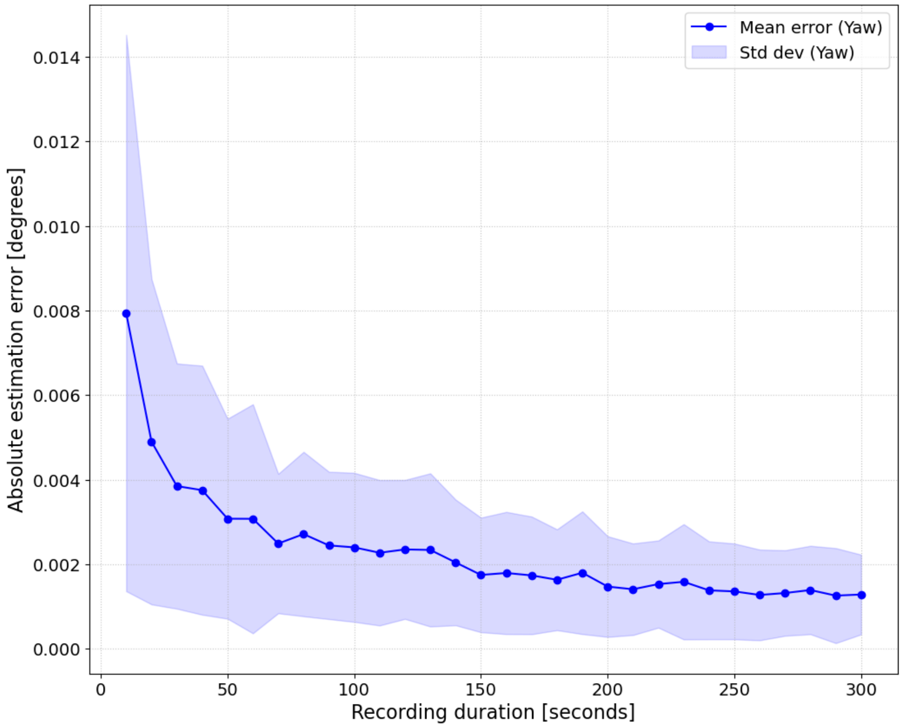

Figure 4.6: MAE of the yaw estimation (Step 2) as a function of recording duration.

Similarly, Figure 4.6 shows the duration sensitivity for Step 2, where the yaw offset is estimated by averaging the lateral and longitudinal accelerations over time during a straight-line acceleration maneuver. Here the estimation error is also largest for short durations. Increasing the duration reduces the error since the disturbances are averaged out. The graph drops sharply during the initial phase and then levels out.

Figure 4.7 shows the effect of recording time on the horizontal position estimation performed in Step 3 where the lateral and longitudinal offsets are estimated. For this evaluation the algorithm attempts to estimate a true lateral offset of $b = 1.0$ m and a true longitudinal offset of $l = 1.0$ m. Short recordings of 10-40 seconds result in large estimation errors exceeding 0.7 m. For a duration of 60 seconds there is a mean estimation error of more than 10 cm.

As the recording duration increases the estimation error displays an exponentially decaying curve. A distinctive sharp decline in the estimation error is visible up until approximately 100 seconds. Beyond 150 seconds the effect of increasing the duration further is minor.

Finally, Figure 4.8 presents the duration sensitivity for Step 4, which estimates the vertical position offset (h). Similar to the horizontal position estimation in Step 3,

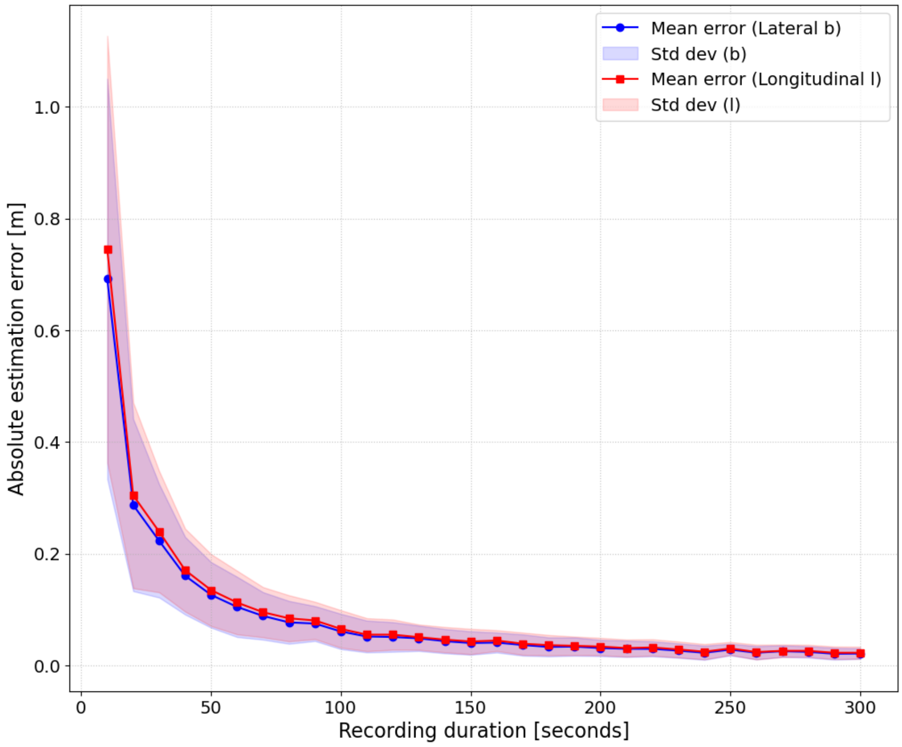

Figure 4.7: MAE of the horizontal position estimation (Step 3) as a function of recording duration.

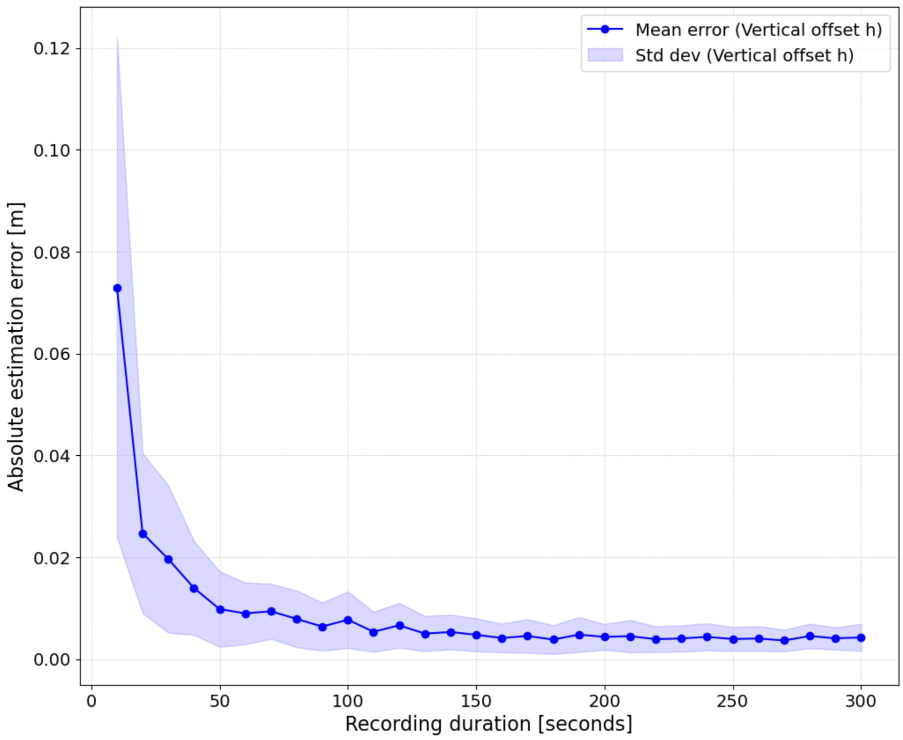

Figure 4.8: MAE of the vertical position estimation (Step 4) as a function of recording duration.

this stage relies on isolating the specific frequency of the physical roll motion using FFT analysis. The true simulated height offset was $h = 1.0$ m. Similar to Figure 4.7 the curve shows an exponential decay. However, the error magnitude is smaller from the outset. For a duration of 30 seconds the mean error is already at 0.195 cm (0.195%).

4.1.3 Error propagation between calibration steps

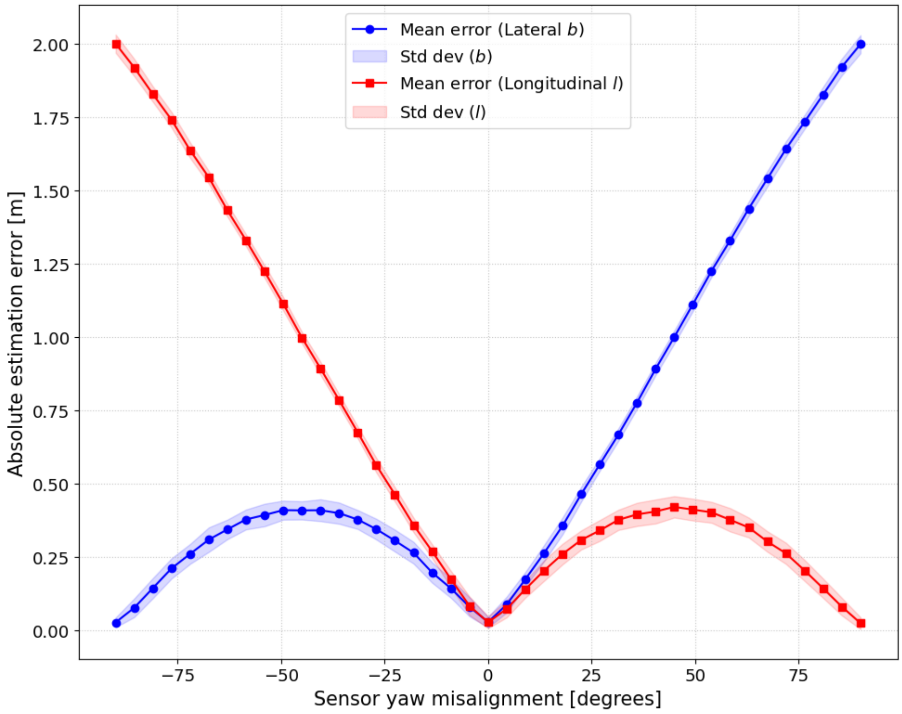

Figure 4.9: MAE of the horizontal position estimation (Step 3) as a function of yaw misalignment from Step 2.

Figure 4.9 isolates the effect of an incorrect ψ_{offset} estimation from Step 2 on the horizontal positions calculated in Step 3. The estimation errors for both the lateral and longitudinal dimensions increase significantly as the sensor yaw misalignment grows. A misalignment of 25 degrees causes an estimation error of more than 25 cm and 50 cm for longitudinal and lateral offset respectively.

4.1.4 Effect of maneuver intensity on signal-to-noise ratio

Figure 4.10 shows that larger forward accelerations reduce the yaw estimation error in Step 2. Even for small accelerations the estimation error is small. For 0.2 m/s^2 the mean estimation error is 0.03 degrees. The curve displays a diminishing return as the acceleration is further increased.

Figure 4.11 shows the effect of increasing the yaw amplitude on the horizontal position estimation in Step 3. The curve is similar to Figure 4.10 and shows an exponential decay. The error reduction is most prominent during the initial increase.

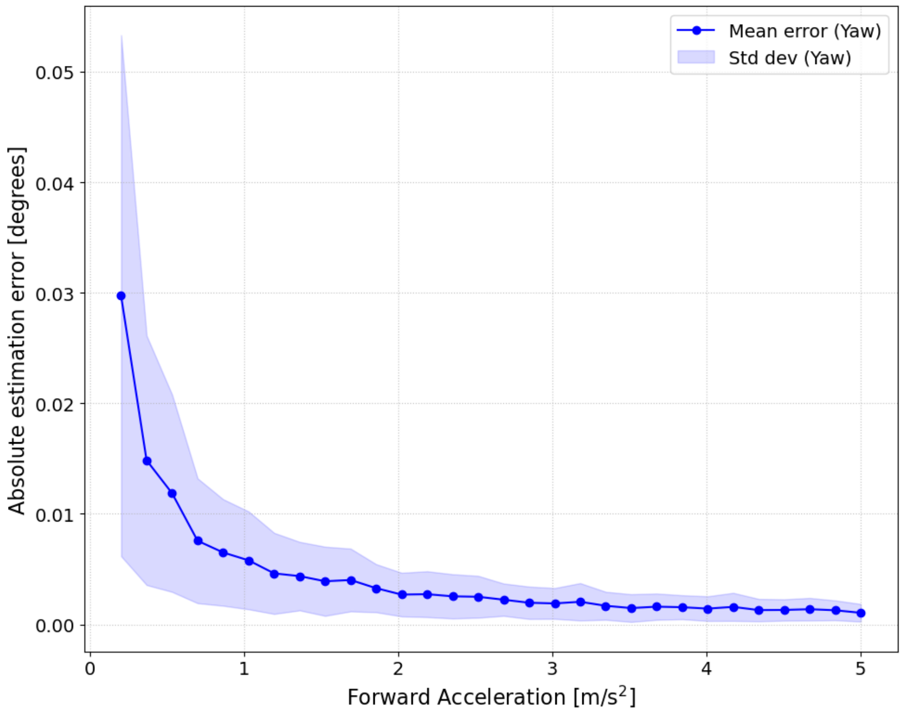

Figure 4.10: MAE of the yaw estimation (Step 2) as a function of forward acceleration magnitude.

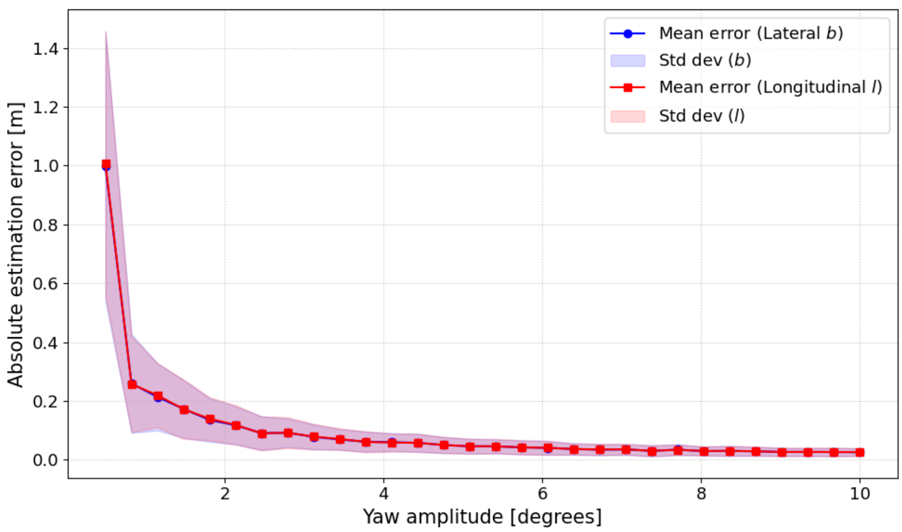

Figure 4.11: MAE of the horizontal position estimation (Step 3) as a function of yaw oscillation amplitude.

For a yaw amplitude of 5 degrees the mean estimation error is slightly less than 5 cm for both offsets.

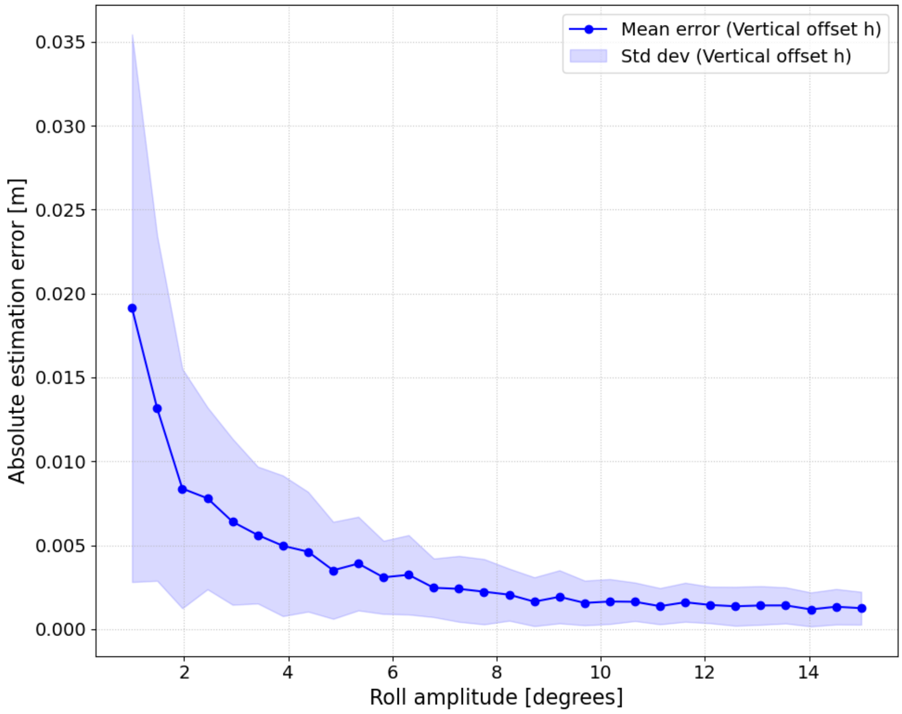

Figure 4.12: MAE of the vertical position estimation (Step 4) as a function of roll oscillation amplitude.

Figure 4.12 confirms this trend for the vertical offset estimation in Step 4. Larger roll amplitudes lead to significantly lower errors when calculating the vertical height (h). The curve shows an exponential decay. The estimation error is low even for small roll amplitudes. An amplitude of 1 degree gives a mean error of less than 2 cm while an amplitude of 4.86 degrees gives a mean error of 3.5 mm. Step 3 shows larger errors at low intensities compared to Step 2 and Step 4.

4.2 Test rig

This section presents the results from testing the calibration pipeline (Steps 1-4) and the fine calibration (GICP) at the office setting.

4.2.1 Step 1: Roll and pitch calibration

Figure 4.13 displays a representative sensor recording showing the measured acceleration over time while the LiDAR was lying flat on a level desk surface. To verify the exact inclination, the surface was measured using an inclinometer, revealing a pitch offset of 0.35° and a roll offset of 0.20°. As shown, the acceleration along the x-axis (blue) and y-axis (orange) is approximately zero, while the acceleration along the z-axis is close to the gravitational constant, $g \approx 9.82$ m/s^2. The pipeline estimated the pitch and roll values to be 1.14° and 0.57°, respectively. This corresponds to an absolute estimation error of 0.79° for pitch and 0.37° for roll. This test was repeated several times with consistent results.

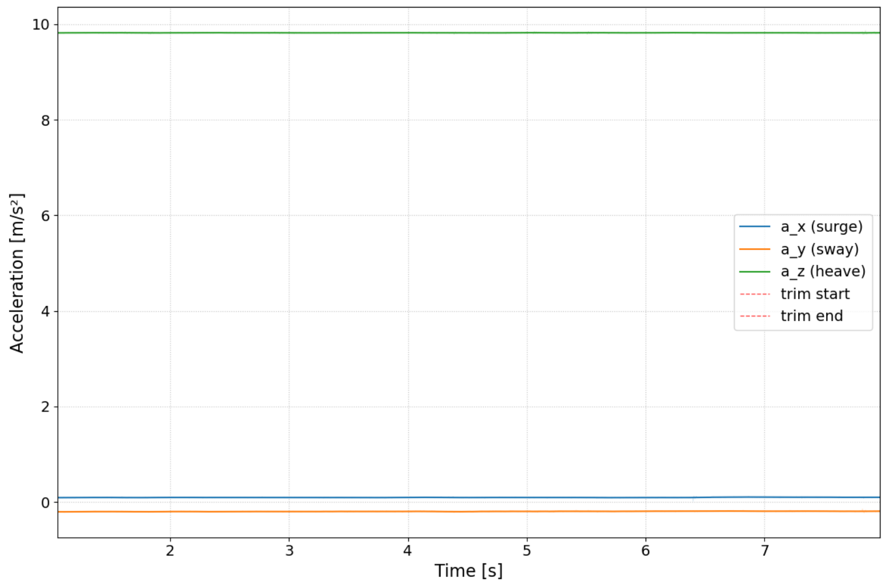

Figure 4.13: Measured acceleration a_x (blue), a_y (orange) and a_z (green) over time when the sensor was lying flat on a surface.

4.2.2 Step 2: Yaw calibration

Figure 4.14 displays the measured horizontal acceleration (a_x and a_y) over time during a cart test. As shown in the Figure, the acceleration begins at approximately 19.5 seconds and lasts for approximately two seconds. The acceleration along the x-axis (blue line) increases steadily before plateauing, while the acceleration along the y-axis (orange line) remains at zero.

The data in Figure 4.14 represents the configuration where the LiDAR dome faces the front of the cart with the cable pointing upwards (see Figure 3.12). Prior to this estimation, the data was compensated for roll and pitch using Step 1. For this orientation, the expected yaw value is 0° with an estimated accuracy of ± 3°. The pipeline's estimated yaw value from this specific measurement was -0.18°, which falls well within this margin.

Several trials were conducted at various mounting orientations. The trial shown in Figure 4.14 was selected as representative of a well-executed straight-line push, where the near-zero a_y confirms that the cart moved in the intended direction.

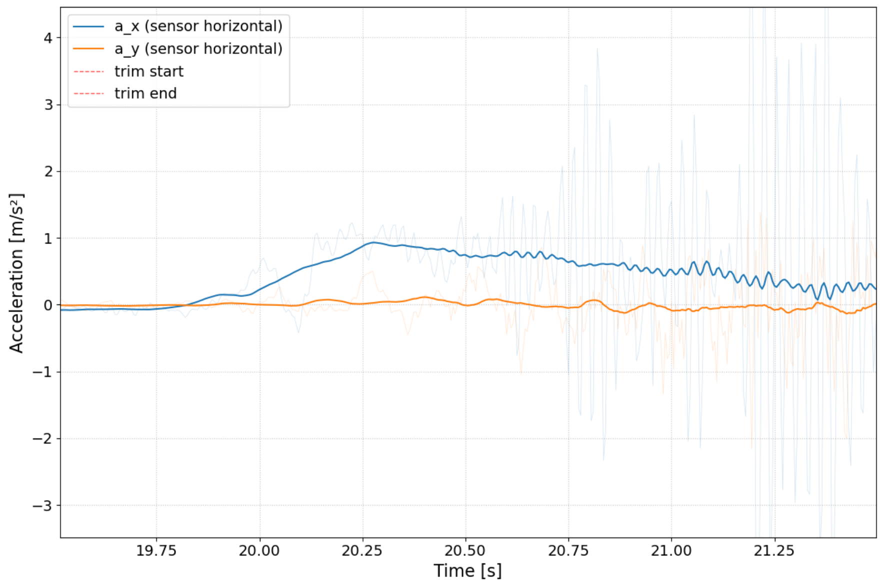

Figure 4.14: Measured horizontal acceleration (a_x and a_y) over time during a single cart test trial. Light traces show the raw sensor signal and bold traces show smoothed values.

4.2.3 Step 3: Horizontal position estimation

For this test, the rig was configured for horizontal (yaw) oscillation. The LiDAR was attached such that its physical center was approximately 27.2 cm longitudinally and 0 cm laterally from the CoR (the motor shaft), with an estimated measurement accuracy of ±0.5 cm.

4.2.3.1 Time-domain signals

Figure 4.15 shows the corrected IMU signals during the yaw oscillation maneuver. The rotation correction from Step 1 and the manual yaw have been applied. The yaw rate (top panel) shows a clean and approximately sinusoidal waveform with an amplitude of roughly ±100 deg/s. The amplitude remains stable across the entire recording. The surge acceleration (middle panel) displays a relatively small periodic component. The sway acceleration (bottom panel) shows a periodic signal with peaks reaching approximately 2.5-4 m/s^2.

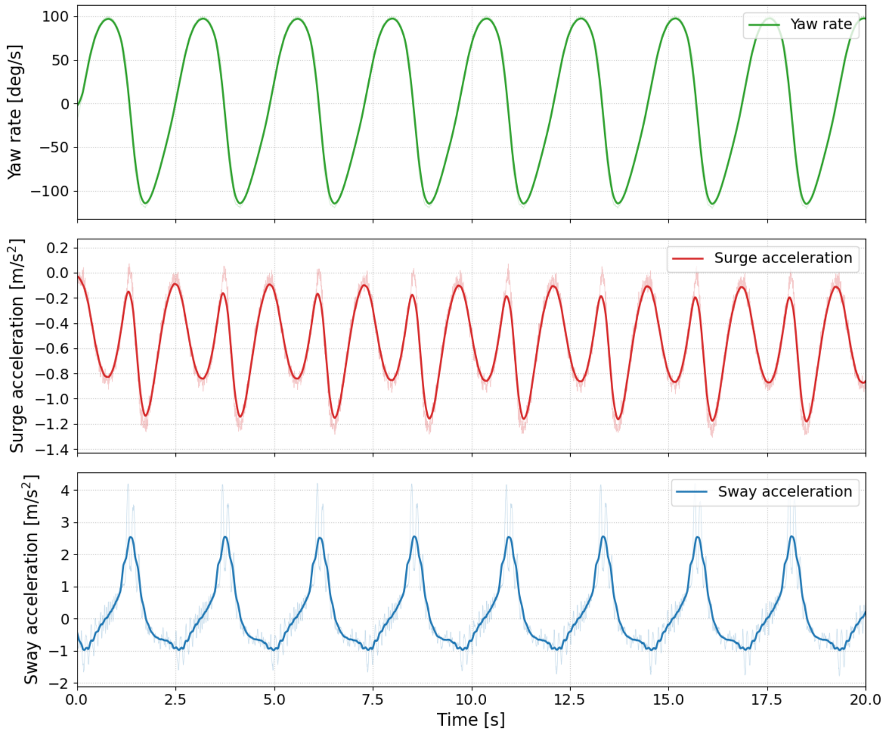

Figure 4.15: First 20 seconds of the measured time-domain yaw rate, surge acceleration and sway acceleration during Step 3 rig oscillations. The oscillation period is approximately 2.4 s.

4.2.3.2 Frequency-domain analysis

Figure 4.16 shows the single-sided FFT magnitude spectrum of the three signals. The green-shaded region shows the bandpass range [0.1, 1.0] Hz. The yaw rate spectrum (top panel) shows a clear distinct peak at approximately 0.42 Hz, corresponding to the oscillation period of roughly 2.4 s.

The surge acceleration spectrum (middle panel) shows a small peak at the fundamental frequency (≈ 0.42 Hz). The sway acceleration spectrum (bottom panel) has a strong peak at the same fundamental frequency.

As described in Section 3.4, the algorithm identifies the peak frequency from the yaw rate spectrum and extracts the amplitude and phase of each signal to compute the lateral and longitudinal offsets.

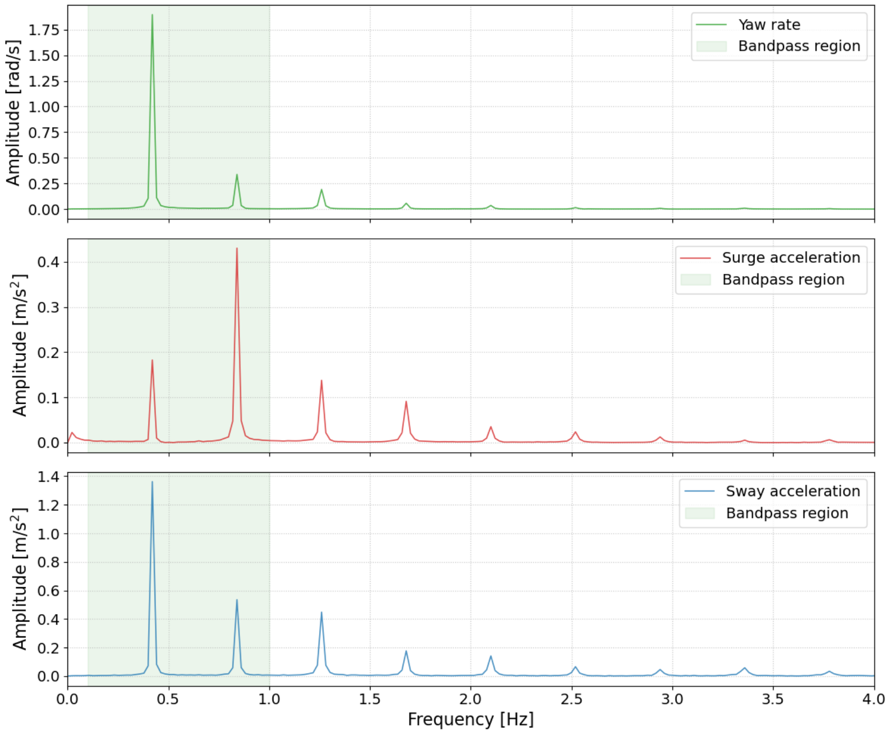

Figure 4.16: FFT magnitude spectrum of the Step 3 signals. The green-shaded region indicates the bandpass range [0.1, 1.0] Hz.

4.2.3.3 Estimation results

Table 4.1 summarizes the Step 3 estimation results for the LiDAR's physical bottom center relative to the CoR. The known distances was measured by hand with an estimated accuracy of ± 0.5 cm.

Table 4.1: Step 3 results: estimated sensor position relative to the CoR.

Parameter	Ground truth	Estimated value	Absolute error
Longitudinal (l)	27.2 ± 0.5 cm	26.8 cm	≤ 0.9 cm
Lateral (b)	0 ± 0.5 cm	3.0 cm	≤ 3.0 cm

The estimated longitudinal offset of 26.8 cm falls within the uncertainty range of the known value. The estimated lateral offset of 3.0 cm is larger than what can be explained by the placement uncertainty alone. Errors are reported only in centimeters because a lateral ground truth of zero makes percentage errors undefined.

4.2.4 Step 4: Vertical position estimation

For Step 4 the test rig was rotated 90 degrees compared to Step 3 so that the motor produced roll oscillations instead of yaw oscillations. The LiDAR was attached at the end of the aluminum plate at a known vertical distance of 27.2 ± 0.5 cm from the CoR and approximately zero lateral offset.

4.2.4.1 Time-domain signals

Figure 4.17 shows the roll rate and sway acceleration recorded during the maneuver. The roll rate (top panel) shows a sinusoidal curve with an amplitude of roughly ±125 deg/s. The combined sway acceleration (bottom panel) reaches peaks of approximately +10 to −7 m/s^2.

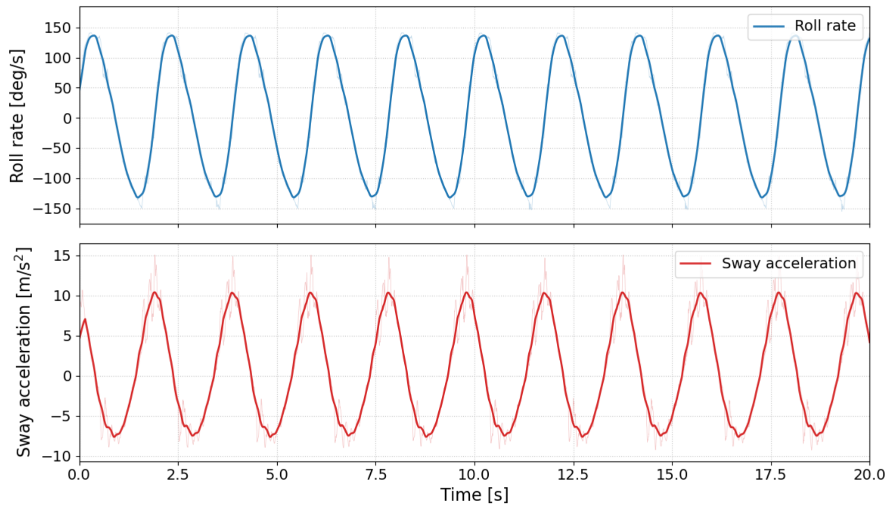

Figure 4.17: First 20 seconds of the measured time-domain roll rate and sway acceleration during Step 4 rig oscillations. The oscillation period is approximately 2.0 s.

4.2.4.2 Frequency-domain analysis

Figure 4.18 shows the FFT magnitude spectrum of both signals. A clear peak is visible at the oscillation frequency of 0.51 Hz in both the roll rate and the sway acceleration. The algorithm extracts the amplitudes at this frequency and uses the velocity method described in Section 3.5 to estimate the vertical offset. After compensating for the gravity contribution using the integrated roll angle, the remaining signal amplitude is used together with the roll rate amplitude to compute the vertical distance.

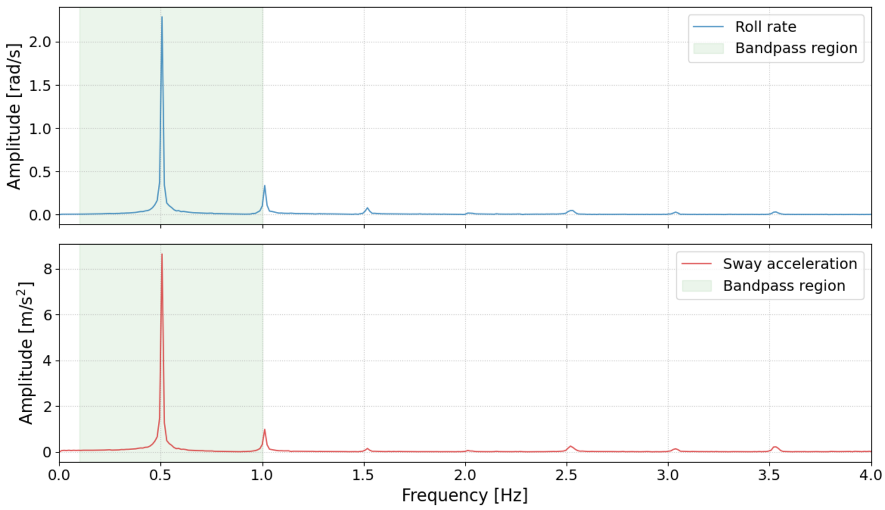

Figure 4.18: FFT magnitude spectrum of the Step 4 signals. The green-shaded region indicates the bandpass range [0.1, 1.0] Hz.

4.2.4.3 Estimation results

Table 4.2 summarizes the Step 4 estimation results for the sensor's physical bottom center relative to the CoR. The known distance was measured by hand with an estimated accuracy of ±0.5 cm.

Table 4.2: Step 4 results: estimated sensor position relative to the CoR.

Parameter	Ground truth	Estimated value	Absolute error
Vertical (h)	27.2 ± 0.5 cm	28.2 cm	$\leq$ 1.5 cm

The estimated vertical offset of 28.2 cm is 1.0 cm above the nominal known value of 27.2 cm. Taking into account the measurement uncertainty of ±0.5 cm, the absolute error is at most 1.5 cm.

4.2.5 Fine calibration sensitivity

The results of the GICP sensitivity analysis, conducted at the office, are presented in Table 4.3. The table illustrates the maximum initial guess error the GICP algorithm can tolerate before failing to converge to the true displacement.

Table 4.3: GICP convergence at varying displacements.

Ground truth	Max tolerated error	RMSE
0.3 m	$\geq$ 80%	0.06
1.5 m	$\geq$ 80%	0.26
3.0 m	50%	0.38
4.0 m	20%	0.51
5.0 m	Fails	0.90

The ability of the GICP algorithm to converge decreases as the physical distance between the scans increases. At smaller displacements between 0.3 m and 1.5 m, the algorithm shows robust convergence. In this range, it easily recovers from initial guess errors of 80% while maintaining a low RMSE.

When the displacement increases to 3.0 m, the algorithm still converges but requires a better initial guess. At 4.0 m, the geometric fit worsens as the RMSE rises to 0.51. At this distance, the process becomes sensitive to the initial guess and requires an error of no more than 20% to find the correct solution. Finally, at a displacement of 5.0 m, the registration fails regardless of the quality of the initial guess.

4.3 Marine testing

This section presents the results from the full pipeline, including fine calibration, and CoR estimation conducted on the 13 m test vessel at sea. The CoR estimation is presented first, followed by the four pipeline steps and then the fine calibration.

4.3.1 Center of rotation estimation

Table 4.4 summarizes the CoR estimation results. For yaw, the estimated distance from the aft sensor to the CoR (r_{aft}) was +1.002 m, where the positive sign indicates a position forward of the sensor. Given that the aft sensor was mounted 3.55 m from the stern, this places the yaw CoR at $3.55 + 1.002 = 4.55$ m from the stern. Since the longitudinal separation between the two sensors was 2.41 m, the CoR is situated between them. The two-sensor differential method proved necessary. Because process noise produced lateral accelerations that were much larger than those from the yaw rotation alone, attempting to use only a single sensor gave an R^2 of 0.008, confirming the necessity of common-mode cancellation.

For roll, the uncorrected single-sensor OLS gave $r_{\text{fwd,raw}} = +2.646$ m ($R^2 = 0.87$). After common-mode correction (estimated bias $\hat{\beta} = +1.459$ m), the corrected result was $r_{\text{fwd}} = +1.187$ m ($R^2 = 0.75$). The forward sensor was located 1.06 m above the lower deck floor, so the roll CoR is $1.06 - 1.187 = -0.13$ m, i.e. 0.13 m below the lower deck floor. The forward sensor was used as the primary result. The aft sensor showed more noise, likely from vibrations at its mounting location.

Table 4.4: CoR estimation results from the marine tests.

Axis	Method	R^2	Distance	CoR position
Yaw	Two-sensor differential	0.61	$r_{\text{aft}} = +1.002$ m	4.55 m from stern
Roll	Two-sensor common-mode corr.	0.75	$r_{\text{fwd}} = +1.187$ m	0.13 m below lower deck

4.3.2 Four-step pipeline results

The following subsections present the pipeline results for the starboard bow LiDAR. The bow port sensor produced similar signals and is omitted for brevity. The final position estimates for both sensors are summarized in Table 4.5.

Compared to the test rig, the boat maneuvers have longer oscillation periods and lower signal-to-noise ratios. The bandpass filter lower cutoff was therefore reduced from 0.1 Hz (test rig) to 0.01 Hz for Step 3 (yaw) and kept at 0.1 Hz for Step 4 (roll). The upper cutoff was set to 0.7 Hz for both steps. The data segments used for estimation were selected manually by inspecting the time-domain signals and choosing intervals with stable oscillations. The ground truth positions were measured by hand on the boat, relative to the aft and lower deck floor, with an estimated accuracy of ±10 cm.

4.3.3 Steps 1 and 2: Orientation estimation

Step 1 estimated the sensor's roll and pitch offsets from a stationary recording. Step 2 estimated the yaw offset from a straight-line acceleration maneuver. These orientation estimates are used internally by Steps 3 and 4 to rotate the accelerometer readings into the vessel coordinate frame. Since the true sensor orientation could not be measured independently on the boat, these values could not be validated against a ground truth. However, the same algorithms were validated in the controlled environment (Sections 4.2.1 and 4.2.2) where ground truth was available. The pipeline proceeded with the estimated orientation, and any errors in Steps 1-2 propagate into the position estimates of Steps 3-4.

4.3.4 Step 3: Horizontal position estimation

The horizontal position estimation was conducted using the data from the 10° yaw oscillation maneuver, at a period of about 14.4 s, as this recording provided the most consistent sinusoidal waveform for the frequency analysis.

4.3.4.1 Time-domain signals

Figure 4.19 shows the corrected IMU signals during the yaw maneuver. The yaw rate (top panel) shows a periodic waveform with an amplitude of about ±7 deg/s. The waveform is not smooth, the peaks are sharp and the shape resembles a triangle wave rather than a sinusoid. This results from the difficulty of steering the vessel into a consistent sinusoidal yaw motion manually.

The surge acceleration (middle panel) has a small periodic component visible in the smoothed trace, but the raw signal is dominated by noise with a range of approximately ±2 m/s^2. The sway acceleration (bottom panel) similarly shows a periodic component in the smoothed trace with peaks of about ±0.5 m/s^2, but the raw signal spans ±10 m/s^2.

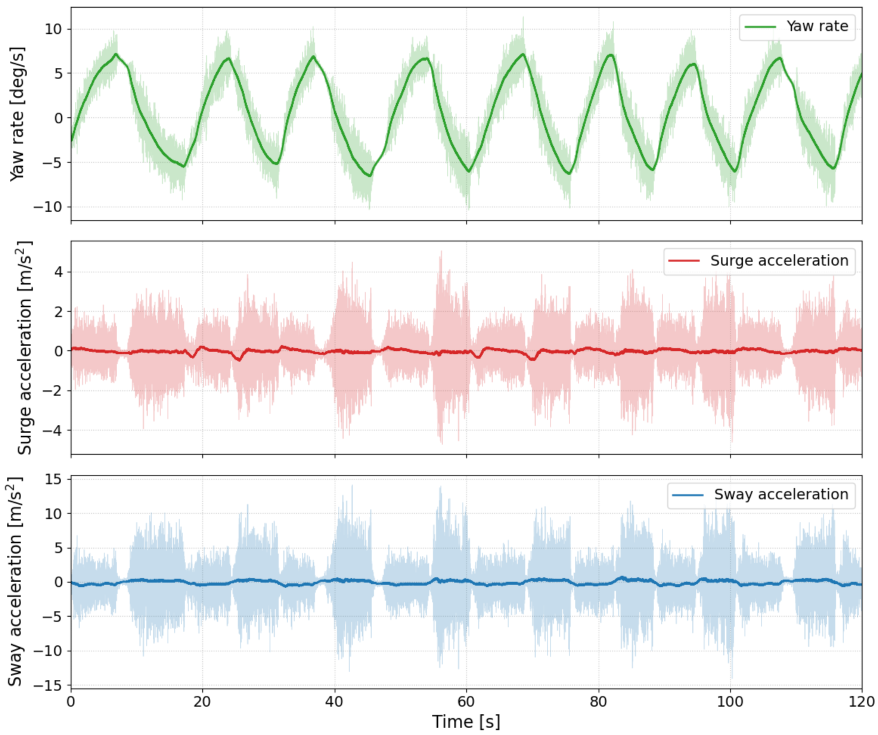

Figure 4.19: 120 seconds of the measured signals during yaw oscillations on the boat (Step 3). Light traces show the raw sensor signal and bold traces show smoothed values.

4.3.4.2 Frequency-domain analysis

Figure 4.20 shows the FFT magnitude spectrum. The green-shaded region indicates the bandpass range [0.01, 0.7] Hz. All three signals show a distinct peak at 0.069 Hz, corresponding to the oscillation period of about 14.4 s.

The sway acceleration peak (bottom panel) is the largest (≈ 0.30 m/s^2), which is consistent with the large longitudinal offset of the sensor from the CoR. The surge peak (middle panel) is smaller (≈ 0.05 m/s^2), consistent with a smaller lateral offset.

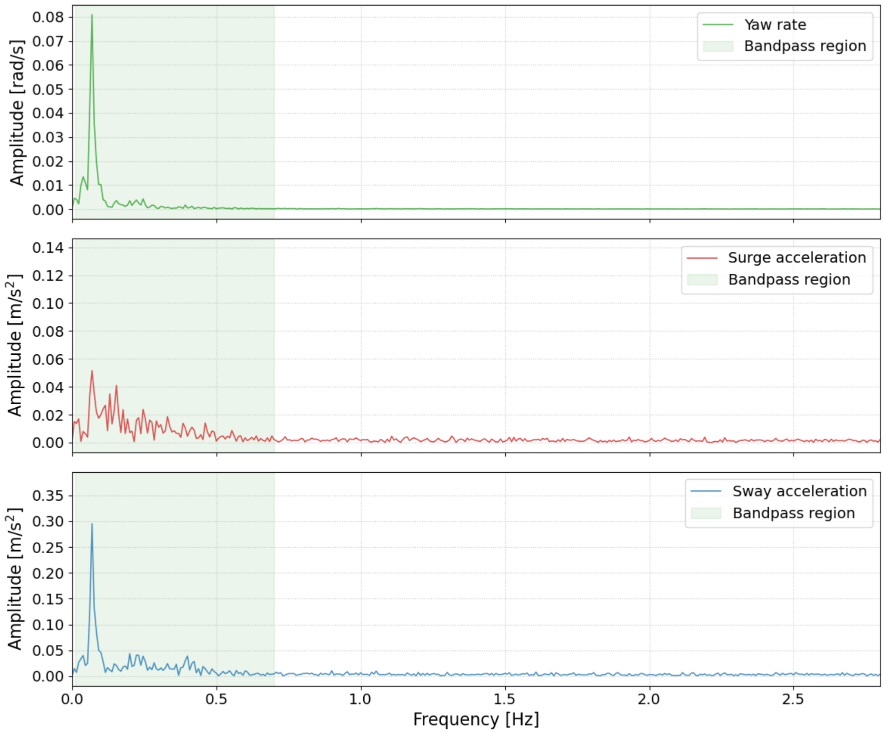

Figure 4.20: FFT magnitude spectrum of the yaw oscillation signals on the boat (Step 3). The green-shaded region indicates the bandpass range [0.01, 0.7] Hz.

4.3.5 Step 4: Vertical position estimation

For Step 4, the boat performed roll oscillations with a period of approximately 3.0 s.

4.3.5.1 Time-domain signals

Figure 4.21 shows the roll rate and sway acceleration during the maneuver. The roll rate (top panel) has an amplitude of roughly ±5 deg/s and is approximately sinusoidal, though with some variation in amplitude between cycles. The sway acceleration (bottom panel) contains both the kinematic signal from the vertical offset and a gravity component from the changing roll angle. The smoothed trace shows a small periodic component (±0.5 m/s^2), while the raw signal spans approximately ±10 m/s^2.

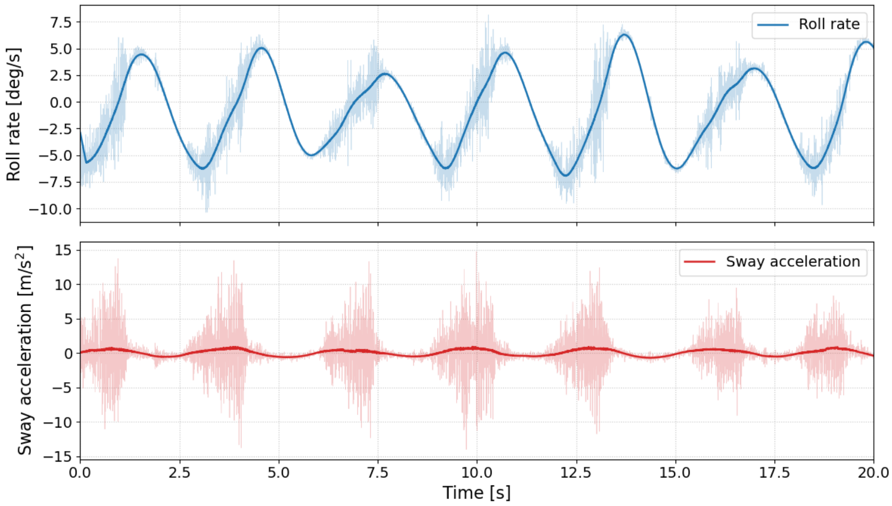

Figure 4.21: First 20 seconds of the measured signals during roll oscillations on the boat (Step 4). Light traces show the raw sensor signal and bold traces show smoothed values.

4.3.5.2 Frequency-domain analysis

Figure 4.22 shows the FFT magnitude spectrum. Both signals peak at approximately 0.33 Hz, corresponding to the period of 3.0 s. The sway acceleration peak (≈ 0.63 m/s^2) includes the gravity component $g\sin(\varphi)$ in addition to the kinematic component $h\ddot{\varphi}$.

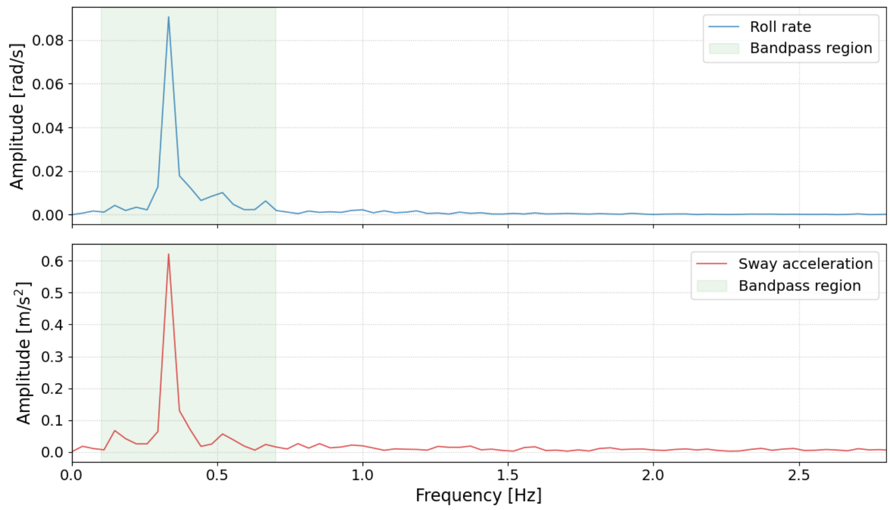

Figure 4.22: FFT magnitude spectrum of the roll oscillation signals on the boat (Step 4). The green-shaded region indicates the bandpass range [0.1, 0.7] Hz.

4.3.6 Position estimation summary

Table 4.5 presents the estimated sensor positions for both the starboard and port bow LiDARs, expressed as distances from the CoR. The lateral error ranges from

0.05 m (3.0%) for the starboard sensor to 0.17 m (10.3%) for the port sensor. The longitudinal error is about 1 m for both sensors (15-17%), placing the estimated position further forward than the measured position.

The vertical estimates have errors of 0.33-0.42 m (23-29%). Step 4 estimated the sensor height above the roll CoR as $h = 1.03$ m for the starboard sensor and $h = 1.12$ m for the port sensor, compared to the true height of 1.45 m above the CoR. The ground truth was measured by hand with an accuracy of ±10 cm. Therefore, errors at or below this level can be caused by measurement uncertainty.

Table 4.5: Pipeline estimation results for starboard and port bow LiDARs. All positions are relative to the CoR (positive lateral values indicate starboard). The CoR is 4.55 m from the stern and 0.13 m below the lower deck floor.

Sensor	**Axis**	**Ground truth**	**Estimated**	**Abs. Error**	**Error (%)**
Starboard	Longitudinal	6.85 m	7.89 m	1.04 m	15.2%
	Lateral	+1.65 m	+1.70 m	0.05 m	3.0%
	Vertical	1.45 m	1.03 m	0.42 m	29.0%
Port	Longitudinal	6.85 m	8.04 m	1.19 m	17.4%
	Lateral	-1.65 m	-1.82 m	0.17 m	10.3%
	Vertical	1.45 m	1.12 m	0.33 m	22.8%

Relative to the boat, the sensors are located 11.40 m from the stern and 1.32 m above the lower deck floor, and are estimated to be 12.44 m (starboard) and 12.59 m (port) from the stern, and 0.90 m (starboard) and 0.99 m (port) above the lower deck floor.

4.3.7 Fine calibration

The fine calibration was tested on the recorded LiDAR data from the vessel. The sensor pair with the most overlapping FOV was identified and used for the GICP registration.

4.3.7.1 Sensor pair selection

Not all sensor pairs produced usable results. The two bow sensors (starboard and port) had a yaw difference of approximately 142°, so the overlap between these two sensors was too small. The sensor pair that did produce a successful registration consisted of the starboard bow-mounted RoboSense Airy and the starboard roof-mounted RoboSense Helios. These two sensors were separated by approximately 8.1 m on the vessel. However, both could simultaneously observe the pier environment.

4.3.7.2 Point cloud overview

Figure 4.23 shows both point clouds transformed into the vessel coordinate system using the measured mounting poses. The roof-mounted Helios (blue) covers a wider lateral range along the shoreline and docks, but with sparser scan lines. The bow-mounted Airy (red) has a narrower lateral coverage but captures the building and wooden dock with higher point density.

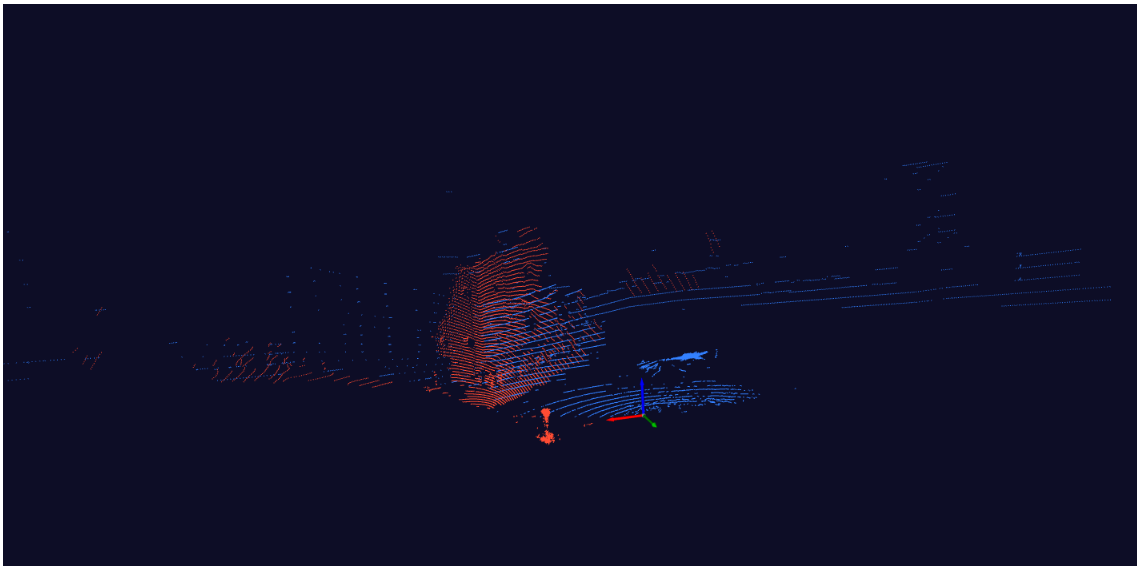

Figure 4.23: Both LiDAR point clouds in the vessel coordinate system. Red: bow-mounted Airy. Blue: roof-mounted Helios. The building and the wooden dock (below the building) on the pier are visible to both sensors.

4.3.7.3 Overlap detection

Using the measured mounting poses to transform both point clouds into the vessel coordinate system, a KD-tree nearest-neighbor search was performed with a radius threshold of 1.2 m. After cropping, 70.8% of the Airy's points and 92.1% of the Helios's points were discarded as non-overlapping. The Helios has a larger discard fraction because most of its wide-angle coverage extends beyond what the Airy can see. Figure 4.24 shows the overlap detection result.

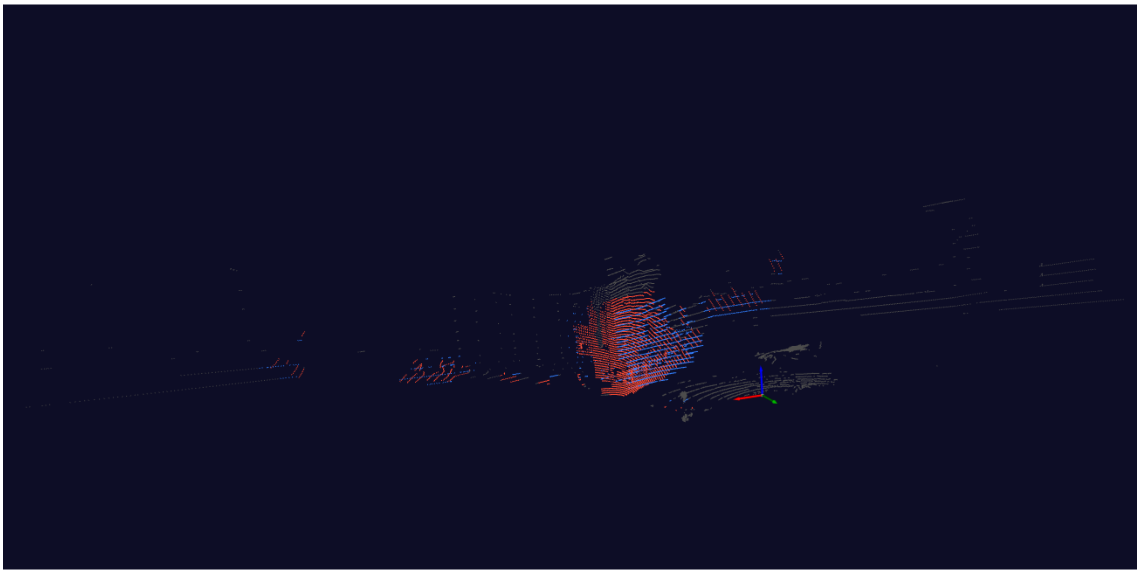

Figure 4.24: Overlap detection result. Red: Airy points in the overlap region. Blue: Helios points in the overlap region. Gray: discarded non-overlapping points. The overlap radius threshold was 1.2 m.

4.3.7.4 Registration result

GICP was run on the cropped overlap regions, with correspondence distances decreasing from 1.5 m to 0.15 m. The algorithm converged and produced a refined relative transform between the two sensors. Table 4.6 shows the correction that GICP applied to the measured mounting poses.

Table 4.6: GICP correction from the measured mounting poses. The correction represents the difference between the manually measured relative pose and the GICP-refined alignment.

Parameter	**Correction**
Longitudinal (Δx)	-625 mm
Lateral (Δy)	-443 mm
Vertical (Δz)	-7 mm
Roll ($\Delta\varphi$)	$+6.60°$
Pitch ($\Delta\theta$)	$+1.98°$
Yaw ($\Delta\psi$)	$-7.67°$
Total translation ($\|\Delta t\|$)	766 mm
Total rotation ($\|\Delta \text{rpy}\|$)	10.3°

The total correction was 766 mm in translation and 10.3° in rotation. This means the manually measured poses contained errors of that magnitude. The GICP-refined result serves as the ground truth baseline for the convergence region test described below.

4.3.7.5 Convergence region analysis

To determine whether the pipeline errors from Table 4.5 are small enough for GICP to converge, a convergence region test was performed. The initial guess was displaced by known amounts in the vessel coordinate system and GICP was re-run from each displaced starting point. For each displacement level, 20 trials were run with randomly chosen sign (positive or negative). Table 4.7 shows the per-axis convergence results. The displacement was applied along one axis at a time while keeping the other two at zero.

Table 4.7: Per-axis convergence region. Each row shows the convergence rate (out of 20 trials) at a given displacement magnitude along a single axis.

Displacement	**X (longitudinal)**	**Y (lateral)**	**Z (vertical)**
0.25 m	20/20 (100%)	20/20 (100%)	20/20 (100%)
0.50 m	20/20 (100%)	20/20 (100%)	20/20 (100%)
0.75 m	20/20 (100%)	20/20 (100%)	20/20 (100%)
1.00 m	20/20 (100%)	20/20 (100%)	20/20 (100%)
1.25 m	20/20 (100%)	20/20 (100%)	8/20 (40%)
1.50 m	20/20 (100%)	20/20 (100%)	20/20 (100%)
1.75 m	0/20 (0%)	0/20 (0%)	0/20 (0%)
2.00 m	0/20 (0%)	0/20 (0%)	0/20 (0%)

The convergence boundary is approximately 1.5 m for all three axes. Above 1.75 m, convergence drops to 0% for all axes. The result at Z = 1.25 m (40%) may indicate a local minimum at that particular vertical offset for some sign directions.

Table 4.8 shows the results for combined scenarios where all three axes are displaced simultaneously. The displacement magnitudes are taken directly from the pipeline errors in Table 4.5.

Table 4.8: Combined convergence scenarios with simultaneous displacement along all three axes. The pipeline error magnitudes are taken from Table 4.5. Signs are randomized across 20 trials.

Scenario	Δx	Δy	Δz	$\|\Delta t\|$	**Rate**
Relative	0.15 m	0.22 m	0.09 m	0.28 m	20/20 (100%)
SB individual	1.04 m	0.05 m	0.42 m	1.12 m	20/20 (100%)
PT individual	1.19 m	0.17 m	0.33 m	1.25 m	20/20 (100%)
Worst-case	1.19 m	0.22 m	0.42 m	1.28 m	20/20 (100%)

All four scenarios converge at 100%. This is consistent with the per-axis results: the largest single-axis component in the worst-case scenario is $\Delta x = 1.19$ m, which is below the 1.5 m x-axis boundary.

The relative scenario represents the difference between the errors of two sensors calibrated in the same session. The longitudinal errors from Table 4.5 are both positive (both sensors estimated too far forward) so the relative longitudinal error is $|1.19 - 1.04| = 0.15$ m. For the lateral axis, the errors have opposite sign ($+0.05$ m for starboard, -0.17 m for port) so the relative lateral error is $0.05 + 0.17 = 0.22$ m. The vertical relative error is $|0.42 - 0.33| = 0.09$ m since both sensors are estimated too low.

5

Discussion

This chapter discusses the results presented in the previous chapter. The discussion is structured to follow the same progression: first the simulated sensitivity analysis, then the test rig experiments and finally the marine tests.

5.1 Sensitivity analysis of the pipeline with simulated data

The results from the simulations provide insights into how the 4-Step calibration pipeline behaves. This section discusses what these findings can mean for a real-world LiDAR-IMU calibration system on a marine vessel.

5.1.1 Sensitivity and robustness to sensor noise

The estimation error for yaw is more sensitive to noise compared to roll and pitch (Figure 4.1). This difference occurs because roll and pitch can be corrected using the gravity vector which provides a strong and stable reference. Yaw lacks a similar absolute reference in the horizontal plane making it naturally more sensitive to sensor noise. Despite this sensitivity the maximum orientation errors remain low. The errors are independent of the initial starting angles confirming that the orientation algorithm is robust across arbitrary mounting positions.

The sensitivity analysis also shows that the vertical parameter (h) is more robust than the lateral (b) and longitudinal (l) offsets. This difference is largely due to the underlying measurement principles, the height estimation relies on signal amplitude, which is easier to extract in noisy environments. In contrast, horizontal positioning depends more on phase differences, which are sensitive to noise.

A practical way to mitigate the distance-dependent errors is to increase the measurement time. A longer recording period provides more samples (N) for the FFT and temporal averaging process. This statistically reduces the standard error of the mean by a factor of $1/\sqrt{N}$. Using a high-quality IMU such as the one integrated in the RoboSense Airy LiDAR also gives a low baseline noise.

5.1.2 Sequential calibration and orientation accuracy

The results show how sensitive the position estimation in Step 3 and Step 4 is to poor orientation estimations (Figure 4.9). Even a small error in the ψ_{offset} from Step 2 produces large estimation errors for the position parameters.

This means that the pipeline must be strictly sequential. The frequency-based position method relies entirely on the LiDAR axes being well-aligned with the boat axes. An inaccurate yaw calibration makes the position estimation unreliable. It is therefore essential to perform the orientation calibration in Step 1 and Step 2 before the position calibration in Step 3 and Step 4. The orientation calibration has to be performed accurately in order for the position calibration to be accurate.

5.1.3 Lever-arm effect and distance-dependent uncertainties

Errors in horizontal positioning increase the further away the sensor is from the boat CoR (Figures 4.2, 4.3 and 4.4). This is caused by the lever-arm effect. Small angular errors caused by noise ($\Delta\theta$) translate into position errors (Δx) that are proportional to the physical distance (r) following the formula $\Delta x \approx r \cdot \Delta\theta$. In practice this means that a LiDAR placed far out on the bow or stern of a large vessel will have a larger horizontal uncertainty than a sensor placed near the CoR. To counter this sensors located far from the CoR require longer recording times and strong maneuvers to achieve an acceptable accuracy.

5.1.4 The effect of distinct maneuvers

Physical maneuver intensity directly affects the estimation accuracy (Figures 4.10-4.12). Stronger accelerations and larger yaw and roll amplitudes increase the signal amplitude. This improves the separation between the target signal and the noise floor. This increase in SNR is the reason for the error reduction.

Because of this, it will probably not work to run the calibration passively while the boat naturally moves due to waves and wind. Without sufficient vessel motion the kinematic signals required for calibration are drowned out by background noise. The maneuvers required in the proposed pipeline have to be performed intentionally. Data recording should be performed when the boat is maneuvered as required for each calibration step.

Step 3 shows a larger sensitivity causing larger errors for small intensities. The horizontal position estimation relies on calculating two separate parameters (b and l) and depends on phase shifts. It could therefore be beneficial to perform Step 3 more aggressively than the roll maneuvers in Step 4. This is important if there is a lot of process noise.

5.1.5 Balancing calibration time and frequency-domain resolution

There is a clear trade-off between calibration speed and estimation accuracy. Short recording times lead to spectral leakage and large estimation errors in Step 3 and Step 4. Because frequency resolution is tied to time ($\Delta f \approx 1/T$) a short measurement causes the FFT to blur frequencies together. Noise with a substantial amplitude at 0.3 Hz from waves will leak into a target 0.2 Hz steering maneuver. This causes large estimation errors for short durations (Figures 4.7 and 4.8). By extending the recording duration, the frequency resolution increases, making it easier to separate the boat maneuvers from the wave noise. Trying to calibrate in just a few seconds on a moving boat is therefore not feasible. The system requires continuous and stable data collection during the maneuver.

5.1.6 Sensor noise vs process noise

Although sensor noise increases calibration errors it is a minor problem in practice. In a real scenario, the primary source of error is most likely process noise caused by waves, currents, winds and vibrations. These will cause physical disturbances that are larger than the sensor's own noise. Overcoming this process noise is therefore the main challenge when transitioning from simulations to real-world testing.

5.2 Test rig

This section discusses the findings from the testing at the office, where the complete four-step calibration pipeline as well as fine calibration was executed in a controlled environment.

5.2.1 Step 1: Office setup

The results in Section 4.2.1 show a difference between the LiDAR's IMU estimates and the physical surface tilt measured by the inclinometer. By subtracting the surface tilt from the estimates the actual sensor bias is calculated to be 0.79° for pitch and 0.37° for roll. Factoring in the inclinometer accuracy of ±0.1°, this confirms a small internal error in the IMU of the LiDAR.

5.2.2 Step 2: Yaw calibration

The estimated ψ_{offset} of $-0.18°$ is close to the expected value of 0°, confirming that the method works in practice. The main source of error across trials was the quality of the manual cart push. Achieving a perfectly straight and smooth acceleration by hand was difficult, and trials where the cart deviated laterally produced larger errors. The representative trial was selected because the near-zero a_y signal confirms a clean straight-line motion.

5.2.3 Step 3: Lateral estimation error

The longitudinal offset was estimated within the measurement uncertainty of the known value for the chosen period. The sway acceleration had a large amplitude because the longitudinal lever arm was large, which gave the algorithm a clear signal to work with. The strong sway signal and the small surge signal are consistent with the kinematic model (Section 3.4), where $v_{\text{sway}} = -\dot{\psi} \cdot l$ and $v_{\text{surge}} = \dot{\psi} \cdot b$.

The lateral offset was estimated to 3 cm despite the sensor being placed at nominally zero lateral distance. Several factors may explain this error. First, the yaw angle was manually set to $-90°$ rather than being estimated through a Step 2 measurement. Secondly, the simulation results in Section 4.2.3 showed that the lateral and longitudinal estimates are sensitive to yaw misalignment (see Figure 4.9). The observed lateral error is consistent with the magnitude predicted by the error propagation analysis for a small yaw offset.

5.2.4 Step 4: Vertical error

The vertical offset was estimated at 28.2 cm compared to a known value of 27.2 ± 0.5 cm, giving an error of approximately 1.0 cm. This is larger than the longitudinal error in Step 3 but still within a reasonable margin. As observed in the time-domain data the sway acceleration reached higher peak values than in Step 3 because gravity adds to the kinematic signal during roll oscillations.

A likely contributor to this error is imperfect gravity compensation. The velocity method used in Step 4 removes the gravity component $g \sin(\varphi)$ by integrating the roll rate to obtain the roll angle φ. This integration is sensitive to gyroscope bias drift. Even a small constant bias in the roll rate accumulates over time and causes the estimated roll angle to drift, which in turn introduces inaccuracies in the gravity subtraction.

It should also be noted that on the test rig, the sensor oscillates over approximately 83 degrees, which means the gravity projection becomes large near the turning points. On a real vessel, the roll amplitude is typically much smaller, which reduces the relative magnitude of the gravity term. Furthermore, CPAC's sensor fusion algorithms, which are used in their dedicated IMU to produce gravity-free acceleration signals, could potentially be applied to the LiDAR IMU data as well. This would likely improve the vertical estimate.

5.2.5 Choice of sensor for test rig measurements

During the test rig measurements (Steps 3 and 4), both a RoboSense Airy LiDAR and a separate CPAC IMU were evaluated. The results for Step 3 were similar across both sensors. For Step 4, however, the heavier CPAC IMU caused noticeable vibrations in the rig at the turning points of the oscillation. This introduced additional noise and spiky artifacts in the measured signals.

Additionally, the test rig was unstable for long lever arms and slow oscillation periods. When the lever arm was extended, the higher torque meant the rig motor struggled to maintain a constant speed, especially as the period was increased. This caused the sensor to take longer to travel downward than upward due to gravity, producing angular velocity curves that were spiky rather than sinusoidal. For this reason, results from longer periods were excluded. On a real vessel, where the sensor is rigidly attached to a large structure, these weight-induced vibrations and asymmetries would not occur. However, other sources of noise such as waves and engine vibrations would be present instead.

5.2.6 Fine calibration sensitivity

The results from the fine calibration verification show that the GICP algorithm depends on having an overlapping FoV and enough geometric structures therein, even when the initial guess is close to the true value. At a small displacement of 0.3 m, the sensors share an almost identical view of the room. This large FoV overlap creates stable and clear matching conditions, allowing GICP to achieve a robust geometric fit (RMSE = 0.06) and successfully handle initial guess errors as large as 80%.

The algorithm's failure at a 5-meter displacement (Table 4.3) was caused by the environment rather than the algorithm itself. Because the test took place in a narrow corridor at the office, moving the sensor 5 meters left no shared geometric features between the two views. Without these common points, the algorithm could not find a match, leading to the high RMSE values shown in the data.

This result is important for the final setup on a vessel. To ensure a successful calibration, the vessel must be in a feature-rich area, such as a harbor. If there are enough features for the LiDARs to see at once, in their overlapping FoV, the algorithm will have enough overlapping data to converge.

5.3 Marine testing

This section discusses the results from the full pipeline tested on the 13 m vessel at sea, including the CoR estimation, the four pipeline steps and the fine calibration.

5.3.1 Center of rotation estimation

The yaw CoR at 4.55 m from the stern places it slightly aft of center on the 13 m vessel. The roll CoR at 0.13 m below the lower deck floor is physically reasonable.

For yaw, the two sensors were on opposite sides of the CoR, producing anti-phase lateral signals. The differential $a_1 - a_2$ cancels any shared lateral acceleration and isolates the rotational component. For roll, both sensors were above the CoR (in phase), so a different approach was needed.

The single-sensor OLS regression for roll showed a large positive bias (+1.459 m). The most likely explanation is a shared lateral acceleration $C(t)$ that is correlated with the angular acceleration α. Such a correlation could arise if the side-thruster that drives the roll also generates a lateral hull motion. In the data, the correlation between the estimated common-mode signal and α was 0.82. The two-sensor common-mode correction reduced the observed bias in the regression by identifying and subtracting the shared component. After correction, the estimated CoR moved from 2.73 m to 1.27 m below the cabin floor.

The yaw R^2 of 0.61 reflects the non-sinusoidal character of the maneuver: the angular acceleration was concentrated in short transition periods while the angular velocity was approximately constant in between. Smoother oscillations would likely give a higher R^2.

The ground truth values were measured by hand on the boat with an estimated accuracy of ±10 cm. Part of the reported error may therefore come from measurement uncertainty. The data quality, in particular the high noise level and non-sinusoidal yaw waveform, is a likely source of additional error.

5.3.2 Four-step pipeline

This subsection discusses the performance of the four-step pipeline during the marine tests, including position accuracy, signal quality and environmental limitations.

5.3.2.1 Position estimation accuracy

The vertical error (0.33-0.42 m, 23-29%) has the highest percentage error, though the longitudinal error is larger in absolute terms (1.04-1.19 m). For the vertical, several factors contribute. The roll CoR was estimated with $R^2 = 0.75$ and any error in its position shifts the vertical estimate by the same amount. Additionally, Step 4 is harder than Step 3 since the pipeline must subtract the gravity contribution before estimating h, and small errors in the subtraction produce large relative errors in the remaining signal. The LiDAR IMU outputs raw accelerometer data, so the gravity removal is done in software using the integrated gyroscope. In contrast, the CPAC IMUs used for the CoR estimation output gravity-free acceleration signals using internal sensor fusion algorithms. If similar fusion could be applied to the LiDAR IMU data, the vertical estimation accuracy would likely improve. The maneuver waveforms were also imperfect compared to the simulations and test rig, which further reduces estimation accuracy.

The longitudinal error is the largest in absolute terms (1.04-1.19 m). Both sensors overestimate l by approximately 1 m, while the difference between them is only 0.15 m. This shared bias is consistent with the CoR being located approximately 1 m further aft than estimated. The CoR estimation for the yaw axis had $R^2 = 0.61$ (Table 4.4), indicating uncertainty in the longitudinal CoR position. As a percentage

(15-17%) the longitudinal error is below the vertical (23-29%) because the reference distance is larger (6.85 m vs 1.45 m). The lateral position has the smallest error (0.05-0.17 m) because the lateral CoR is simply the vessel centerline and does not depend on the estimated CoR.

5.3.2.2 Signal quality and data selection

The yaw maneuvers on the vessel did not produce clean sinusoidal curves (Figure 4.19). The waveform was spiky and resembled a triangle wave rather than a smooth sinusoid, with varying peak amplitudes and non-uniform spacing between cycles. Because the amplitude and period are not constant, the FFT peak broadens and the extracted amplitude represents an average across cycles rather than a precise value. Similarly, the roll oscillations in Step 4 (Figure 4.21) had varying amplitude between cycles. These non-ideal waveforms reduce the precision of the frequency-domain estimation compared to the sinusoidal inputs used in simulations and on the test rig.

Data selection was performed manually by inspecting the time-domain signals and choosing intervals with stable oscillations. In a production system, this could be automated using signal quality metrics.

5.3.2.3 Environmental conditions

During the marine tests, the sea conditions were relatively calm. However, waves were still present and contributed to the noise level in the acceleration signals. The vessel is 13 m long, which provided some damping of short-period waves. Further testing under rougher conditions would be needed to determine the method's robustness limits.

5.3.3 Fine calibration

This subsection discusses the prerequisites, performance and limitations of the fine calibration during the marine tests.

5.3.3.1 FOV overlap as a prerequisite

GICP requires overlapping FOV with geometric features. On the test vessel, the starboard and port bow sensors pointed in nearly opposite directions (142° apart), which meant the overlap was too small for registration to succeed. This is not a fundamental limitation of the method but rather a constraint on the sensor configuration, specifically sensor positioning, mounting angles and the total number of units. If 360° coverage is the goal, adjacent LiDARs must share some overlapping FOV with at least one neighbor. In practice, this means that sensor mounting angles should be chosen so that each sensor's FOV partially overlaps with at least one other unit.

5.3.3.2 Convergence region and pipeline errors

All four pipeline error scenarios tested in Table 4.8 converged at 100%, including the worst case with individual axis components up to 1.19 m. Since the per-axis convergence boundary is approximately 1.5 m, this result is expected. The largest single-axis pipeline error (1.19 m longitudinal) is below the boundary.

GICP successfully corrects the maximum observed pipeline errors for this sensor pair. Because the worst-case relative error (1.28 m) remains within the 1.5 m convergence boundary, the fine calibration succeeds regardless of whether the initial pipeline errors of the two sensors are correlated or not. However, this convergence relies on the feature-rich environment. In environments with fewer geometric features or a smaller FOV overlap, the convergence boundary would likely be narrower.

5.3.3.3 Environmental requirements

The registration was performed using a recording where the vessel was approximately 10 m from a building on a pier. In open water, far from any external structures, the fine calibration depends on whether the sensors can observe features on the vessel itself. If two sensors have overlapping FOV that includes part of e.g. the hull, railing or superstructure, these vessel features could potentially serve as registration targets even without external geometry.

In a production system, the software could instruct the operator to position the vessel near a feature-rich environment during the calibration procedure. A harbor with quay walls, other vessels and buildings provides good conditions. The calibration software could also monitor the point cloud overlap and feature density in real time and notify the operator when conditions are sufficient.

5.3.3.4 Absolute vs relative calibration

An important distinction is that GICP aligns sensors relative to each other but not to an absolute vessel reference frame. The pipeline (Steps 1-4) provides each sensor's pose relative to the CoR. When GICP refines the alignment between two sensors, it corrects their relative positioning but does not independently verify the absolute position of either sensor on the vessel.

The absolute positioning depends on the accuracy of the initial pipeline estimate. A longitudinal error of approximately 1 m, as observed in Table 4.5, means the system does not know precisely where the sensors are relative to the bow or stern. However, if at least one sensor can observe part of the vessel in its point cloud, for example a corner at the bow or the stern, the measured distance to these geometric features provides an additional constraint. In that case, the absolute position of that sensor could be determined from the point cloud data itself. The remaining sensors could then be aligned to it with GICP, propagating the absolute reference through the sensor chain. This would require that each sensor shares overlapping FOV with at least one neighbor, forming a connected chain across the vessel.

5.4 Future work

In addition to the improvements discussed above, such as applying gravity compensation algorithms to the LiDAR IMU, different areas for future work can be identified.

Automated maneuvering software could control the vessel during calibration (Step 3 and 4). This would produce clean sinusoidal motions at a known frequency, allow the bandpass filter to be set precisely around that frequency and enable real-time monitoring of signal quality. The software could stop the calibration automatically when sufficient data has been collected. This would reduce operator skill requirements, improve repeatability and likely reduce the total calibration time.

The CoR estimation used in this thesis requires two separate IMUs and manual measurements of their positions. If the CoR is not provided by the manufacturer, future work could explore automating this. Alternatively, the need for a precise CoR could potentially be eliminated entirely. In practice, the CoR is located on the vessel's centerline, roughly in the middle of the vessel and near the waterline. A rough estimate based on these standard dimensions could therefore provide a sufficient initial guess for the pipeline. Once the relative alignment between sensors is refined using GICP, the entire fused point cloud could be aligned to the vessel's reference frame by detecting known structural features, such as the bow or stern, within the LiDAR data, as discussed above. This approach would make the extrinsic calibration fully independent of manual measurements.

For the fine calibration, future work could explore accumulating multiple consecutive LiDAR frames. As the vessel moves slightly between sweeps, combining frames would create a denser point cloud in the overlap region, which could improve registration robustness. Additionally, using detected vessel geometry (e.g. bow or stern corners visible in the point cloud) to determine absolute sensor position would complement the relative alignment provided by GICP.

A production system could also monitor the calibration over time and detect when recalibration is needed. By periodically comparing overlapping point clouds between neighboring sensors, a drift detection algorithm could trigger a new calibration when the alignment degrades beyond a threshold.

The method should be validated on different vessel types and sizes. Testing across a range of platforms would determine how the recording duration and maneuver intensity should be adapted. Finally, robustness testing under rougher sea conditions would determine the method's limits. The current marine tests were performed in relatively calm water. Stronger waves introduce more noise and may require longer recording durations or stronger maneuvers.

6 Conclusion

This thesis developed and validated an online, target-less method for automatic extrinsic calibration of LiDARs on marine vessels. The method uses a four-step pipeline based on frequency-domain analysis of IMU data and rigid-body dynamics to estimate the full six degrees of freedom pose of a sensor relative to the vessel's CoR. A final fine calibration step using GICP refines the relative alignment between sensors using point cloud data.

The method was tested in three environments: simulations with synthetic data, a controlled test rig and a 13 meter vessel. The simulations showed that accurate estimates require sufficiently long recordings and large maneuver amplitudes, and that position errors scale linearly with the sensor distance from the CoR. On the test rig, at a known sensor distance of 27.2 cm from the CoR, the longitudinal, lateral and vertical positions were estimated within 1 cm, 3 cm and 1.5 cm respectively. This confirms that the frequency-domain approach works when the motion is approximately sinusoidal and the signal-to-noise ratio is sufficient.

During real vessel testing, the lateral position was estimated with an error of 0.05-0.17 m (3-10%), the longitudinal position with an error of 1.04-1.19 m (15-17%) and the vertical estimate with an error of 0.33-0.42 m (23-29%). All errors are calculated relative to the CoR. The vertical estimate is limited by gravity compensation when using raw accelerometer data. Furthermore, the estimation of CoR introduces uncertainties that affect the longitudinal and vertical estimates, while the lateral estimate remains unaffected due to its location on the centerline.

The fine calibration using GICP was successfully demonstrated on a sensor pair observing a building approximately 10 m from the vessel. The convergence region test showed that GICP converges for initial guess errors below approximately 1.5 m per axis, confirming that the pipeline provides a sufficiently accurate starting point for relative calibration between LiDARs. The absolute pose of each LiDAR on the vessel however still depends on the accuracy of the pipeline estimates.

In summary, this thesis shows that it is possible to automatically estimate the 6-DOF pose of an arbitrarily mounted LiDAR on a marine vessel using its internal IMU, frequency analysis and vessel maneuvers. The main remaining challenges are the CoR estimation, gravity compensation and the difficulty of producing sinusoidal maneuvers without dedicated maneuvering software.

Bibliography

[1] W. Wu, J. Li, C. Chen, B. Yang, X. Zou, Y. Yang, Y. Xu, R. Zhong, and R. Chen, "AFLI-Calib: Robust LiDAR-IMU extrinsic self-calibration based on adaptive frame length LiDAR odometry," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 199, pp. 157–181, 2023.

[2] E. Vinande, P. Axelrad, and D. M. Akos, "Mounting-Angle Estimation for Personal Navigation Devices," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 3, pp. 1129–1138, 2010, doi: 10.1109/TVT.2009.2034667.

[3] Y. Li and J. Ibanez-Guzman, "Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems," *IEEE Signal Processing Magazine*, vol. 37, no. 4, pp. 50–61, 2020, doi: 10.1109/MSP.2020.2973615.

[4] E. Demirbas, B. Boucher, M. Baker, J. Andrews, W. Cruz, S. Mueller, and S. Serna-Otalvaro, "A Review of Solid-State LiDAR Principles and Metasurface-Based LiDAR Sensors," *Sensors*, vol. 26, no. 1, Art. no. 1, 2026, doi: 10.3390/s26010001.

[5] N. Lopac, I. Jurdana, A. Brnelić, and T. Krljan, "Application of Laser Systems for Detection and Ranging in the Modern Road Transportation and Maritime Sector," *Sensors*, vol. 22, no. 16, Art. no. 5946, 2022, doi: 10.3390/s22165946.

[6] M. Khader and S. Cherian, *An Introduction to Automotive LIDAR*, White Paper SLYY150D, Texas Instruments, 2020. [Online]. Available: `https://www.ti.com/lit/wp/slyy150d/slyy150d.pdf`

[7] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, Ltd., 2021.

[8] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, "Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review," *Sensors*, vol. 21, no. 6, Art. no. 2140, 2021, doi: 10.3390/s21062140.

[9] M. Kok, J. D. Hol, and T. B. Schön, "Using Inertial Sensors for Position and Orientation Estimation," *Foundations and Trends in Signal Processing*, vol. 11, nos. 1–2, pp. 1–153, 2017.

[10] M. F. Abdel-Hafez, "On the development of an inertial navigation error-budget system," *Journal of the Franklin Institute*, vol. 348, no. 1, pp. 24–44, 2011, doi: 10.1016/j.jfranklin.2009.02.008.

[11] B. B. Worstell, S. K. Poppenga, G. A. Evans, and S. A. Prince, *Lidar Point Density Analysis: Implications for Identifying Water Bodies*. U.S. Geological Survey Scientific Investigations Report 2014-5191, 2014. [Online]. Available: `https://pubs.usgs.gov/sir/2014/5191/pdf/sir2014-5191.pdf`

[12] N. Muhammad and S. Lacroix, "Calibration of a rotating multi-beam lidar," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 5648–5653, doi: 10.1109/IROS.2010.5651382.

[13] R. Sultan and S. Greiser, "Time and Frequency Domain Analysis of IMU-Based Orientation Estimation Algorithms with Comparison to Robotic Arm Orientation as Reference," *Sensors*, vol. 25, no. 16, Art. no. 5161, 2025, doi: 10.3390/s25165161.

[14] G. Fukuda and N. Kubo, "Feasibility of Initial Bias Estimation in Real Maritime IMU Data Including X- and Y-Axis Accelerometers," *Sensors*, vol. 25, no. 21, Art. no. 6804, 2025, doi: 10.3390/s25216804.

[15] L. Brown and G. Lindberg, "Extrinsic calibration of multiple 2D Lidars using a Genetic Algorithm: A robust method to estimate the x, y and yaw coordinates of vehicle-mounted 2D Lidars using data acquired when moving," Master's thesis, Dept. Mech. Maritime Sci., Chalmers Univ. Technol., Gothenburg, Sweden, 2019.

[16] T. L. Nordgård and O. R. Skjølingstad, "Calibration of 3D lidars: A fully automatic and robust method for calibrating multiple 3D lidars using only point cloud data," Master's thesis, Dept. Mech. Maritime Sci., Chalmers Univ. Technol., Gothenburg, Sweden, 2020.

[17] E. Claesson and S. Marklund, "Calibration of IMUs using Neural Networks and Adaptive Techniques: Targeting a Self-Calibrated IMU," Master's thesis, Dept. Mech. Maritime Sci., Chalmers Univ. Technol., Gothenburg, Sweden, 2019.

[18] S. Das, B. Boberg, M. Fallon, and S. Chatterjee, "IMU-based Online Multi-lidar Calibration," in *Proc. 2024 IEEE Intelligent Vehicles Symposium (IV)*, Jeju Island, Korea, June 2024.

[19] J. Chang, D. Duan, F. Yu, Y. Zhang, and S. Fan, "A swing online calibration method of ship-based FOG-IMU," in *Proc. 2017 CPGPS Forum on Cooperative Positioning and Service*, Harbin, China, 2017, pp. 33–38.

[20] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *Proc. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan, 2004, pp. 2301–2306.

[21] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[22] F. Pomerleau, F. Colas, and R. Siegwart, "A Review of Point Cloud Registration Algorithms for Mobile Robotics," *Foundations and Trends in Robotics*, vol. 4, no. 1, pp. 1–104, 2015.

[23] R. Hänsch, T. Weber, and O. Hellwich, "Comparison of 3D interest point detectors and descriptors for point cloud fusion," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. II-3, pp. 57–64, 2014.

[24] M. Wahde, *Biologically Inspired Optimization Methods: An Introduction.* Southampton, UK / Boston, MA: WIT Press, 2008.

[25] E. Dahlin and H. Jonsson, "Semantic segmentation in marine environment: Using 2D spherical projection and convolutional neural networks," Master's the-

sis, Dept. Mech. Maritime Sci., Chalmers Univ. Technol., Gothenburg, Sweden, 2022.

[26] E. Lund, "Processing circuitry configured to determine information indicative of a position of a translational movement sensor on a marine vessel," U.S. Patent Application 20250010962, Jan. 9, 2025.

[27] RoboSense, "Airy: The World's First 192-Beam Hemispherical Digital LiDAR," [Online]. Available: `https://www.robosense.ai/en/IncrementalComponents/Airy`. [Accessed: 7-Apr-2026].

[28] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Robotics: Science and Systems V*, Seattle, USA, 2009.

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY