

CONFIDENTIAL

Drift-free Localization of Ship Hull Cleaning Robots on Flat Bottoms

A Sonar-based Framework

T.P.M. van Bergen

Master of Science Thesis



CONFIDENTIAL

Drift-free Localization of Ship Hull Cleaning Robots on Flat Bottoms

A Sonar-based Framework

MASTER OF SCIENCE THESIS

For the degree of Master of Science in Mechanical Engineering at Delft
University of Technology

T.P.M. van Bergen

March 23, 2020

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of
Technology



The work in this thesis was supported by Fleet Cleaner B.V. Their cooperation is hereby gratefully acknowledged.



Copyright © Cognitive Robotics (CoG)
All rights reserved.



Abstract

Fleet Cleaner B.V. is a Dutch company which deploys a wheeled hull-fixed Remotely Operated Vehicle (ROV) to clean biofouling off the Vertical Sides (VSs) and the Flat Bottom (FB) of the hull of seagoing cargo vessels, while berthed at the harbour. To ensure maximal cleaning coverage, accurate localization to keep track of the robot trajectory on the ship hull is essential, but also challenging. Contrary to the VS case, drift-free localization from motion sensors is not applicable on the FB. Instead, the current system relies heavily on operator input, who in turn relies on features (such as the quay wall, a line feature) appearing on imagery from a Forward Looking Sonar (FLS) to determine the robot pose. Human inattentiveness caused by the high workload of cleaning operations makes this method error-prone and sensitive to drift. In the future, Fleet Cleaner aims to develop an autonomous robot to relieve the operators from the task of localization and improve cleaning performance. As no previous work was performed on this matter, this thesis is a contribution to this vision, by developing a sonar-based framework enabling drift-free localization of a ship hull cleaning robot on FBs.

Before the conceptual phase, two objectives are formulated to achieve the thesis goal. The first is to provide a sonar-based algorithm measuring the line parameters of the quay wall, as lateral and heading reference. The second objective seeks to fuse the FLS-based measurements with motion sensors to achieve drift-free localization. From there, the operating conditions relevant to the systems are defined, and both functional and performance requirements are formulated, where the latter define metrics that measure the accuracy of the systems w.r.t. manually defined ground truths.

For the first objective, three candidate solutions for tracking the quay wall line parameters are selected. These are first implemented with recorded data in an iterative tuning procedure, then validated where the Total Least-Squares (TLS)-based line tracker with Gaussian filtering arises as a proof of concept. Although the algorithm is fast enough to function for each incoming sonar image, it does not fulfill the requirements for accuracy and robustness, a problem solvable by tuning with better data and the use of filters. For the second objective, also three sensor fusion algorithms are selected. Analysis of their properties with regards to performance revealed the Extended Kalman Filter (EKF) as the most suitable solution, where tuning was performed heuristically. Validation tests again showed partial success, with

fast computation times and greatly improved but still insufficient accuracy of the FLS measurements, although localization drift was reduced within the required limits. Mainly, the poor predictions from auxiliary sensors degrade performance, but also the imperfect nature of manually defined ground truths exaggerate the errors. To improve and complete the localization system, it is recommended to implement trajectory optimization for an improved fit of estimated poses with sensory data, and Simultaneous Localization And Mapping (SLAM) based on weld line detection to reduce longitudinal drift.

Table of Contents

Acknowledgements	xiii
1 Introduction	1
1-1 Motivation	1
1-1-1 Background	1
1-1-2 Localization	3
1-2 Problem statement	4
1-2-1 Shortcomings	5
1-2-2 Literature review	7
1-3 Thesis scope	9
1-4 Goal, objectives and contributions	9
1-5 Methodology and outline	10
2 Operating conditions	13
2-1 Global operating conditions	13
2-1-1 Forward Looking Sonar	13
2-2 Operating conditions for quay wall tracking	17
2-3 Operating Conditions for FB Localization	18
2-4 Data limitations	20
3 Requirements	23
3-1 Conventions	23
3-2 Quay wall tracking	24
3-2-1 The quay-wall: a line feature	24
3-2-2 Functional requirements	25
3-2-3 Non-functional requirements	26
3-3 Drift-free localization	27
3-3-1 The quay wall: a reference for localization	27
3-3-2 Functional requirements	28
3-3-3 Non-functional requirements	29

4	Quay wall tracking	31
4-1	Image processing	31
4-1-1	The FLS: a rangefinder	31
4-1-2	Subfunctions	32
4-2	Segmentation	32
4-2-1	Inputs	33
4-2-2	Region of Interest	34
4-2-3	Noise filtering	37
4-2-4	Contrast	38
4-2-5	Segmentation	40
4-3	Line fitting	42
4-3-1	Range extraction	42
4-3-2	Line fitting algorithms	42
4-3-3	Hough Transform	43
4-3-4	RANdom SAMple Consensus	44
4-3-5	Total Least-Squares	45
4-4	Implementation	46
4-4-1	Setup	47
4-4-2	Tuning procedure	48
4-4-3	Total Least-Squares	49
4-4-4	RANdom SAMple Consensus	50
4-4-5	Hough Transform	50
4-5	Validation	51
4-5-1	Accuracy	51
4-5-2	Computation time	52
4-6	Discussion	55
4-6-1	Findings	55
4-6-2	Limitations	56
4-6-3	Recommendations	57
4-7	Conclusion	57
5	Drift-free Localization	59
5-1	Subfunctions for localization	59
5-2	State estimation	60
5-2-1	Motion model	60
5-2-2	Sensor model	63
5-3	Filtering	67
5-3-1	Methods	67
5-3-2	Extended Kalman Filter	71
5-4	Implementation	71
5-4-1	Setup	72

5-4-2	Tuning	72
5-5	Validation	74
5-5-1	Accuracy	74
5-5-2	Drift	76
5-5-3	Computation time	79
5-6	Discussion	79
5-6-1	Findings	79
5-6-2	Limitations	80
5-6-3	Recommendations	80
5-7	Conclusion	81
6	Closing remarks	83
6-1	Summary	83
6-2	Main conclusions	85
6-3	Recommendations	86
6-3-1	Both systems	86
6-3-2	Quay wall tracking	87
6-3-3	Drift-free localization	87
6-3-4	Line-based SLAM	88
6-3-5	Trajectory optimization	88
A	Sensory systems	89
A-1	Forward Looking Sonar	89
A-2	Odometry	90
A-3	Recorded data	90
B	Algorithms	95
B-1	Segmentation	95
B-1-1	Range extraction	95
B-2	Line fitting	96
B-2-1	Total Least-Squares	96
B-2-2	RANdom SAMple Consensus	96
B-2-3	Hough Transform	97
B-3	Extended Kalman Filter	98
C	General Arrangement	101
	Bibliography	103
	Glossary	111
	List of Acronyms	111

List of Figures

1-1	Render of the Fleet Cleaner robot cleaning the side of a ship hull while underwater.	1
1-2	Examples of the different types of vessels cleaned by Fleet Cleaner B.V.	2
1-3	Schematic of two-dimensional localization. Localization of a robot with local frame $\mathcal{B}$ amounts to defining the translation t_{AB} described by longitudinal and lateral displacements x_{AB} and y_{AB} along with the rotation angle θ_{AB} (or heading) w.r.t. a reference frame $\mathcal{A}$	3
1-4	On VSs, knowledge of the draft (depth of the FB, determined visually by operators) and depth offers absolute references for the lateral coordinate z . Gravity which points down (negative z -direction) is a useful absolute reference for heading ψ . In combination with speed measurements v , the robot positions can be estimated (red points) which result in a close representation of the true trajectory. In contrast, on FBs depth is constant across the surface, while gravity and the robot's rotation axis become collinear, rendering both superfluous. Instead, the current system relies on the operator to determine the robot's heading from a discrete set of options (forward, backward, right or left), while speed measurements estimate the traveled distance in that direction. Clearly, this is a misrepresentation of the true trajectory with drift as a result.	5
1-5	Example of a sonar image on a ship's FB. The line on the right is the quay wall, which serves as one of the main features from the environment used for navigation. The straight edge from the boundary between dark and light pixels is the bilge. .	6
1-6	Three-dimensional view of a model of a bulk carrier, depicting the features that can be observed on the VS and the FB areas.	7
2-1	A sound wave is propagated across the Field of View (FOV) of θ , elevation angle ϕ and range r . Then, reflections are projected back in the zero-elevated plane (red). This creates an image resembling an upper view of the environment.	14
2-2	Schematic of the homographic model. The reflection from a 3D point $P(r, \theta, \phi)$ is projected in the image plane (red line).	14
2-3	Illustration of the image formation of an FLS. A sound wave is propagated across the 3D space, and is project in the red plane	15
2-4	Left: Polar image. Right: Cartesian image. Notice the beam interference, the speckle noise and the nonuniform illumination.	15

2-5	Illustration of the reflection strength decreasing with incidence angle β . The size of red lobes represent the strength of reflections, which is the strongest at the mirrored incidence angle and becomes weaker for increasing angles.	16
2-6	Example of an FLS image on the FB as displayed to the operator. The quay wall appears as a bright line on the right side and represents a crucial landmark for navigation. It is easy to infer that zoom in to $r_{stop} < 9$ [m] or steering the robot to the left would make the line disappear from the image. Clearly, there is a gap between the ship and the edge of the FB at the bilge (approximate 5 [m]). . . .	18
2-7	Schematic of the robot's motion model, which depends on steering angle α , forward speed v (defined as the minimum between front, right and left wheel speeds v_f , v_r , v_l as justified in Section 2-3) and wheelbase l . The change in heading depends on the instantaneous centre of rotation ICR and the robot's forward speed. . . .	19
3-1	Illustration of the reference frames: sonar (red), local robot frame (black) and the global ship frame (green). Notice that ship frame is flipped w.r.t. to the robot frame. Indeed, when on the FB, the robot is upside down.	24
3-2	Convention for local axes of rotation: roll (x), pitch (y) and yaw(z). This is based on the aeronautic convention.	24
3-3	Normal form of the infinite line representation, described by normal distance ρ and orientation θ with respect to the sonar's local reference frame (red coordinate frame).	25
3-4	In general, the majority of a ship's bottom surface is known to be flat (area enclosed by green dotted line). Furthermore, its length L and breadth B are specified in the General Arrangements (GAs).	27
4-1	Flow chart of the quay wall detection algorithm, translated polar image I_{polar} to a line measurement (ρ_S, θ_S) relative to the sonar frame. The algorithm consists of two stages: image segmentation and line fitting.	32
4-2	Illustration of polar image parameters, where each grid cell corresponds to a pixel. For target detection, any pixel (x, y) in the image frame (blue frame) can be related to polar coordinates (r, t). The number of rows M varies according to the maximal viewing range r_{stop} , which is chosen by operators in practice.	33
4-3	Schematic of the geometrical parameters considered for quay wall detection. For a prior line (ρ^S, θ^S), a prediction of the range $\tilde{r}_n$ than would be ideally measured at beam angle t_n can be obtained, along with the incidence angle β_n between the beam b_n and the quay wall.	35
4-4	Illustration of the Region of Interest (green square), described by its top left corner (r_{lb}, t_{lb}) and bottom right corner (r_{ub}, t_{ub}). It is defined by first defining the minima and maxima of range and angle measurements for a given prior line (black line). Then, an image mask along the line (dark gray) is used to ensure only pixels in the neighbourhood of the quay wall are used for subsequent processing.	36
4-5	Examples of non-filtered and filtered images. In order to clearly see the differences between methods, the kernel size of the filters was set to a large value of $F = 21 \times 21$ pixels for visual purposes.	38
4-6	Comparison of contrast enhancement techniques for low contrast images, and their corresponding intensity histograms. The red line represents the cumulative histogram, where a steep line suggests the dominance of a particular intensity level, causing low contrast.	39
4-7	Contrast enhanced images. The Contrast Limited Adaptive Histogram Equalization (CLAHE) image was enhanced with parameters $C_{clip} = 0.75$ and tile size $H = 8 \times 1$ pixels and the Weng filtered image was obtained with standard values $c = 0.1$ and $r = 1.7$	40

4-8	Representation of the ideal result from the segmentation on the Region of Interest (ROI) of the image from Figure 2-4. Left: ROI of the original image. Right: The background pixels are highlighted in red, the quay wall pixels in green.	41
4-9	Illustration of the working principle of the Hough Transform. For each point (x_i, y_i) in the Cartesian space, the set of lines (ρ_i, θ_i) through that point are determined according to ??, yielding a sinusoidal curve in the Hough space. Through this transformation, the line through all points in the Cartesian space becomes the intersection of the curves in the Hough space.	43
4-10	Illustration of RANDOM SAmple Consensus (RANSAC) algorithm. (1) Sample selection. (2) Line model proposal. (3) Line model fitting. (4) Consensus selection. (5) Best fit selection.	44
4-11	Illustration of the TLS method, with candidate line (ρ, θ) , detection points and their respective orthogonal distance to the line (red lines).	45
4-12	Illustration of the sonar data used for tuning and validation, where each image is the first image of the indicated set. The training set (upper row) was recorded on three different ships (Ship A, B and C). The validation set was recorded on three other ships (Ship D, E and F).	47
4-13	Results of the validation test for the TLS-based quay wall tracker, with plots of the estimated quay wall parameters along with ground truths and the corresponding errors. Furthermore, example images of line tracking for each data set are shown for visual purposes.	54
5-1	Schematic of the robot's motion model, which depends on steering angle α , forward speed v (defined as the minimum between front, right and left wheel speeds v_f, v_r, v_l as defined in Equation (2-6) and wheelbase l . The change in heading depends on the instantaneous centre of rotation ICR and the robot's forward speed. . . .	61
5-2	Illustration of the local position (x, y) covariance of the robot. Clearly, the global uncertainties σ_{xx} and σ_{yy} depend on the local uncertainty leads of the robot's displacement and its orientation ψ	62
5-3	Illustration of localization using quay wall tracking. Clearly, the measurements ρ^S and θ^S require some conversion before obtaining the robot lateral and heading position y_R and ψ_R . First, measurements from the sonar are related to the quay-relative frame $\mathcal{Q}$. Then, using the offset d_{QB} , these can be related to the ship frame $\mathcal{B}$. Finally, measurements from the sonar frame are translated to the robot frame by using the longitudinal offset l_{RS}	63
5-4	Illustration of the line uncertainty model as defined in [75]. Clearly, depending on the sign of orientation error $\Delta\theta$ (green dotted lines), errors in ρ^S with different magnitudes arise (red dotted arrows). Furthermore, the individual range discrepancies $\rho_i - \rho^S$ for each scan point also contribute in that error.	65
5-5	Illustration of the Kalman filter's optimal estimation in the one-dimensional case. First, a state estimate $\hat{x}_k$ is predicted from a previous estimate and control input. Then, the measurement is used to optimize the state estimate. Figure from https://www.mathworks.com/videos/understanding-kalman-filters-part-3-optimal-state-estimator-1490710645421.html , accessed on 24 February 2020.	68

5-6	Illustration of the uncertainty propagation of the Unscented Kalman Filter (UKF) in comparison to an actual sampling and the EKF. Clearly, a set of Gaussian distributed particles (blue points) with given mean and covariance do not yield a Gaussian distribution after a nonlinear transformation $y = f(x)$. The EKF however, applies a linearization $P_y = A^T P_x A$ around a single point (magenta point, i.e. the mean of the prior estimate) yielding another Gaussian distribution, a poor estimate of the true posterior mean and covariance. The UKF on the other hand, uses multiple (sigma-)points (red points) which are propagated through the nonlinear map, resulting in a better approximation (green ellipse). Figure from [82].	69
5-7	Illustration of two consecutive iterations of the particle filter working principle. Given a distribution of particles, a prediction for each particles next state is made. In the update phase the particles weights are updated using the likelihood model. In the last step the particles are resampled according to the weights of the updated particles, and so on.	70
5-8	Illustration of the used data set for validation. The available navigational data is composed of the wheel speeds and steering angles that were measured throughout the recordings of data sets D1, D2, D3 and E1	72
5-9	Comparison between ground truth and EKF-filtered quay wall tracker estimation of ρ and θ . In comparison with the unfiltered measurements shown in Figure 4-13, one particularly interesting observation can be made: the ρ measurements are clearly less noisy, so much that the ground truth measurements are the ones appearing to be noisy.	75
5-10	Trajectories of data sets D1 and D2 along with the estimated quay wall position (dotted black line), as estimated by the localization system ("Operator"), odometry ("Odometry"), the unfiltered quay wall tracker ("Quay Wall Tracker") and the one with an EKF filter ("Kalman Filter"). Clearly, odometry is a mediocre estimator for long term estimations (traveled distance beyond 20 [m] in this case) although it does show a "smooth" trajectory should be expected, while the current system does not reflect the imperfections of the robot's approximately straight trajectories. The quay wall tracker was shown to be accurately tracking the position overall, although its apparent noisiness suggests mediocre short-term accuracy. With the EKF, a much smoother result is obtained.	77
5-11	Trajectories of data sets D3 and E1 along with the estimated quay wall position (dotted black line), as estimated by the localization system ("Operator"), odometry ("Odometry"), the unfiltered quay wall tracker ("Quay Wall Tracker") and the one with an EKF filter ("Kalman Filter"). As was shown in the previous figure, odometry has very irregular results, immediately drifting with D3 while for E1 it remains fairly accurately up to 30 [m] from the starting point, while the current system idealizes the straightness of the path.	78
A-1	Schematic of the steering angle encoder.	90
A-2	Odometry data for data sets D3 and E1.	92
A-3	Odometry data for data sets D3 and E1	93
C-1	Example of a standard General Arrangement, presenting a side view of the ship along with an upper view of its main decks. The best indication if the FB is found in the last row, generally depicting the tank top (i.e. the plating at the inner bottom of the ship) instead of the true FB. Moreover, these documents specify the main dimensions such as the ship length and breadth are given (these were blurred for confidentiality purposes).	101

List of Tables

1-1	Overview of the available sensors onboard the Fleet Cleaner robot along with their measured properties. Through the wheel encoders, wheel speed measurement for the front, right and left wheel are available (v_f, v_r, v_l). Also, in contrast with the auxiliary sensors the sonar and camera currently yield no quantifiable data and return images instead, justifying their separation in the table. *Roll and pitch are referenced to gravity while heading is unreferenced and is reported to drift as much as 20 [deg/h].	4
1-2	Overview of the most commonly encountered features on the FB surface, the range at which they are visible, their variability over time during a cleaning operation and a general description.	6
4-1	Overview of the parameters to be tuned for the selected algorithms in the conceptual design phase of the quay wall detector, along with start values. These were determined heuristically. It should be noted that TLS has no parameters requiring tuning.	48
4-2	Overview of the performance results after tuning the filtering and contrast subfunctions along with the TLS line fitting parameters.	49
4-3	Overview of the performance results after tuning the filtering and contrast subfunctions along with the RANSAC line fitting parameters. The parameters returning the best results are boldfaced.	50
4-4	Overview of the performance results after tuning the filtering and contrast subfunctions along with the Hough Transform (HT) line fitting parameters. The discarded filters are struck through.	51
4-5	Performance results of the TLS, HT and RANSAC quay wall trackers per data set. For almost all the data sets (except for F1), the TLS-based candidate returns the best results in terms of all three performance metrics, although by a close margin compared to TLS. Furthermore, by the same criteria it shows that RANSAC ranks second and HT shows the lowest overall performance of all three candidates. . .	52
4-6	Overview of the computation time for each of the subfunctions of each of the quay wall tracker candidates. Extracting the ROI, filtering the image and applying the threshold for segmentation all have negligible computation times ($t_c < 0.001$ [s]), indicated by the hyphens.	55

5-1	Overview of the parameters to be tuned for EKF motion and sensor models. Note that the first three parameters of the sensor models have no definite value since these are defined from scan points by Equation (5-28), which change for each incoming sonar image.	73
5-2	Performance results of the unfiltered TLS quay wall trackers (labelled as TLS) and the EKF-based system (labelled as EKF), per data set. Clearly, the EKF-based system shows improvements on all metrics compared to the unfiltered quay wall tracker.	74
5-3	Comparison of the drift lateral drift $EBU(y)$ and heading drift $EBU(\psi)$ between the different systems.	76
5-4	Computation times for the front-end and back-end of the localization system. On average, extracting quay wall tracker measurements requires $t_c = 0.023$ [s], leaving an margin average of 0.017 [s] for the EKF-based state estimator to compute the state of the robot. Conveniently, with a negligible computation time of $t_c < 0.001$ [s], it falls within this margin with ease.	79
A-1	Specifications of FLS used by the Fleet Cleaner robot.	89
A-2	Overview of the relevant robot dimensions.	90
A-3	Overview of specifications of the wheel encoders. The [ppr] unit stands for pulse per revolution.	90
A-4	Overview of the data used for the tuning and validation tests for the quay wall detection system and the filtered state estimation. T is the total duration of the recording, Δt is the time step between frames and f is the corresponding frame rate, r_{max} is the max viewing range of the sonar image, dr is the range resolution, Q / S is the the quay wall position w.r.t. the sonar (L for left and R for right), Q / B is the the quay wall position w.r.t. the ship and ADS is the availability of navigational data.	91

Acknowledgements

I would like to thank my supervising university professor J.F.L. Goosen and daily supervisor O. Siebinga at Fleet Cleaner B.V. for their beneficial advice, assistance and guidance during the writing of this thesis.

Delft, University of Technology
March 23, 2020

T.P.M. van Bergen

Chapter 1

Introduction

1-1 Motivation

1-1-1 Background

The Company Fleet Cleaner B.V. is a company from the Netherlands which uses a Remotely Operated Vehicle (ROV) to remove fouling on the hull of sea-going vessels while they are berthed in the harbour (see Figure 1-1) for loading and unloading. Fouling, i.e. the accumulation of undesired biological material on ship hulls, is a well-known problem in the marine industry responsible for increased fuel consumption [1], emission of greenhouse gases [2] and contamination of local marine life [3]. Thanks to powerful magnets and motorized wheels, the robot is attached on the ship hull and maneuvers across its surface. Furthermore, three swing arms hold the cleaning heads with powerful Ultra High Pressure (UHP) water jets close to the hull to remove and recollect the undesired material, which is then transported through an umbilical hose towards a filtration system on a support vessel. From that same vessel the operators have the challenging task to plan and control the trajectory while also localizing the robot, as well as checking the installations on the support vessel, all contributing to a heavy workload. To ease the operations for human operators and enhance cleaning performance in the future, Fleet Cleaner B.V. envisions developing an autonomous cleaning robot executing tasks such as localization, planning and control all by itself.



Figure 1-1: Render of the Fleet Cleaner robot cleaning the side of a ship hull while underwater.

Operations The Fleet Cleaner robot can only maneuver on surfaces with limited curvature, therefore it specialises on cleaning the hull of large marine vessels (ranging between 100 [m] and 400 [m] in length and between 20 [m] and 60 [m] in breadth) such as bulk carriers or oil tankers (as shown in Figure 1-2) which typically have a high block coefficient [4]. Conveniently, this means that the shape of ship hulls approximate that of a rounded rectangular box, comprising of a majority of flat surfaces. Consequently, Fleet Cleaner B.V. currently offers two operationally distinctive cleaning services depending on which surface is to be cleaned:

- **Vertical Side (VS)** i.e. the vertical sides (port and starboard) of the hull.
- **Flat Bottom (FB)** i.e. the horizontal flat bottom of the hull.

Naturally, the goal is to remove as much fouling as possible during the operation. Since ships seek to minimize their downtime in the harbour, time efficiency plays a central role in reaching that goal. To this end, accurate localization of the robot is crucial as it offers a history of previously visited (clean) areas versus unexplored (fouled) areas. With this information, adequate path planning can be achieved to ensure complete cleaning coverage by grooming all fouled areas, but also improve time-efficiency by avoiding known cleaned areas to reduce cleaning overlap. As such, localization contributes to maximizing both cleaning coverage and cleaning speed for an improved cleaning performance [5]. Moreover, it greatly contributes in the accuracy of cleaning coverage reports Fleet Cleaner sends to clients.



Figure 1-2: Examples of the different types of vessels cleaned by Fleet Cleaner B.V.

Main challenge Unfortunately, localizing the robot to keep track of its trajectory on the ship hull is far from trivial. Indeed, the large dimensions of ships and the large surface areas to be cleaned (up to 400×60 [m²]) along with the low speed of the robot (0.15 [m/s] on average) cause these missions to inevitably exceed dozens of hours. To make operations humanly possible, the work is distributed between operators who take turns in twelve hour shifts. On top of that, most of the time the robot is out of sight as it is deployed underwater, making it impossible to locate it visually. Instead, the operator relies on data from onboard motion sensors and imagery from underwater cameras and a Forward Looking Sonar (FLS) to navigate. Based on this data and human input the position of the robot is estimated on a map, called General Arrangements (GAs), consisting of a 2D projection of the VS and FB (for an example refer to Figure C-1). As such, localization is one of the main challenges of operations and greatly contributes in the complexity of cleaning activities for operators.

General approach Before any cleaning operation, the robot is attached to the ship's VS above water, from where navigation can start. Given the elongated property of vessels (see Figure 1-2), the preferred approach is to imitate a lawn-mower trajectory along the length of the ship (or "horizontal slicing"), minimizing cumbersome U-turns. Ideally, this amounts to driving back and forth and laterally shifting after each stroke in order to cover the entire surface. For VS cleaning operations this is done by incrementally tracking a specified constant depth for each stroke, for which an autopilot system has been developed. However, this method is not applicable on FBs (a detailed explanation will be given in Section 1-2). To understand the underlying problem, the main concepts of localization will be laid out first.

1-1-2 Localization

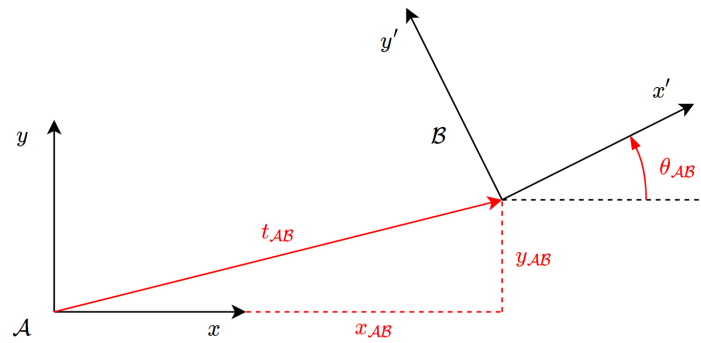


Figure 1-3: Schematic of two-dimensional localization. Localization of a robot with local frame B amounts to defining the translation t_{AB} described by longitudinal and lateral displacements x_{AB} and y_{AB} along with the rotation angle θ_{AB} (or heading) w.r.t. a reference frame A .

Main categories In essence, localization consists of determining the *pose*¹ of a robot's local frame with respect to a global reference frame in the environment [6]. In other words, it is a coordinate transformation between two frames defined by a translation and a rotation (see Figure 1-3). Various methods for localization have been developed over the years [7], which can be separated in two fundamentally distinctive types [8]: *relative* and *absolute* localization. This distinction rests on how information is acquired and used, causing crucial differences in how localization errors behave over time:

- **Relative:** relative localization rests on the principle of integrating motion (derivatives of position such as speed or acceleration) in order to obtain the pose incrementally. Inevitably, errors in motion estimates propagate due to the integration step and the dependence on the previously estimated pose. This causes the estimated pose to drift from the true pose as errors build up over time i.e. errors are **unbounded**. The most widely applied technique for relative localization is to integrate wheel speed and steering angle, a method known as odometry.
- **Absolute:** absolute localization rests on direct measurements of a position w.r.t. an absolute reference instead of derivatives like speed or acceleration. That is, measurements are independent of time and previous estimations. As such, errors do not build

¹The pose corresponds to both position and orientation w.r.t. to some reference.

up over time i.e. these are **bounded**. An example of absolute localization is landmark navigation, where the robot's relative pose w.r.t. landmarks with known positions is used to derive the robot's absolute pose.

Sensors According to [9], three functionalities are required for localization: the robot's ability to perceive its environment (perception) and its own movement (motion) and the ability to create a map of observations (mapping). This is achieved with internal and external measurements in order to extract meaningful information [10]. For that purpose, the Fleet Cleaner robot is equipped with a set of perceptual sensors from which distinctive landmark features of the environment can be observed by the operator and sensors sensing the robot's motion. An overview is given in the table below.

Table 1-1: Overview of the available sensors onboard the Fleet Cleaner robot along with their measured properties. Through the wheel encoders, wheel speed measurement for the front, right and left wheel are available (v_f, v_r, v_l). Also, in contrast with the auxiliary sensors the sonar and camera currently yield no quantifiable data and return images instead, justifying their separation in the table. *Roll and pitch are referenced to gravity while heading is unreferenced and is reported to drift as much as 20 [deg/h].

Sensor	Property	(Parameter)
Sonar	Landmarks	n/a
Camera	Landmarks	n/a
Wheel encoder	Wheel speed	(v_f, v_r, v_l)
Steering angle encoder	Steering angle	(α)
Swing arm encoders	Hull curvature	(κ)
Pressure sensor	Depth	(z)
Accelerometers (IMU)	Accelerations	$(\ddot{x}, \ddot{y}, \ddot{z})$
Gyros (IMU)	Angular rates	$(\dot{\phi}, \dot{\theta}, \dot{\psi})$
Orientation (IMU)	Roll, pitch and heading*	(ϕ, θ, ψ)

Filtering Due to imperfections and disturbances from the environment, sensors inevitably suffer from systematic or non-systematic errors [11]. Although relative localization presents unbounded errors, it is generally considered to have better short-term accuracy [12], whereas absolute measurements are superior on the long-term thanks to bounded errors [8]. Nonetheless, errors can be reduced by means of filtering [13] or fusing multiple sensory channels to obtain better estimates, also known as sensor fusion [14].

1-2 Problem statement

From the previous section, it is clear that localization of the robot is crucial for the successful completion of cleaning operations. Ideally, the localization system should return an accurate estimate of the robot's pose with minimal errors which remain bounded over time, regardless of its location on the ship hull (VS or FB). Unfortunately, this is not the case; operational differences between VS and FB oblige the use of two separate localization systems for each. While Fleet Cleaner reports the VS system to be satisfactory, this is not the case for the FB case. A complete explanation of the shortcomings will be given in the following section.

1-2-1 Shortcomings

Loss of references For VS navigation, the absolute references for localization are depth and gravity, measured by the depth sensor and Inertial Measurement Unit (IMU) respectively. Where these respectively represent absolute references for lateral position (depth z) and heading (yaw ψ) on vertical x - z -planes such as VSs (this is illustrated in Figure 1-4), they become superfluous on horizontal x - y -planes such as FBs, an area across which the depth is constant and the gravity vector coincides with the robot's yaw axis (see Figure 1-4). Although inertial data from the IMU and odometry from encoders are available, these only enable relative localization methods which suffer from error build-up over time, as explained in Section 1-1-2. Furthermore, hull curvature measurements from the swing arm encoders give no additional information on a flat surface like the FB.

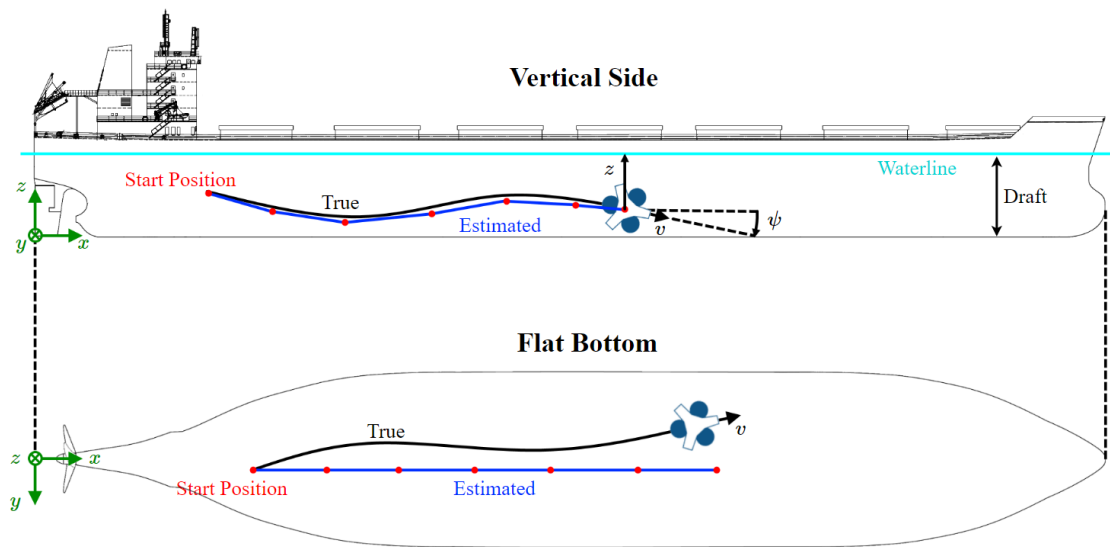


Figure 1-4: On VSs, knowledge of the draft (depth of the FB, determined visually by operators) and depth offers absolute references for the lateral coordinate z . Gravity which points down (negative z -direction) is a useful absolute reference for heading ψ . In combination with speed measurements v , the robot positions can be estimated (red points) which result in a close representation of the true trajectory. In contrast, on FBs depth is constant across the surface, while gravity and the robot's rotation axis become collinear, rendering both superfluous. Instead, the current system relies on the operator to determine the robot's heading from a discrete set of options (forward, backward, right or left), while speed measurements estimate the traveled distance in that direction. Clearly, this is a misrepresentation of the true trajectory with drift as a result.

Manual Localization With the measurements of depth and gravity obtained by sensors, a system which automatically tracks the pose of the robot on VSs is available. Based on this system, an autopilot function enables the robot to drive in straight strokes autonomously by tracking a specific depth. Due to the loss of absolute references, neither the automated localization nor the autopilot functions are available on FBs, therefore an alternative system is used. This system relies on the operator to manually track the robot trajectory on the GA while navigating. First, the operator has to determine if the robot is located on the VS or the FB and select the corresponding map of the GA. Then, the robot's initial position on the

GA is estimated by the operator, next the robot's heading is estimated by selecting one of four possible directions (forward, backward, right or left) and then the traveled distance along that direction is estimated by speed measurements from wheel encoders. From experience, performing these multiple tasks simultaneously over the course of dozens of hours, while taking shifts between different operators, is very challenging and especially prone to errors as illustrated in Figure 1-4.

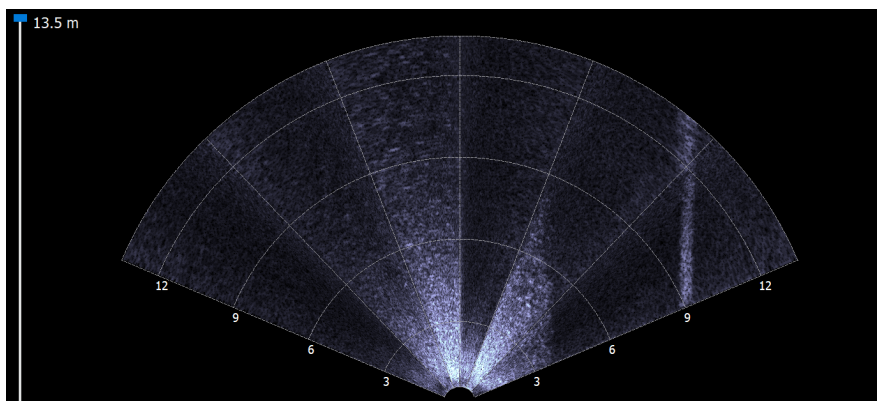


Figure 1-5: Example of a sonar image on a ship's FB. The line on the right is the quay wall, which serves as one of the main features from the environment used for navigation. The straight edge from the boundary between dark and light pixels is the bilge.

Sonar perception In practice, operators predominantly rely on imagery from the FLS instead of cameras to keep track of the robot pose while navigating on FBs. Indeed, the visibility range of cameras is drastically diminished (sometimes well below 1 [m]) due to the turbid water conditions, rendering them futile. In contrast, the FLS has far superior visibility range (up to 100 [m]) thanks to sound waves attenuating much less than light in turbid waters. Empirically, four landmarks are reported to be most commonly observed in sonar images [5]. Table 1-2 gives an overview of their characteristics and Figure 1-6 offers an illustration.

Table 1-2: Overview of the most commonly encountered features on the FB surface, the range at which they are visible, their variability over time during a cleaning operation and a general description.

Feature	Range	Variability	Description
Bilge	10-20 [m]	None	Characterized by an edge between high and low intensity pixels, most clearly visible at close range. Sometimes a stabilizing keel is present, appearing as line in the sonar image.
Weld lines	<10 [m]	Medium	Appear as (grid of) thin lines of high intensity pixels, mostly only visible at close distance. Sometimes become visible only after cleaning when the hull suffers from heavy fouling.
Fouling	10-20 [m]	High	Depending on the type, fouling is either easily visible (barnacles, mussels etc.) or hardly visible on the contrary (grass, slime). When fouling is clearly visible, clean strokes become apparent giving an indication of previously visited areas.
Quay	>50 [m]	None	The quay wall appears as a thick line of high intensity pixels, visible from high distances. Practically serves as a substitute for the waterline (lateral position and orientation) on FBs.

However, the sonar current only serves as a visual assistance tool for navigation. That is, no image processing is performed to track these targets in order to automatically cross-reference the visual information to the rest of the sensory data for position estimation. Instead, this task is fulfilled by operator, who interpret the data differently.

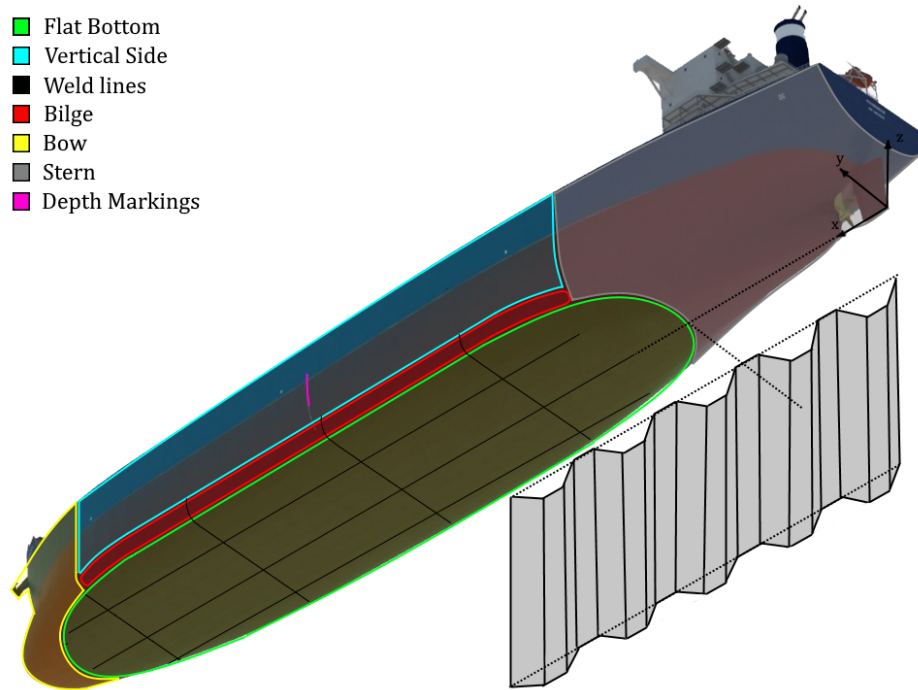


Figure 1-6: Three-dimensional view of a model of a bulk carrier, depicting the features that can be observed on the VS and the FB areas.

Incomplete Map As mentioned earlier, localization of the robot refers to GAs. Apart from the dimensions of the ship, these only show 2D side views of the ship and top views of the different decks down to the tank top², but not the actual Flat Bottom (see Figure C-1 in Appendix C). That is, there is no distinction between the curved areas (bilges³, bow⁴, stern⁵) and the flat parts of the ship's bottom (FB). Unfortunately, none of these features are depicted in GAs. Naturally, the assumption of 2D horizontal motion is only valid for flat parts of the ship's bottom, which are not unequivocally known beforehand.

1-2-2 Literature review

Previous work The main focus of research for Fleet Cleaner has been in developing and improving the localization functionality. Before the robot was operational, efforts focused on finding methods for VS cleaning operations, for which beacon-based solutions were advocated to achieve absolute localization under water [15] and above water [16]. The impractical

²The tank top is the plating above the flat bottom, creating a space for fuel or ballast water.

³The bilge is the curved edge between the VS and the FB.

⁴The bow is the forward part of the ship.

⁵The stern is the rear part of the ship.

installation and costs of the proposed systems led Fleet Cleaner to opt for more affordable alternatives by implementing wheel odometry, supported by a depth sensor, an IMU and visual assistance from cameras and an FLS. Follow-up research proposed a sensor fusion scheme combining IMU measurements to correct wheel slip and orientation drift errors by implementing an Extended Kalman Filter (EKF) [17]. With the introduction of the FLS, an obstacle detection scheme was proposed [18] where detection of the bilge and gratings was achieved, however the robot localization problem was not considered. Concurrently, a framework for underwater Simultaneous Localization And Mapping (SLAM) was proposed, for which one of its components, a Fourier-based algorithm for sonar-based odometry [19] was developed to estimate linear and rotational motion. However, although linear motion estimations were more accurate compared to wheel odometry, rotational shifts showed larger errors than IMU readings. Furthermore, this motion estimation method only enables relative localization (as explained in Section 1-1-2).

State-of-the-Art In the literature survey preceding this work [5], acoustic imaging devices such as FLSs prove to be the most adequate perceptual sensors for underwater settings, from which position and motion can be captured [20, 21]. Still, filtering and sensor fusion are often required to improve estimations due to the noisy nature of sonars. In underwater robotics, common practices are Kalman filtering [22] and variants [23], particle filters [6] or SLAM [24]. For unknown environments, which is often the case in the underwater setting [25], SLAM is the approach of choice for absolute localization, which incrementally constructs a map of observations of landmarks while concurrently localizing the w.r.t. to the mapped landmarks [24, 26]. Since accurate models of ship hulls are generally not available, this is also a commonly used approach for hull-relative localization based on features such as fouling, zink anodes, protrusions, weld lines or man-made features [27, 28, 29, 30, 31, 32, 33], or harbour walls as line features [34, 35], all of which are observable in sonar images. In contrast with Fleet Cleaner, the research community commonly uses hovering submersibles which are not constrained to the hull, meaning hull curvature is not an obstacle and access between VSs and FB through the bilge is possible. As such, the redundancy of the depth sensor on FBs can be addressed by performing vertical slices instead of horizontal slices for ship surveys, where above water structures and planar features can be used as checkpoints for re-localization [31, 32]. By extension, the pose-graph formulation of SLAM proves to reduce drift particularly well in comparison with dead-reckoning methods, by optimizing the entire robot trajectory to obtain the best fit to motion and observation measurements [27, 28, 32, 33]. This method also shows superior results when applied exclusively on the flat bottom of a ship using cooling pipes and zink anodes as references, as has been shown in the work by Johannsson et al. [27], although this was done on a relatively small area (1×2 [m²]). It should be noted that this represents the only attempt to tackle the problem of localization on FBs in the surveyed literature.

Research gap To the best of the author’s knowledge, the problem of reducing localization drift for ship-hull cleaning robots on a ship’s FB has not yet been visited altogether. Therefore, the goal of this thesis will be to *develop a proof-of-concept for a sonar-based framework enabling drift-free localization of a ship-hull cleaning robot on flat bottoms.*

1-3 Thesis scope

Localization on Flat Bottom During cleaning operations, navigation entails various responsibilities for operators of the Fleet Cleaner robot: localization, path planning and control. As no method has been previously proposed for the specific case, this thesis will exclusively consider the task of localization suited for FBs. Moreover, as it represents the major area of the bottom of the ship hull, only the Flat Bottom of the ship hull is considered, enabling the two-dimensional simplification of the localization problem.

Robot Perception Although the camera images are sometimes useful for FB navigation, the generally turbid water cause the images to be of mediocre quality most of the time, while FLSs are much less sensitive to turbid waters and have much superior range (as will be explained in the next chapter). Furthermore, as a constraint set by Fleet Cleaner, the installation of new sensors will not be considered. For this reason, this thesis will focus on adding sensory abilities solely from extracting measurements from FLS imagery.

Features and References Among the commonly used features for localization of the robot on the FB that were previously listed, the quay wall is the preferred reference [5]. Indeed, this linear feature can be observed from large ranges and just as the waterline for VSs, the quay wall offers an absolute lateral and heading reference for sonar measurements. For this reason, the quay wall as a reference will be the focus of this thesis. Furthermore, the absence of a longitudinally referenced measurement is considered acceptable since it does not impede the system to enable successful navigation in the VS case, according to Fleet Cleaner. For this reason, this thesis will not consider solutions for adding such a reference.

Implementation Given time constraints and the relatively rare occurrence and unpredictability of FB cleaning operations compared to the VS case, this work will not attempt to implement the proposed sonar-based framework in real-time. Instead, the implementation will be performed with self-written code on recorded data from past cleaning operations. As a company requirement, the code has to be written in the Python programming language.

1-4 Goal, objectives and contributions

Thesis goal From the problem statement in Section 1-2-1, the dominant pain point of Fleet Cleaner can be summarized by the loss of absolute localization references on a ship's FB disabling automated drift-free navigation capabilities. Instead, the current system relies on manual inputs from the operator, who bases his estimations on sonar imagery which is devoid of measurements while limited a-priori knowledge of the FB environment is available. As these factors present obstacle robot's autonomy, this justifies the goal for this thesis:

Develop a proof-of-concept for a sonar-based framework enabling drift-free localization of a ship-hull cleaning robot on flat bottoms.

Objectives To achieve the thesis goal, two objectives will be pursued. The first one contributes in turning the visual assisting tool that is the FLS into a sensor returning absolutely referenced measurements, with the quay wall as a reference. The second objective contributes in adequately incorporating the FLS measurements to the rest of the sensory arsenal of the robot to achieve drift-free localization:

1. *Provide a proof of concept for an FLS-based algorithm measuring the quay wall line parameters.*
2. *Provide a proof of concept fusing FLS measurements with auxiliary sensors to enable drift-free localization on flat bottoms.*

Contributions This thesis offers contributions to the development of Fleet Cleaner’s ship hull cleaning robot towards autonomy and the research field of FLS-based underwater localization of mobile robots in the ship hull environment, more particularly for the FB-case.

1-5 Methodology and outline

Methodology In order to achieve the presented objectives, this document will follow a structured approach. This methodology can be described by four distinct phases:

1. **Task clarification:** In this preliminary phase, the tasks involved to achieve the goal of this thesis are clarified. This is already partially done by formulating the scope and the goal of the thesis, along with the two corresponding objectives described in the previous sections. As a last step to this preliminary phase, a description of the operating conditions (i.e. practical constraints) and list of requirements relating to the objectives will be formulated.
2. **Conceptual design:** During the conceptual phase, insights from the previous phase are used to identify the fundamental problems to be solved. From there, the required subfunctions are identified and the existing field of knowledge is explored in order to find candidate solutions. Then, based on considerations w.r.t. the requirements, the most promising candidates are selected for further evaluation.
3. **Implementation:** Upon completion of the conceptual phase, candidate solutions are implemented throughout experiments in order to gather concrete results with regards to the performance. Prior to this, the algorithms are implemented in self-written code or built-in Python functions. From there, the parameters for each candidate algorithms can be tuned using the recorded data.
4. **Proof of concept:** In the final design phase, the best combination of subfunctions is selected, based on experiments run on recorded data from past operations in order to reflect the real-world situation. These experiments serve as a proof of concept to evaluate to what extent the requirements for the objectives have been fulfilled.

Document outline The methodology described above is adopted for both objectives, and defines the structure of this document. **Chapter 2** describes the practical constraints the

objectives of this thesis are subjected to, along with the limitation of the available data. Given the central role of the FLS in this thesis, the working principles for and weaknesses of the acoustic images from this sensor will be elaborated in details. In **Chapter 3**, a list of functional and non-functional requirements for both objectives is given. The first describes the allowed inputs and required outputs from each system, while the second specifies criteria that will be used for evaluation. **Chapter 4** goes through step 2 through 4 of the methodological procedure to deliver and evaluate the proof of concept for the first objective of this thesis. In the same manner, **Chapter 5** does the same for the second objective of this thesis. As a concluding chapter, **Chapter 6** compiles the findings and evaluates the fulfilment of objectives for this thesis and closes off with recommendations for future research.

As complements, this document contains three appendices. **Appendix A** gives the sensor specifications of the FLS and the auxiliary sensors, along with details on recorded data. **Appendix B** gives an overview of the self-written code that was used for implementing candidate solutions. Finally, **Appendix C** gives an example of a GA used for navigation.

Operating conditions

As was explained in the introduction, the goal of this thesis is to *develop a proof-of-concept for a sonar-based framework enabling drift-free localization of a ship-hull cleaning robot on flat bottoms*. As a preliminary part of the research, it is essential to establish the operating conditions of the Fleet Cleaner robot in the setting of Flat Bottom (FB) cleaning operations, as these have a strong influence on the choices to be made for the design of the system. As such, constraints to the problem at hand can be formulated and the research can be narrowed down. From there the requirements for the proposed localization system will be formulated in the next chapter, as a basis for the evaluation of the proposed framework and its subsystems in the concluding chapter of this thesis.

2-1 Global operating conditions

The Fleet Cleaner robot is subjected to constraints imposed by its own design, the environment it works in and the circumstances that FB cleaning operations entail. Some general conditions were already discussed in Chapter 1:

- The Fleet Cleaner robot is confined to the hull of the ship, and drives on the FB between the bilges, the stern and bow.
- The motion of the robot on the FB is non-holonomic and two-dimensional.
- The Forward Looking Sonar (FLS) serves as the only perceptual sensor.

2-1-1 Forward Looking Sonar

To perform localization, robots rely on perception and sensing motion with the help of sensors to make internal and external measurements. However, sensors have weaknesses caused by their own design and the environment they work in. Given its central role in this thesis, a general description of the FLS sensor and the perceptive limitations will be elaborated in more details. The rest of the sensors sensing motion will be discussed in Section 1-1-2.

General description

Working principle As explained earlier, FLSs are more adequate perceptual sensors than cameras for underwater environments, thanks to the property of sound propagating more efficiently than light [30]. The FLS is an active acoustic sensor which emits sound waves (i.e. insonification) into the environment. By analyzing the Time of Flight (ToF) of the reflections with knowledge of the speed of sound, the range ρ^S and bearing θ^S of an object i with respect to the sonar frame can be found [36]. Reflections from 3D space are outputted in a 2D acoustic image according to an orthographic approximation (Figure 2-2) [37] creating an "upper view" of the environment (Figure 2-1), enabling an operator to observe the presence of relief or bodies over much longer ranges than with cameras (up to 100 [m]).

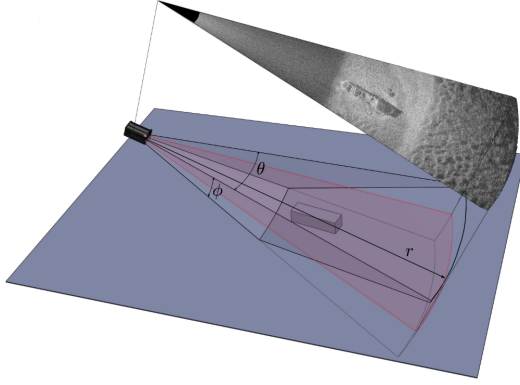


Figure 2-1: A sound wave is propagated across the Field of View (FOV) of θ , elevation angle ϕ and range r . Then, reflections are projected back in the zero-elevation plane (red). This creates an image resembling an upper view of the environment.

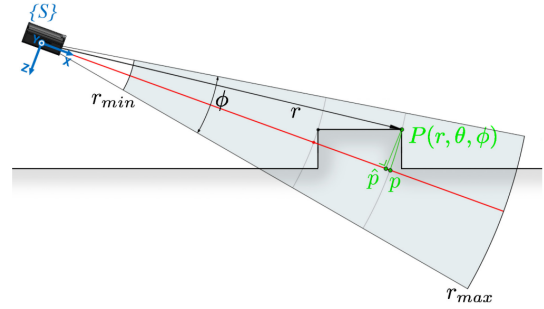


Figure 2-2: Schematic of the homographic model. The reflection from a 3D point $P(r, \theta, \phi)$ is projected in the image plane (red line).

Specifications The Fleet Cleaner robot uses the phased array Teledyne BlueView M-900-2250 Dual Frequency Series FLS [38]. This sonar has $N = 768$ sonar beams spanning across the FOV of $\Theta = 130$ [deg] for an elevation angle of $\Phi = 20$ [deg] detecting reflections at a range between $r_{min} = 0.60$ [m] and $r_{max} = 100$ [m]¹. A visualization of the FOV parameters is depicted in Figure 2-3. Furthermore, the sonar can operate at two different frequencies² (see sonar specifications in table A-1) with a trade-off between range resolution dr and maximal range of vision r_{max} . In practice, the long range mode of operation is preferred because of its superior range; therefore this thesis will only consider this mode of operation.

¹The optimal range is at 60 [m]. For larger ranges, time delay of echos become non-negligible.

²In the time of writing this thesis, only the long range operating frequency was functional, becoming the standard mode of operation.

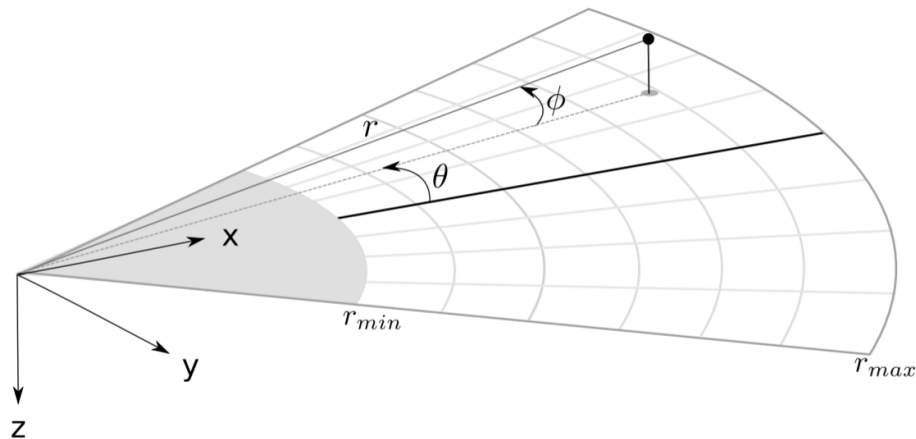


Figure 2-3: Illustration of the image formation of an FLS. A sound wave is propagated across the 3D space, and is project in the red plane

Image Output From the measurement of acoustic reflections, the FLS can output images. The native data returns echo intensities for each range increment r and angle increment t . These echo intensities are used to form images, where bright pixels represent strong reflections and vice versa. From there, two types of images can be returned: the polar images $I(r, t)$ spanning the range and bearing (r, t) space, and Cartesian images $I(x, y)$ spanning the (x, y) space. An example for each type is shown in Figure 2-4.

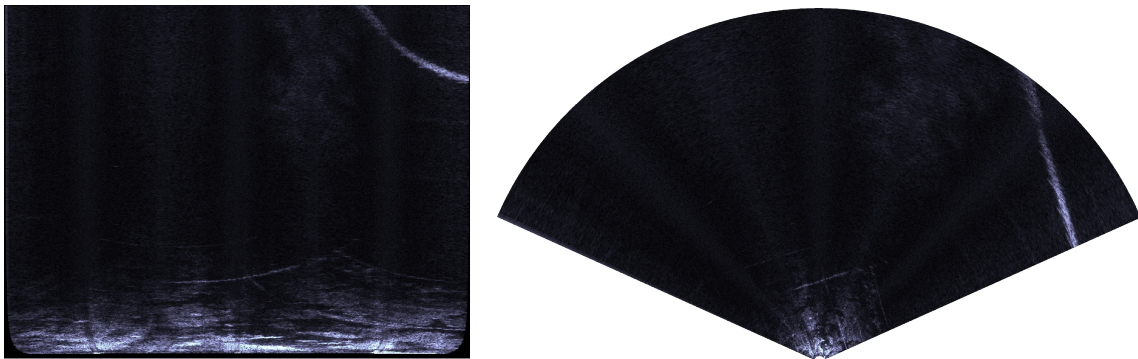


Figure 2-4: **Left:** Polar image. **Right:** Cartesian image. Notice the beam interference, the speckle noise and the nonuniform illumination.

Limitations

Transmission losses Similarly to light, one of the main drawbacks of acoustic signals in the aqueous medium is the phenomenon of transmission losses, although to a lesser extent. As the spherical wave front from the sonar expands through the medium, absorption of sound energy causes the acoustic intensity to decrease quadratically with range R in homogeneous media [36]:

$$I \sim \frac{1}{R^2} \quad (2-1)$$

For waves reflected back to their source, this loss become even greater [36]:

$$I \sim \frac{1}{R^2} \frac{1}{R^2} = \frac{1}{R^4} \quad (2-2)$$

Next to the range R , the incidence angle of a reflected beam i.e. the angle between the normal of the reflection surface and the incoming beam, also contributes in the reflection losses. For reflected sound waves, the strongest intensity is in the direction mirroring the incoming beam whereas it dissipates in deviating directions. In other words, the strength of reflections back to the source decreases with the angle of incidence. This is illustrated in Figure 2-5.

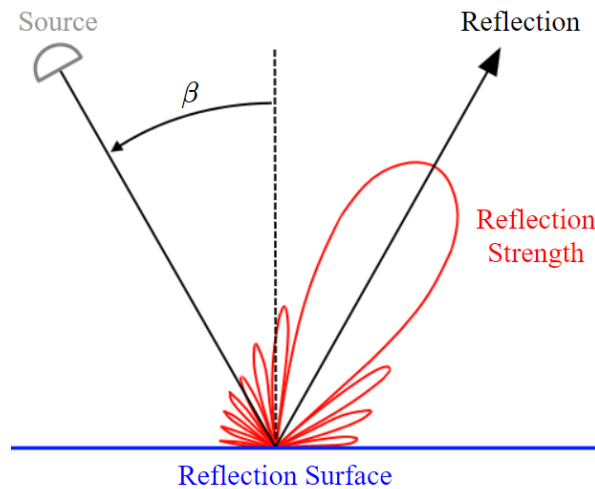


Figure 2-5: Illustration of the reflection strength decreasing with incidence angle β . The size of red lobes represent the strength of reflections, which is the strongest at the mirrored incidence angle and becomes weaker for increasing angles.

Disturbances Although sonars are largely preferable to optical cameras underwater, the use of an acoustic sensor still has limitations [5, 18, 19], all degrading the quality of images and potential measurements:

- **Speckle noise:** this type of noise is characterized by its speckle pattern of random intensity and is produced by the mutual interference of a set of wave fronts. It is a multiplicative noise, meaning it is in direct proportion to the local intensity of the area, leading to the typically low Signal to Noise Ratio (SNR) for sonar sensors. As such, it is hard to distinguish the speckle noise from the real data, making it challenging to remove without affecting the original data [39].
- **Nonuniform resolution:** As mentioned earlier, the acoustic data registered by the FLS returns an echo intensity for each range and bearing increment, spanning the polar (r, t) space. Translation to the Cartesian image yields more sparse measurements for increasing ranges, leading to a degraded image resolution for distant targets. This is best seen in Figure 2-3, where the grid cells dilate in the Cartesian space for increasing ranges.

- **Nonuniform image illumination:** the M-900-2250 FLS includes a Time Varying Gain (TVG) mechanism³ that compensates transmission losses for increasing ranges. According to BlueView, this is to optimize the sonar imagery over a wide range of sonar uses with respect to the dynamic range of the system and the noise floor [40]. As such, identical objects at different ranges return the same pixel intensities [37]. In practice, objects tend to show higher intensities as they are approached. Unfortunately, the BlueView software does not facilitate adjustment of the gain.
- **Noise at beam boundaries:** the FLS sensor is made of multiple transducer arrays which are mounted together. As there is a slight overlap between transducer beams, vertical beams of noise appear at the boundaries between different beams (Figure ??), dividing sonar images in six main beam regions. This effect not only observed on overlapping beams but also on the left and right image boundaries. Furthermore, similar targets sometimes appear different depending on which of the beam regions it is viewed from. For example, in Figure 2-4 the quay wall is visible in two of these beam regions, where compared to the first it appears slightly thicker in the second beam region.

2-2 Operating conditions for quay wall tracking

Fleet Cleaner's philosophy is to offer ship hull cleaning services while ships are berthed in the harbour to unload and load goods to minimize downtime, which most commonly takes place along a quay wall. This is convenient, since large planar structures by nature return strong reflections and are therefore very suitable for detection by acoustic sensors [34]. Indeed, these appear as a distinctive illuminated line on sonar images (see Figure 2-6), a very useful reference for navigation on the FB in practice.

Operator tasks Assuming that the cleaning operation is performed on a ship berthed along a quay wall, three conditions are to be fulfilled by the operator for the quay wall to be visible in FLS images in order to be tracked as a reference for navigation:

- The robot is located on the FB between the bow, the stern and bilges
- The robot is directed such that the quay wall falls within the sonar's FOV
- The maximum viewed range r_{stop} is selected such that the quay wall is in range

Furthermore, as mentioned in Section 1-1-1, the preferred approach is to imitate a lawnmower pattern along the length (x -axis) of the ship. As such, for the majority of the time the operator drives the robot along the quay wall while attempting to keep it at a constant distance in order to clean in straight lanes. This means that the quay wall appears as a thick vertical line on the right or left side of the image the majority of the time.

Quay wall As the ship lies along the quay wall, it is assumed that the quay can be represented by an infinite vertical plane parallel to the x - z -plane of the ship with a certain offset. In the 2D workspace of the robot, the quay wall becomes an infinite line feature, described by an

³Its standard value is set at 0.10 [dB/m] by the BlueView software.

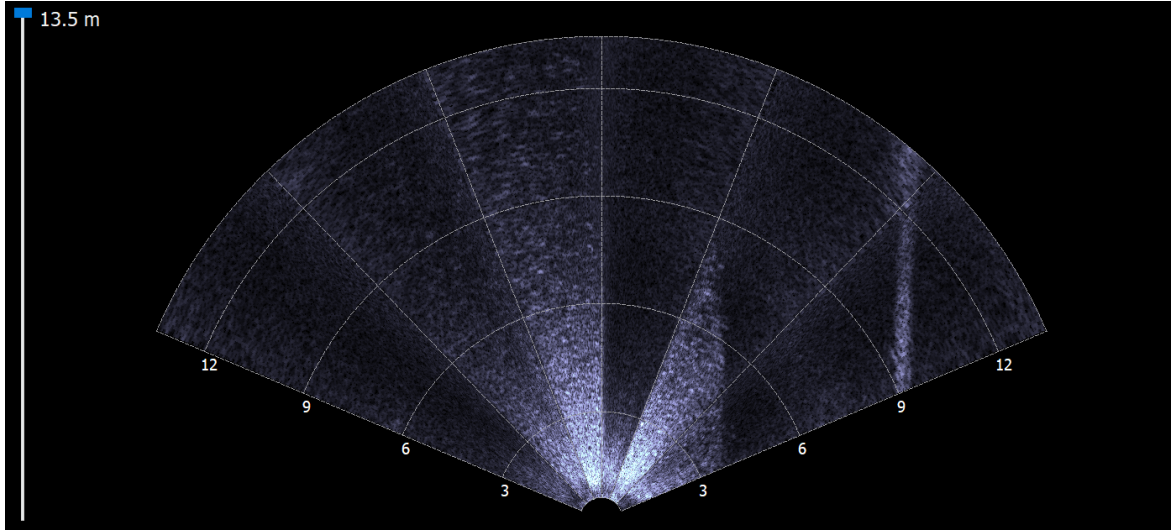


Figure 2-6: Example of an FLS image on the FB as displayed to the operator. The quay wall appears as a bright line on the right side and represents a crucial landmark for navigation. It is easy to infer that zoom in to $r_{stop} < 9$ [m] or steering the robot to the left would make the line disappear from the image. Clearly, there is a gap between the ship and the edge of the FB at the bilge (approximate 5 [m]).

offset d_{QB} in the y -direction (see Figure 1-6). This offset is always larger than half of the ship's width B , meaning there is always gap between the quay wall and the ship's Vertical Side (VS), appearing as a dark region between the bilge and the quay wall (see Figure 2-6). Furthermore, since sonar beams are reflected back by the quay wall, strong reflections cannot originate from anywhere past the wall. As such, it is typically surrounded by dark regions on both sides, as shown in Figure 2-6.

2-3 Operating Conditions for FB Localization

Initial conditions As the robot is initially attached to one of the VSs and the bilge cannot be crossed due to keels or high hull curvature, the FB is accessed underwater through the bow or the stern. As soon as the operator estimates the FB is reached, he reinitializes the position of the robot on the General Arrangement (GA), and its heading: forward, backward, left or right. Unfortunately, GAs commonly depict the tank top (see Appendix C) instead of the FB and its features, making the initial and ensuing pose estimates prone to errors.

Flat Bottom Since operations always start on VSs, the first step of localization is to recognize when the robot arrives on the FB of the ship hull. It is characterized by its flat and horizontal surface and is delineated by the curved and tilted edges around the bow, the stern or the bilges (Figure 1-6). From these properties, three indicators exist for the FB to be recognized in theory:

- Hull curvature κ , i.e. zero curvature implies a flat surface

- Depth z , i.e. a depth corresponding to the a-priori known draught⁴
- Inclination (ϕ, θ) , i.e. zero roll and pitch imply a horizontal surface

However, in practice the ship is not always exactly horizontal, where *trim*⁵ of up to 3 [m] for ships as long as 300 [m] can be observed, nor does its depth remain constant due to loading and unloading. Also, the frequent detachments of cleaning heads often deteriorate curvature measurements. All these factors make the FB more difficult to recognize in practice.

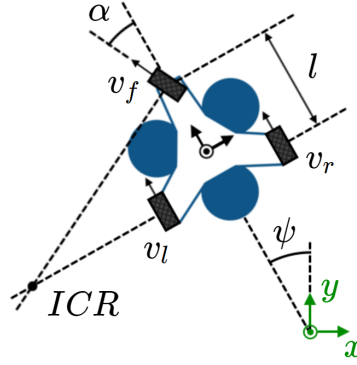


Figure 2-7: Schematic of the robot's motion model, which depends on steering angle α , forward speed v (defined as the minimum between front, right and left wheel speeds v_f, v_r, v_l as justified in Section 2-3) and wheelbase l . The change in heading depends on the instantaneous centre of rotation ICR and the robot's forward speed.

Current localization system As mentioned in Section 1-2-1, once the FB is considered to be reached, an initial pose of the robot is estimated and from there the localization system relies on the wheel encoders to estimate the motion of the robot along the direction (heading ψ) that was selected by the operator. This is done using a basic transition model for each time instance k :

$$x_k = x_{k-1} + v_k \cdot \cos \psi \cdot dt \quad (2-3)$$

$$y_k = y_{k-1} + v_k \cdot \sin \psi \cdot dt \quad (2-4)$$

$$\psi_k = \psi_{k,op} \quad (2-5)$$

Where v_k is the estimated forward speed of the robot, dt is the time step and $\psi_{k,op}$ is the robot heading as defined by the operator (chosen from discrete set of values $\{0, \frac{\pi}{2}, \pi, -\frac{\pi}{2}\}$ [rad] corresponding to forward, left, backward and right respectively). To estimate the forward speed of the robot, the current method selects the minimum value of the encoder's speed measurements for each wheel:

$$v = \min(\{v_f, v_l, v_r\}) \quad (2-6)$$

The reason for this choice is to alleviate the common over-estimation of a wheel's actual speed w.r.t. to its driving surface, causing the robot to appear to travel larger distances than in

⁴The draught of a ship is the distance from the waterline to the FB.

⁵The trim is defined as the difference between the draft at the bow and the draft at the stern, equivalent to a pitch angle.

reality (more on this in the next paragraph). As a consequence, the obvious limitation of this system is that it does not take steering into account, assuming strictly rectilinear estimated trajectories while in reality these are rather curvilinear. Nonetheless, as mentioned in the previous section, most of the time the robot is navigated so as to drive parallel to the quay wall; in practice this leads to approximately straight trajectories.

Disturbances In the previous chapter, it was mentioned that odometry from wheel speed and steering angle is known to suffer from error build-up. This is for most part due to wheel slip, caused by the ship's slippery surface and the heavy umbilical hose pulling the robot down when located on VSs [17]:

- **Longitudinal slip:** difference between tangential speed of the rotating wheel and linear speed of the center of the wheel with respect to the contact surface.
- **Lateral slip** (or slip angle): angle between linear motion of wheel and direction the wheel is pointing at.

Due to these discrepancies, longitudinal and lateral slip generally cause over-estimation of wheel speed and effective steering angle α (and in turn the change of heading), respectively. It should be noted however that, according to Fleet Cleaner, much less umbilical-related disturbances are experienced on Flat Bottom since the umbilical hose mostly pulls in the vertical direction which, in the FB case, is not in the robot's plane of motion. Also, in the time of writing this thesis, the Inertial Measurement Unit (IMU) was reported to be subjected to strong magnetic disturbances from actuators in the robot's manifold causing measurements to be excessively noisy⁶ for proper navigation [5].

2-4 Data limitations

As mentioned in Section 1-3, the scope of this thesis dictates that the implementation is to be performed on recorded data rather than real-time testing. However, during the writing of this thesis a large volume of data was already present, therefore the focus was initially on setting up a functioning platform for implementation. At the moment acquisition of additional data was intended, no opportunity presented itself for additional data recording of relevant FB operations. Furthermore, the data from both motion sensors (henceforth called navigational data) and the FLS data were found to have limitations, limiting even further the volume of useful data for implementation.

Limited volume Although Fleet Cleaner currently executes cleaning operations on a weekly basis, relevant image and navigational data is limited due to the relatively low prevalence of FB compared to VS cleaning operations. Furthermore, the navigational data is also limited due to faulty software that was only recently corrected, rendering a portion of the available data non adequate (time steps exceeding 10 [s]). To limit the large amount of data due to the long duration of operations, the frequency of the data stream recording is limited to 1 [Hz].

⁶During the writing of this thesis, the IMU was subject to design changes of the robot to solve this issue.

Non-uniform and non-synchronised data Although the FLS has a range of up to $r_{max} = 100$ [m], the sonar imagery is often zoomed in at much lower ranges, where the preferred range of view is rarely chosen above $r_{stop} = 50$ [m]. Unfortunately, due to software practicalities the sonar imagery can only be recorded at the resolution as it is being displayed, leading to images with varying resolutions dr and dt . Furthermore, the frame rate of the sonar recordings are variable (roughly between 1 - 13 [Hz]), leading to varying time steps between frames (ranging between roughly 0.07 - 1.00 [s]). That is, it is not synchronized with the rest of the data which is recorded at 1 [Hz]. Given the slow dynamics of the robot, it is assumed that interpolation of navigational data to synchronise it with sonar data yields negligible errors.

Inaccessible raw data As explained in section 2-1-1, the sonar outputs an array of echo intensities across ranges r and beam angle increments t , resulting in a polar image. Unfortunately, to protect its product, Teledyne's software does not permit the access of the raw data, which is encoded in a self made data type (.son). Instead, BlueView provides a function which translates the raw data to either polar or Cartesian images.

Chapter 3

Requirements

In the previous chapters the thesis goal, the objectives and an overview of the operating conditions were given. The purpose of this chapter is to set up a list of requirements for the subsystems which will be used for drift-free localization on Flat Bottoms (FBs). For each, two types of requirements will be considered. First, the functional requirements will describe the allowed inputs and the desired outputs from the subsystems of the proposed system. Then, the non-functional requirements will describe the qualitative and quantitative requirements. These will serve as metrics for the evaluation of the systems in later chapters. The proposal of this thesis will comprise of two parts: first, the requirements for the quay wall tracker will be discussed, then the requirements for the drift-free localization system.

3-1 Conventions

Reference frames As was mentioned in Section 1-1-2, localization amounts to finding the relation between a local and a global reference frame, described by a translation and a rotation. Therefore, it is important to identify and define the reference frames of interest in this thesis:

- **Ship frame $\mathcal{B}$:** This is considered the *global* frame, fixed to the ship with the origin in the middle, at the back and at the greatest depth of the flat bottom (see Figure 1-6).
- **Quay wall frame $\mathcal{Q}$:** This is the reference to which the quay wall detector relates its measurements. It has an offset d_{QB} in the y -axis with respect to ship frame $\mathcal{B}$ (see Figure ??).
- **Robot frame $\mathcal{R}$:** this is the *local* frame, fixed at the centre of the robot body (see Figure 3-1).
- **Sensor frame $\mathcal{S}$:** the reference frame from which the sensor draws measurements. It is attached to the robot frame with known translation and rotation w.r.t. the robot frame (see Figure 3-1).

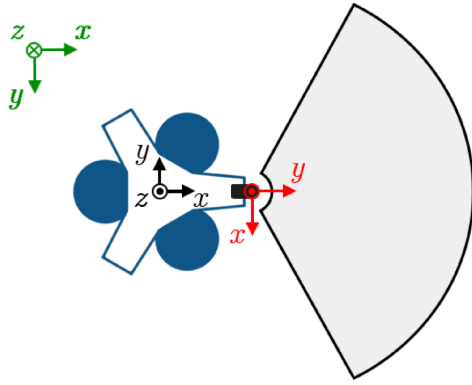


Figure 3-1: Illustration of the reference frames: sonar (red), local robot frame (black) and the global ship frame (green). Notice that ship frame is flipped w.r.t. to the robot frame. Indeed, when on the FB, the robot is upside down.

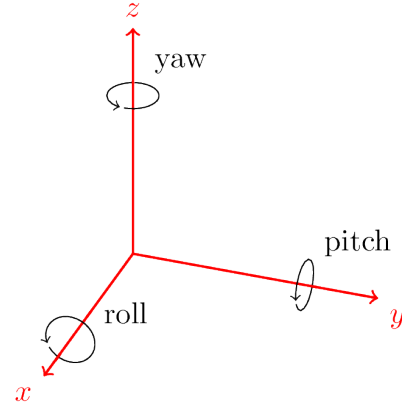


Figure 3-2: Convention for local axes of rotation: roll (x), pitch (y) and yaw (z). This is based on the aeronautic convention.

Notation To determine the pose w.r.t a reference frame, specifying the referenced frame is useful to avoid ambiguity. This is done by using the superscript notation to indicate the frame of reference and the subscript for the located frame. E.g. the robot position w.r.t. the quay wall frame is $x_{\mathcal{R}}^{\mathcal{Q}}$. When subscripts and superscripts are left out for (x, y, ψ) , ship-relative localization of the robot frame is of interest.

3-2 Quay wall tracking

As stated in Section 1-3, the goal of this thesis is to achieve absolute localization by using the quay wall as a landmark line feature. Incorporating this functionality into the robot involves two possible solutions that are closely related. The first is called object detection, which consists of recognizing and locating objects in images, without prior knowledge. Upon detection, the prior location can be used to keep track of an object while it moves locally across subsequent frames; this is called object tracking. As a closely related thesis prior to this one considered the issue of detecting linear features (bilge edges) while omitting tracking [18], this work will focus on the latter and assume prior knowledge from an initial detection is available. This quay wall tracking system represents the front-end of the proposed system, turning sonar images into absolutely referenced measurements to enable the back-end of the system to achieve drift-free localization.

3-2-1 The quay-wall: a line feature

The chosen landmark upon which sonar measurements will be based is the quay wall. This is convenient, as large planar structure like harbour walls and dam walls appear as thick illuminated lines in sonar images [34]. In other words, measuring the quay wall as a line feature is equivalent to finding the parameters describing the line (as shown in Figure 3-3).

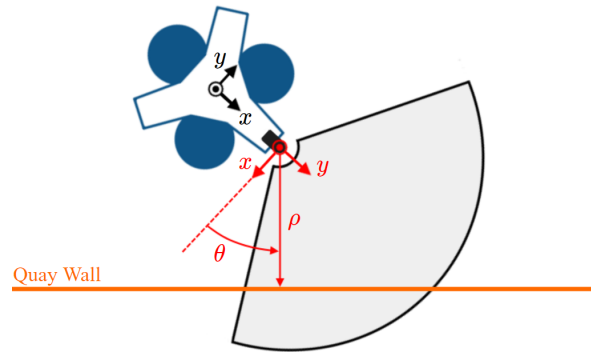


Figure 3-3: Normal form of the infinite line representation, described by normal distance ρ and orientation θ with respect to the sonar's local reference frame (red coordinate frame).

Infinite line representation Regardless of the length of the ship, it is assumed that the quay wall extends well beyond its extremities. Based on this, it is fair to define the quay wall as an infinite line feature with no endpoints [41], described by the normal distance ρ and orientation θ of the normal to this line:

$$L = \begin{bmatrix} \rho \\ \theta \end{bmatrix} \quad (3-1)$$

This polar representation, also called the normal form, is preferred to the slope-intercept line representation defined by $y = mx + b$, as its parameters (ρ, θ) directly relate to the robot's relative pose w.r.t. the quay wall, i.e. perpendicular distance and orientation (Figure 3-3). Moreover, it is more robust as no singularity is encountered for vertical lines, where slope m would explode for the slope-intercept form.

3-2-2 Functional requirements

The functional requirements are described by the allowed inputs and the desired outputs from the system.

Inputs Upon the assumption that the operator ensures the presence of the quay wall in sonar images as explained in Section 2-2, the sonar-based quay wall tracking scheme can use the following items as inputs:

- An image from the Forward Looking Sonar (FLS) (in polar or Cartesian form)
- A prior line estimate from a previous measurement $(\rho_{prior}^S, \theta_{prior}^S)$

Since it was mentioned in 2-2 that it is the operator's task to assure the presence of the quay wall in the sonar image, instances of the quay wall disappearing and reappearing in the Field of View (FOV) of the recorded images will be omitted.

Outputs From the previously listed inputs, the sonar’s quay wall tracker should be able to extract the line appearing in images and express the configuration of the quay wall w.r.t. the sonar frame origin:

- Normal distance ρ^S of the quay wall relative to the sonar frame
- Orientation θ^S of the quay wall relative to the sonar frame

With these outputs, the FLS-based quay wall tracker turns the sonar into a sensor delivering distance and orientation measurements referenced to a feature of the environment i.e. the quay wall.

3-2-3 Non-functional requirements

The other set of requirements define the desired performance of the system. The requirements have quantifiable metrics in order to be able to compare between solutions and to evaluate the results.

Accuracy Ideally, the quay wall tracker should be able to track the quay wall for all the FLS images containing a quay wall line feature. Given that the tracking functionality rests on a previous line measurement (ρ^S, θ^S) , it is easy to infer that this can only be achieved with sufficient accuracy, where a mediocre prior estimation could cause the system to lose track of the quay wall. Moreover, highly accurate estimates naturally leads to more accurate localization and therefore a requirement on that respect should be set. One of the most popular methods for measuring the accuracy of estimated variables is the Root-Mean-Square Error (RMSE) [42]. It is closely related to the standard deviation and for any variable $\hat{x}$ it is computed as follows:

$$RMSE(\hat{x}) = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{x} - x)^2} \quad (3-2)$$

Where $\hat{x}$ is the noisy estimate of the value of variable x . For a zero-mean Gaussian distributed error, the probability that the error remains inside the 95% interval for a given RMSE value is $P(\hat{x} - x < 2 \cdot RMSE(x)) \approx P(\hat{x} - x < 2\sigma) = 0.95$. In practice, consecutive cleaning lanes are shifted approximately 1 [m] apart depending of the quality of conditions, which corresponds to approximately 0.50 [m] overlap (the robot’s cleaning width is approximately 1.50 [m]). Here it is assumed that the 95% interval should not exceed the overlap. This translates to $RMSE(\rho) < 0.125$ [m]. Furthermore, the constraint for the orientation errors is set at $RMSE(\theta) < 0.500$ [deg].

Robustness Although the RMSE is a good indicator of an estimator’s accuracy, it does not reflect the robustness of an estimator. Indeed, the RMSE value could for example reflect a strong bias or a high variance with large outliers. Since the system uses prior estimations, misleading outliers present a risk for the tracking system to fail; therefore a measure for the outliers in the quay wall line estimations is of interest. The chosen metric is the *ACC*

score, measuring the proportion of estimates of some variable $\hat{x}$ which fall inside some interval around the ground truth value x :

$$ACC(x) = 100\% \times \frac{1}{N} \sum_{i=1}^N [[|\hat{x} - x| < x_{lim}]] \quad (3-3)$$

where N is the number of measurements and $[[\cdot]]$ is the Iverson bracket¹. In order to penalise outliers in either of the line parameters (ρ^S, θ^S) , the average between $ACC(\rho)$ and $ACC(\theta)$ is used, where the intervals are set at $\rho_{lim} = 0.25$ [m] and $\theta_{lim} < 1.0$ [deg], corresponding to the intervals defined in the previous paragraph. Therefore, the minimum accuracy should be set to $ACC(x) < 95\%$.

Computation time In general, it is preferable to have sensors with high sample rates, as it provides a higher time-resolution for measurements which enables to better capture the motion of the robot. Therefore, in order to obtain the maximal amount of measurements from the FLS, the quay wall detector should be able to complete the process of extracting the distance and orientation measurements (ρ^S, θ^S) for each incoming frame. According to the specifications, the frame rate of the FLS does not exceed 25 [Hz], therefore the computation time should not exceed $t_c < 0.040$ [s].

3-3 Drift-free localization

In the previous section, the requirements for the front-end of the system were described. Thanks to line tracking from FLS imagery, absolute measurements are added to the sensory capabilities of the robot. This added input is in turn processed by the back-end of the system to enable localization with reduced drift. Before defining the requirements for the back-end of the localization scheme, the quay wall as a reference should be discussed.

3-3-1 The quay wall: a reference for localization

In order to properly use the quay wall as a reference for localization, considerations on the relationship between relevant reference frames require to be specified.

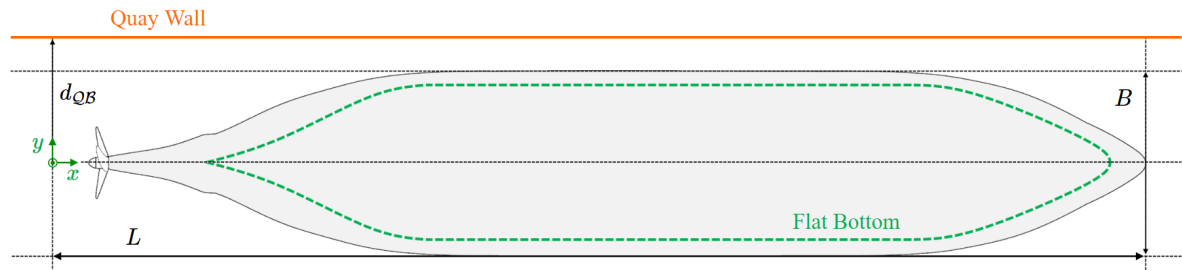


Figure 3-4: In general, the majority of a ship's bottom surface is known to be flat (area enclosed by green dotted line). Furthermore, its length L and breadth B are specified in the General Arrangements (GAs).

¹The Iverson bracket returns one if the condition inside the brackets is true, otherwise zero.

Assumptions In the Flat Bottom's 2D space, the quay wall is a line at a fixed distance from the ship, and is parallel to the ship along its length (x -axis). That is, it can simply be described by one single parameter with respect to the ship frame: its lateral offset distance $\pm d_{QB}$, where the quay wall is at the port-side (left) for positive values and at the starboard-side (right) for negative values. However, although the sign of this distance is known beforehand, its magnitude is a-priori unknown. Since it is parallel to the longitudinal x -axis, the quay wall can only offer a lateral and heading reference i.e. in the y - and ψ -coordinate of the robot. An illustration of the quay wall configuration along the ship is given in Figure 3-4.

Defining the offset Naturally, in order to localize the robot w.r.t. the ship frame while using the quay wall as a reference, the distance d_{QB} has to be determined. Based on the knowledge of the geometry of the environment, two approaches can be considered. The first method is based on a prior estimate of the robot pose $(x, y, \psi)_{prior}$ with respect to the ship frame (provided by the operator as explained in Chapter 1) and a measurement of the quay wall distance ρ^S from the sonar frame:

$$d_{QB} = \begin{cases} y_{prior} + \rho^S, & \text{if quay left} \\ y_{prior} - \rho^S, & \text{if quay right} \end{cases} \quad (3-4)$$

Another method is to determine the offset between the edge of the ship at the bilge and the quay wall. This could be obtained by first extracting the quay wall ρ^S in parallel with the bilge detector from [18] to measure the distance to the bilge ρ_b^S . Since the breadth B of the ship is known, the offset can be deduced:

$$d_{QB} = \begin{cases} \frac{B}{2} + \rho^S - \rho_b^S, & \text{if quay left} \\ -\left(\frac{B}{2} + \rho^S - \rho_b^S\right), & \text{if quay right} \end{cases} \quad (3-5)$$

Ideally, the offset distance d_{QB} should be based on actual measurements instead of manual approximations by the operator. However, during the writing of this thesis, Steensma's bilge detector [18] was not available for implementation. For this reason, the first method relying on operator approximations as described in Equation (3-4) is to be used instead.

3-3-2 Functional requirements

Inputs For the localization of the robot, the following inputs can be used:

- Prior pose estimate defined by the operator $(x, y, \psi)_{prior}$
- A range and orientation estimate (ρ^S, θ^S) from the quay wall tracking system
- Measurements from the auxiliary sensors

Outputs The localization system should be able to estimate pose of the robot frame $\mathcal{R}$ with respect to the ship frame $\mathcal{B}$, where the longitudinal x -coordinate is allowed to be unreferenced since the quay wall offers no longitudinal reference:

- Longitudinal x -coordinate (unreferenced)

- Lateral y -coordinate (quay wall as absolute reference)
- Heading ψ (quay wall as absolute reference)

In other words, localization of the robot is done with the help of the quay wall tracker, sensory and operator inputs.

3-3-3 Non-functional requirements

Accuracy and robustness As mentioned in Section 1-1-2, sensor measurements often suffer from noise disturbances, an issue which can be alleviated by fusing complementary sensors together to improve estimates. Since the back-end of the proposed system can use auxiliary sensors next to the FLS, it will include the required filter for sensor fusion. It follows that the filtered back-end will be subjected to the same accuracy requirements as defined in Section 3-2-3), with the additional condition that its performance must be improved compared to the unfiltered front-end. The same is done for the robustness scores.

Computation time In order to still satisfy the requirement to extract measurements for each incoming sonar image, the processing time of the back-end in addition to the quay wall tracker's processing time should remain under the FLS's update rate $t_c < 0.040$ [s].

Drift As mentioned in the introduction, the goal of this thesis is to achieve drift-free localization. Since the quay wall as a line feature only offers an absolute reference for lateral position y and heading ψ , the localization system should be able to reduce the drift along these coordinates, while the drift in the longitudinal position will be omitted since this falls outside the scope of this thesis. The chosen measure for drift will be inspired from [43], taking the absolute error in the final position of some coordinate x , however omitting the traveled distance:

$$EBU(x) = |\hat{x}_f - x_f| \quad (3-6)$$

where x_f and $\hat{x}_f$ are the true and estimated final positions. The reason to omit the traveled distance is that it is desired to constrain the drift below a certain limit (or keep the same accuracy) regardless of distance (or time). On that basis, the error build-up constraints should match the intervals the accuracy requirements are based on, this yields $EBU(y) < 0.250$ [m] and $EBU(\psi) < 1.000$ [deg].

Chapter 4

Quay wall tracking

In the previous chapter, the operating conditions and the requirements for the quay wall tracker and the drift-free localization system were defined. These will serve as guidelines for the the objective of this chapter to *provide a proof of concept for an Forward Looking Sonar (FLS)-based algorithm measuring the quay wall line parameters*. This will be achieved by identifying the required image processing subfunctions to detect the quay wall as a line feature. Next, the solutions will be implemented on recorded sonar imagery and the results will be compared, in order to choose the optimal solution.

4-1 Image processing

As explained in Section 2-1-1, an FLS senses echo intensities of sound waves along with their spatial origin. Naturally, to translate this information into the measurement of the quay wall's line parameters, processing of the image data is required. To achieve this, the rest of this section will first describe how the FLS can be treated as a rangefinder sensor useful for wall detection and then the required subfunctions for tracking will be justified.

4-1-1 The FLS: a rangefinder

In the literature surrounding localization of mobile robots in structured indoor environments, rangefinders are the standard sensors for wall detection, as shown for example in [44, 45, 46, 47]. Unlike imaging sonars, rangefinders directly return the range of the closest target or obstacle for specific bearing angles, yielding dense point clouds from which line parameters can be extracted. In a similar manner, wall detection with underwater sonars equates to finding the range corresponding to the reflection originating from the wall, for each beam. Then, the normal distance and orientation to the wall is determined by extracting the line that best fits the point cloud that arises from the spatial detection points [48, 34, 49, 50, 51, 52].

4-1-2 Subfunctions

Although FLS images are in essence a depiction of echo intensities measurements, the pixels forming them inherently carry spatial information, i.e. their location in the image correspond to a range r and bearing t as explained in Section 2-1-1. With adequate processing, the pixels corresponding to a specific target can be identified to obtain a set of range r_i and bearing t_i measurements from FLS images, or scan points. In other works considering dam walls, defining the range can be as simple as selecting the highest intensities [48, 51], since large wall structures generally return the strongest reflections, especially when no other target is present in the environment. However, as explained in Chapter 2, not only the quay wall but also other targets return strong reflections e.g. fouling, making the distinction between quay wall and background from reflection intensities much less trivial. Additionally, the quality of FLS images is degraded by, among other things, speckle noise and non-uniform illumination. Therefore, the FLS require careful processing in order to correctly separate pixels corresponding to the quay wall from pixels corresponding to the background or irrelevant artefacts. Only then can range measurements be performed from which line parameters can be obtained. From this explanation, three necessary subfunctions for quay wall detection arise:

- **Segmentation** i.e. identifying the pixels in the image which correspond to the quay wall, as such separating them from artefacts and the background.
- **Line extraction** i.e. finding the line parameters that best fits the range measurements to describe the quay wall relative position.

A flowchart of the quay wall detection system and its subfunctions (segmentation, range extraction and line fitting) is depicted in Figure A-1, where the gray blocks represent the already discussed higher level subsystems, and the coloured blocks are the underlying functions, which will be discussed in the foregoing sections.

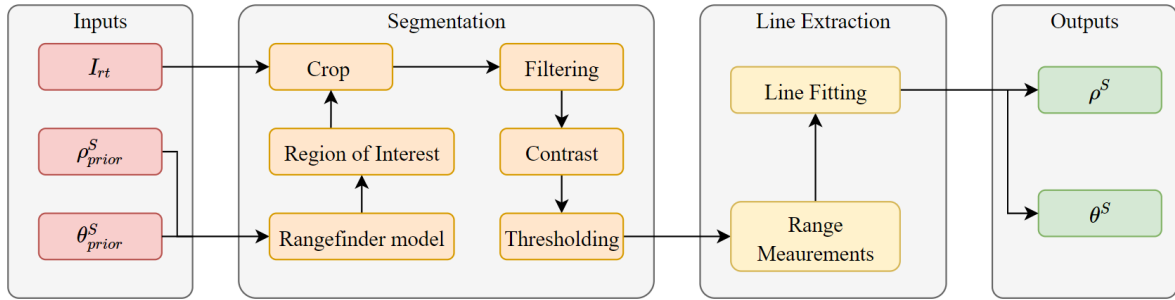


Figure 4-1: Flow chart of the quay wall detection algorithm, translated polar image I_{polar} to a line measurement (ρ_S, θ_S) relative to the sonar frame. The algorithm consists of two stages: image segmentation and line fitting.

4-2 Segmentation

This section describes the first step of the FLS quay wall tracking system: segmentation. The purpose of segmentation is to separate quay wall pixels from background pixels, from which

range measurements will be possible afterwards. First, the inputs of the system defined in Section 3-2-2 will be described in more details.

4-2-1 Inputs

The requirements allow the quay wall tracker to use polar or Cartesian images for line detection. As the polar image represents the native data from the FLS and Cartesian images suffer from nonuniform resolution, the polar representation is preferred. Furthermore, since object tracking relies on the prior information from previous detections, prior line parameters $(\rho_{prior}^S, \theta_{prior}^S)$ are also considered as inputs.

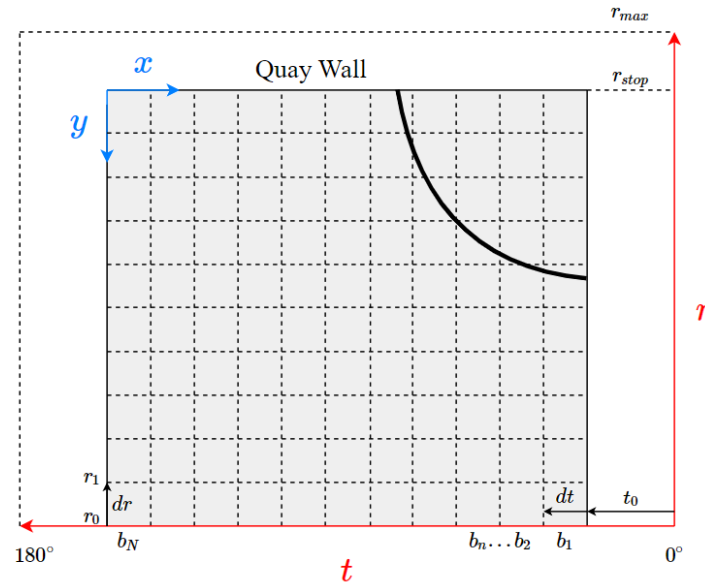


Figure 4-2: Illustration of polar image parameters, where each grid cell corresponds to a pixel. For target detection, any pixel (x, y) in the image frame (blue frame) can be related to polar coordinates (r, t) . The number of rows M varies according to the maximal viewing range r_{stop} , which is chosen by operators in practice.

Image parameters Figure 4-2 offers an illustration of the incoming input polar image along with its parameters. It consists of $N = 768$ beams (columns) at bearing angles $t_i = t_1 \dots t_N$ and M range increments (rows)¹ $r_j = r_1 \dots r_M$. The spatial correspondence of a pixel at the n -th column and m -th row depend on range and bearing resolutions dr and dt , respectively:

$$t_n = t_0 + n \cdot dt - \frac{dt}{2} \quad (4-1)$$

$$r_m = r_0 + m \cdot dr - \frac{dr}{2} \quad (4-2)$$

¹In contrast with the number of beams N , the number of range increments M is variable (depending on the zoom i.e. chosen stop range r_{stop}).

where t_0 is the angle between the x -axis of the sonar frame and the first beam² and r_0 is the sonar's minimal range³ i.e. the bottom row of the image.

Parameter space Next to the incoming polar image, the requirements of the quay wall tracker system allow the use of previously detected line $(\rho_{prior}^S, \theta_{prior}^S)$. Next to the prior line, assumptions on the robot's motion between frames enable to define a parameter space describing the set of possible lines. The benefit of a parameter space is two-fold: firstly, the search space can be reduced to a local region of the image, improving the computational efficiency of the system and secondly, robustness is improved as outliers can be easily rejected. Indeed, given the robot's low forward speed ($v \approx 0.15$ [m/s]) and the high sonar frame rate (up to 25 [Hz]), the change of the robot's normal distance and the orientation should be minimal between subsequent sonar frames. Therefore, it is fair to assume that the new line will be close to the previously found line $(\rho_{prior}^S, \theta_{prior}^S)$:

$$\rho_{prior}^S - d\rho_{prior}^S \leq \rho^S \leq \rho_{prior}^S + d\rho_{prior}^S \quad (4-3)$$

$$\theta_{prior}^S - d\theta_{prior}^S \leq \theta^S \leq \theta_{prior}^S + d\theta_{prior}^S \quad (4-4)$$

Where $d\rho_{prior}^S$ and $d\theta_{prior}^S$ define the magnitude of the line parameter increments between subsequent frames. Based on the kinematic model from Section ??, the maximal change of the line parameters can be assumed to be constrained:

$$d\rho_{prior}^S < \left| v \cdot \sin \theta_{prior}^S \cdot \Delta t \right| \quad (4-5)$$

$$d\theta_{prior}^S < \left| \frac{v \cdot \sin \alpha}{l} \cdot \Delta t \right| \quad (4-6)$$

where Δt is the time step between subsequent sonar images. In practice time steps are no bigger than $\Delta t = 1$ [s] according to Section 2-4; filling in the values in the above equations yields $0 \leq d\rho_{prior}^S \leq 0.15$ [m] depending on θ_{prior}^S and $d\theta_{prior}^S \leq 5$ [deg].

4-2-2 Region of Interest

As mentioned previously, the parameter space can be used to reduce the search to a local Region of Interest (ROI) instead of the entire image. If used adequately, an ROI can be defined to encompass quay wall pixels while excluding irrelevant objects e.g. fouling, reducing the chance of faulty measurements. This will play a key role in correctly selecting the pixels corresponding to the quay wall (more details in Section 4-2-5).

² $t_0 = (\pi - N \cdot dt)/2$

³ r_0 is given by the sonar's internal data output.

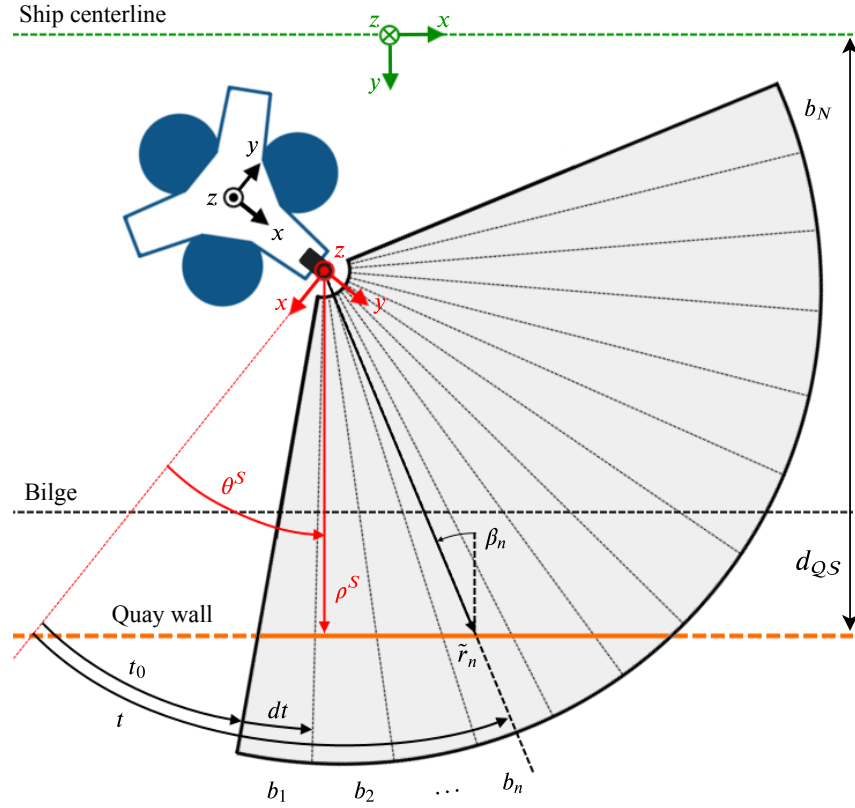


Figure 4-3: Schematic of the geometrical parameters considered for quay wall detection. For a prior line (ρ^S, θ^S) , a prediction of the range $\tilde{r}_n$ than would be ideally measured at beam angle t_n can be obtained, along with the incidence angle β_n between the beam b_n and the quay wall.

Beam selection From Figure 4-2 and Figure 4-3, it is clear that the quay wall is not observable for all beams b_i . Consequently, the space of polar image coordinates (r, t) where a line is to be detected can be reduced to the beam angles t_i intersecting the quay wall within the sonar's Field of View (FOV). These can be found by first predicting the range measurements $\tilde{r}_i$ of each beam angle t_i for a given prior line $(\rho_{prior}^S, \theta_{prior}^S)$ with a simple rangefinder model. From the geometry of Figure 4-3, it is easy to infer that the rangefinder model should be related to incidence angles β_i :

$$\beta_i = t_i - \theta_{prior}^S \quad (4-7)$$

$$\tilde{r}_i = \frac{\rho_{prior}^S}{\cos(\beta_i)} \quad (4-8)$$

Then, the set T_1 of beams crossing the quay wall line within the sonar's FOV is found by selecting the beam angles t_i for which predicted range measurements $\tilde{r}_i$ fall within the image's lower and upper range boundaries r_{min} and r_{stop} :

$$T_1 = \{t_i \mid r_{min} \leq \tilde{r}_i \leq r_{stop}\} \quad (4-9)$$

Furthermore, as mentioned in Section 2-1-1, the intensity of reflected echoes becomes weaker for both increasing ranges $\tilde{r}_i$ and increasing incidence angles β_i , resulting in the quay wall

to become less bright and thus less distinguishable from the background. Although the loss due to increasing range is compensated for by the sonar's built-in Time Varying Gain (TVG), the transmission loss from increasing incidence angles is not considered. Moreover, it was also mentioned that the quay wall line appears to dilate for increasing ranges (or conversely, for increasing incidence angles β_i as can be deduced from Equation 4-8), creating more uncertainty of quay wall parameters in the sonar image. These issues strongly suggest that, next to the range constraint described from Equation 4-9, also a constraint for incidence angles β_i should be determined for optimal line tracking performance. That is, the selection of beams should be further reduced by only choosing beams below some incidence angle β_{max} :

$$T_2 = \{t_i \mid \text{abs}(\beta_i) \leq \beta_{max}\} \quad (4-10)$$

Moreover, as sonar images tend to show noisy illuminations along the beams at the image extremities (Section 2-1-1), a constant k_m is defined to reject the k_m left-most and right-most beams to tackle this problem:

$$T_3 = \{t_i \mid k_m < i < N - k_m\} \quad (4-11)$$

The final set of beam angles T to define the ROI is the set satisfying the range and incidence angle conditions and the outer beam rejection:

$$T = T_1 \cap T_2 \cap T_3 \quad (4-12)$$

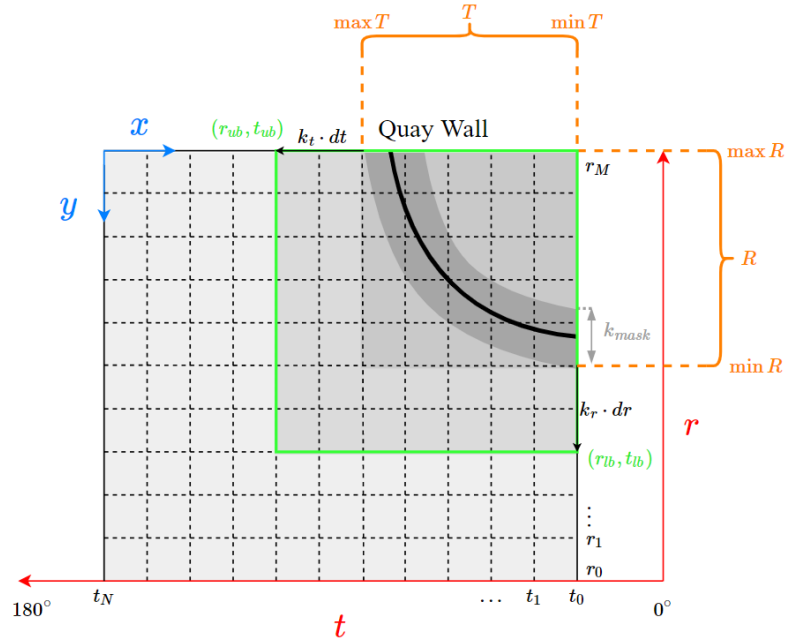


Figure 4-4: Illustration of the Region of Interest (green square), described by its top left corner (r_{lb}, t_{lb}) and bottom right corner (r_{ub}, t_{ub}) . It is defined by first defining the minima and maxima of range and angle measurements for a given prior line (black line). Then, an image mask along the line (dark gray) is used to ensure only pixels in the neighbourhood of the quay wall are used for subsequent processing.

Range bounds Based on beam angles in T , the respective set R containing the predicted range measurements $\tilde{r}_i$ can be defined:

$$R = \{\tilde{r}_i \mid t_i \in T\} \quad (4-13)$$

From there the corner points describing the lower and upper boundaries of the ROI, respectively (r_{lb}, t_{lb}) and (r_{ub}, t_{ub}) , are defined by adding a margin to the minimum and maximum values of R and T (see Figure 4-4):

$$r_{lb} = \begin{cases} \min(R) - k_r \cdot dr, & \text{if } \min(R) - k_r \cdot dr > r_0 \\ r_0, & \text{otherwise} \end{cases} \quad (4-14)$$

$$t_{lb} = \begin{cases} \min(T) - k_t \cdot dt, & \text{if } \min(T) - k_t \cdot dt > t_0 + k_m \cdot dt \\ t_0 + k_m \cdot dt, & \text{otherwise} \end{cases} \quad (4-15)$$

$$r_{ub} = \begin{cases} \max(R) + k_r \cdot dr, & \text{if } \max(R) + k_r \cdot dr < r_M \\ r_M, & \text{otherwise} \end{cases} \quad (4-16)$$

$$t_{ub} = \begin{cases} \max(T) + k_t \cdot dt, & \text{if } \max(T) + k_t \cdot dt < t_N - k_m \cdot dt \\ t_N - k_m \cdot dt, & \text{otherwise} \end{cases} \quad (4-17)$$

Where the margins k_r and k_t are there to account for the shift in normal distance $d\rho_{prior}^S$ and orientation $d\theta_{prior}^S$ described in Section 4-2-1, respectively.

Line mask It is clear from Figure 4-4 that the ROI's square shape does not match the quay wall line shape, extending relatively far from the line at corners. In order to precisely enclose the ROI around the quay wall line, an image mask in the form of a thick line is used. The thickness in pixels of the line mask is defined by k_{mask} (see Figure 4-4).

4-2-3 Noise filtering

As explained in Section 2-1-1, the quality of acoustic imagery is typically degraded by speckle noise. Due to its multiplicative property, it is particularly challenging to separate the noise from the true signal. To tackle this problem, numerous filters have been developed, of which the performance has been compared in [39, 53] for different applications e.g. ultrasound imagery. Across their work, two main classes were identified: scalar and adaptive filters.

Scalar filters Scalar filters are non-adaptive filters which individually scale pixel values based on local image statistics inside a neighbourhood window, or *kernel*, defined by its size $F = n_r \times n_t$ (width and height respectively) in pixels. According to [54], the most commonly used filters for sonar imagery are the median and Gaussian filters. As the name suggests, the median filter outputs a new pixel value by selecting the median intensity value inside window of surrounding pixels. This operation effectively removes noise while preserving edges [53]. In contrast, the Gaussian filter convolves the image with a kernel that approximates a Gaussian profile, creating a smoothing effect. An example for both filters is given in Figure 4-5 for comparison. Non-adaptive filters have advantage of straightforward implementation and low

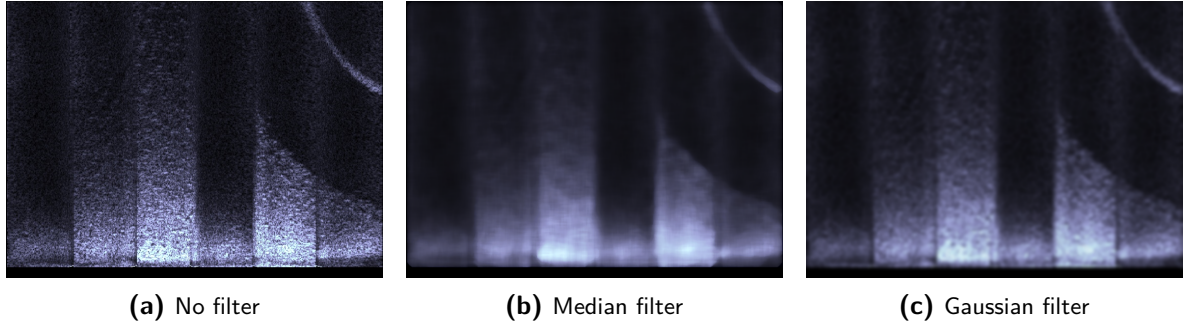


Figure 4-5: Examples of non-filtered and filtered images. In order to clearly see the differences between methods, the kernel size of the filters was set to a large value of $F = 21 \times 21$ pixels for visual purposes.

computation cost compared to adaptive filters [53]. For FLS image denoising Weng et al. [55] compared the mean, Gaussian and median filter and found the median filter with kernel size $F = 3 \times 3$ pixels to show superior Peak Signal to Noise Ratio (PSNR). Similarly, the use of the median filter is often advocated for its minimal blurring and better preservation of edges compared to smoothing filters such as the Gaussian filter [40, 56].

Adaptive filters In contrast, adaptive filters adapt the kernel depending on local, statistical or spatial properties of the input image. In terms of noise reduction in ultrasound imagery, adaptive filters (such as the Lee [57], Frost [58], Kuan [59] or Ho filter [60] to name a few) have proven to offer superior performance [53]. Among those, the Frost filter is identified as the most suited filter for object detection with FLS devices [39]. However, as pointed out in [18], the Frost filter is not ideal for real-time application due to its computational cost. Since minimal computation time is desired (Section 3-2-3), the Frost filter and the rest of the adaptive filters will not be further considered in this work.

Chosen method Furthermore, as mentioned earlier, the incoming sonar images suffer from various disturbances and limitations (Section 2-1-1). As the ROI was designed to tackle the majority of these problems, it is believed that the benefit from complex adaptive filters will be minimal. Furthermore, the median and the Gaussian filter have already proven to offer a great trade-off between computational efficiency and noise reduction performance, therefore these two are selected for the noise filtering subfunction.

4-2-4 Contrast

An important aspect of object detection is ensuring sufficient contrast, i.e. a noticeable difference in pixel intensity in order to make objects more salient compared to the background. In other words, the higher the intensity discrepancy between targets, the more contrast is observed and the easier these can be distinguished.

Histogram-based methods Based on contrast's relation with the range of intensity values present in the image, a large majority of contrast enhancement methods as surveyed in [61]

are based on improving intensity histograms of images. The most straightforward approach is called contrast stretching, which consists of stretching an initially narrow intensity histogram to a wider range of represented values (used in [50]). More elaborate methods seek to evenly spread out the frequency of pixel intensities globally across the image, such as Histogram Equalization (HE), or locally within smaller tiles of the image, such as Contrast Limited Adaptive Histogram Equalization (CLAHE). An illustration of both techniques is shown in Figure 4-6 for visualization. For the FLS-based bilge detector, the CLAHE method was chosen above the HE algorithm especially for its ability to compensate for the nonuniform illumination of sonar images and minimal noise amplification [18] (as shown in Figure 4-7).

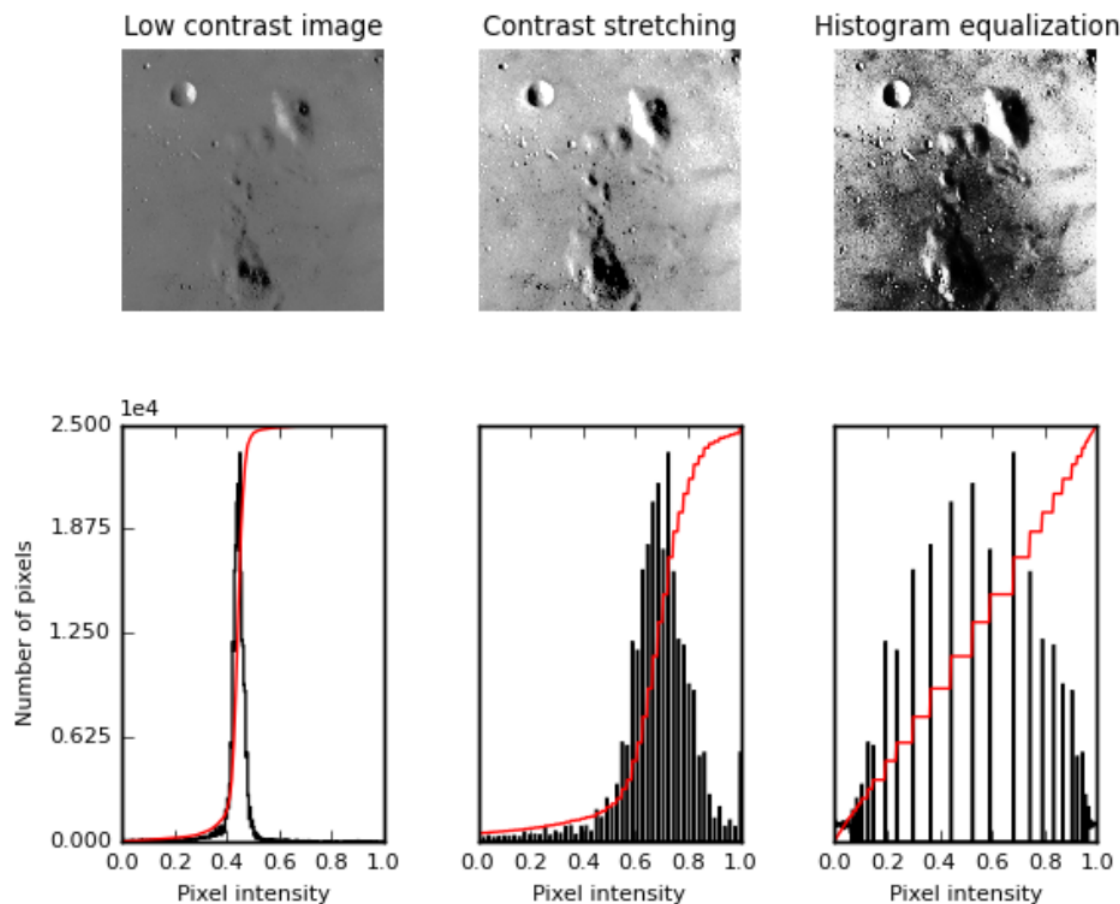


Figure 4-6: Comparison of contrast enhancement techniques for low contrast images, and their corresponding intensity histograms. The red line represents the cumulative histogram, where a steep line suggests the dominance of a particular intensity level, causing low contrast.

Alternative method However, the bilge is characterized by an edge between relatively high and low intensity pixels whereas objects like the quay wall are identified by typically high intensity reflections. Alternatively, Weng et al. proposed a pixel-based contrast enhancing method which was validated on an FLS device comparable to the one considered in this thesis (BlueView P450E). This filter accentuates the contrast of reflections by using a basic form of

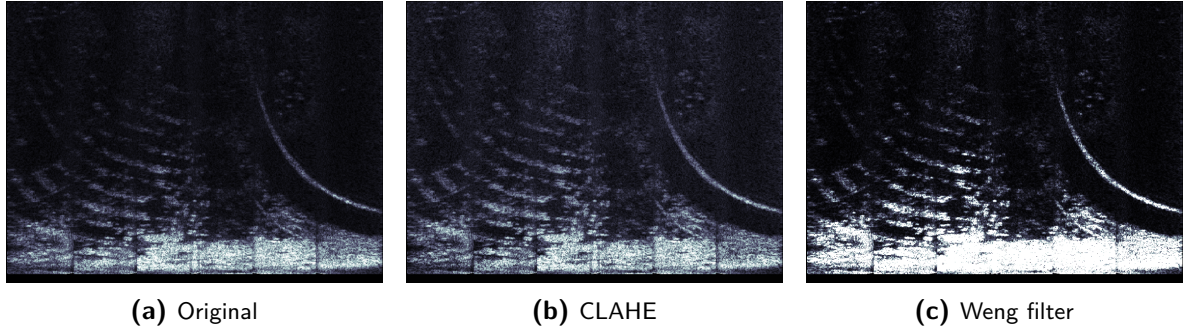


Figure 4-7: Contrast enhanced images. The CLAHE image was enhanced with parameters $C_{clip} = 0.75$ and tile size $H = 8 \times 1$ pixels and the Weng filtered image was obtained with standard values $c = 0.1$ and $r = 1.7$.

the power-law transformation which amplifies high intensities and suppresses low ones:

$$g(x, y) = c \cdot f(x, y)^r \quad (4-18)$$

where $f(x, y)$ and $g(x, y)$ are the input and output grayscale pixel intensities at image location (x, y) respectively, and the constants set at $c = 0.1$ and $r = 1.7$ showed the best detection results [55]. In contrast with HE methods, output pixels from Weng's method saturate for input intensities above a certain level, with an over-representation of white pixels as a result (as can be witnessed in Figure 4-7).

Chosen method Although B. Steensma showed in his work that the CLAHE method successfully compensates for diminishing illumination for higher ranges, for this work's application it is arguably not ideal. Indeed, as was mentioned in Section 2-1-1 and further explained in Section 4-2-2, the FLS's intensity measurements are clearly less reliable at higher ranges. Conversely, closer reflections are more salient and thus more reliable. Therefore, it makes more sense to use a method that effectively accentuates already strong reflections and suppresses weak reflections. For this reason, Weng's method [55] is considered more suitable and it therefore chosen for implementation.

4-2-5 Segmentation

All of the previous steps (defining the ROI, filtering and contrast enhancement) are designed to facilitate the last step of the image segmentation phase, where the filtered image will be partitioned into two classes of pixels: background and quay wall pixels.

Thresholding According to [18], the most widely applied method for segmentation is thresholding. Based on the assumption that objects have a distinct intensity compared to background pixels, a carefully selected intensity threshold T_I is used for classification. In other words, a pixel $I(x, y)$ is considered to represent an object if its intensity exceeds T_I :

$$I(x, y) = \begin{cases} 1 & \text{if } I(x, y) \geq T_I \\ 0 & \text{otherwise} \end{cases} \quad (4-19)$$

Threshold selection Different methods exist for defining the value of the threshold. The most straightforward approach is to use a fixed global intensity threshold that can be determined empirically [52]. In the context of underwater wall detection [48, 49, 50, 51], the most common approach is to apply a threshold manually or empirically as the highest intensity pixels along a beam of acoustic images generally correspond to the presence of large planar structures, although errors may occur [50] due to spurious measurements. However, these methods are justified by the fact that the highest intensities consistently correspond to wall reflections. For less trivial problems, the threshold can be found through statistical methods such as k-means clustering [62] or Otsu's method [63], a widely applied method which proves to be effective to separate distinctive classes of pixels. For highly variable and unpredictable environments, adaptive thresholds [40], such as hysteresis thresholds [39] or two-stage thresholding [21] can offer improved robustness compared to static techniques. By the same token, locally defined thresholds [64] give more robust results for spatially varying environments.

Chosen method As explained in Section 2-1-1, in the given problem setting, determining the intensity threshold to isolate the quay wall is not straightforward:

- The highest echo intensities often come from close by targets such as fouling of weld lines instead of the much further situated quay wall. This yields multiple reflection peaks along each beam.
- The average intensity of quay wall pixels vary temporally and spatially. That is, it can vary over time and, as explained in Section 2-1-1, increasing range and incidence angles diminish reflection intensity of targets.

However, as explained in Section 2-2, the quay wall is typically surrounded by large regions of dark pixels. The ROI around the line as described in section 4-2-2 greatly simplifies this problem however, by leaving only two classes of pixels: the background and the quay wall. As early trials showed that Otsu's method functioned well compared to manual and empirical thresholds, it was considered that little improvement was to be gained from more elaborate methods.

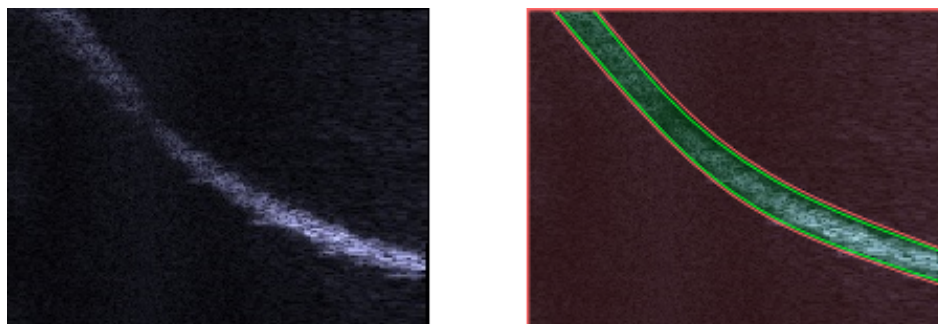


Figure 4-8: Representation of the ideal result from the segmentation on the ROI of the image from Figure 2-4. **Left:** ROI of the original image. **Right:** The background pixels are highlighted in red, the quay wall pixels in green.

4-3 Line fitting

From the image segmentation phase separating the quay wall pixels from the background, only quay wall pixels are preserved. From this image, the range from the sonar to the quay wall can be extracted for each beam, as such using the FLS as a rangefinder (Section 4-3-1). These range-bearing measurements can later be used for fitting the line describing the quay wall (Section 4-3-2).

4-3-1 Range extraction

Selection process Range measurements can be extracted simply by finding the maximum reflection intensities for each beam for raw images [48, 49, 51], or segmented images [50]. Alternatively, Karras et al. selects the first non-zero bin close to the sonar for the line fitting process, based on the assumption that the wall is the closest object of reflection [52]. In a similar manner, it is fair to assume the quay wall to be the furthest detection reflection, given nothing can be detected beyond the wall. For simplicity, it is assumed that the line representing the true quay wall corresponds to the line through the middle (splitting the thick line in two thinner lines).

Binary vs. grayscale A choice for the explained segmentation procedure, as it can yield two different results: all pixels above the defined threshold are set to white, or all pixels above the defined threshold keep their original grayscale intensity while others are rejected (set to black). For each beam of the sonar image, this yields either a block or a peak of nonzero pixel intensities flocking at specific ranges. Through trial and error, combinations of the above methods with binary or grayscale were tested for range extraction. Surprisingly, it was found that results from these methods were unreliable, due to spurious pixels causing range measurements to deviate from the blocks or peaks. To solve this issue, the intensities for each beam were smoothed with a Gaussian filter, where the maximum intensity is selected as the range corresponding to the quay wall for that beam. As a starting point, the smoothing standard deviation at $\sigma_r = 10.0$ was found to visibly better results than other methods.

Detection points When the range readings are extracted from the segmented sonar image, these are coupled to their respective beam angle to yield polar detection points $\{(r_i, t_i)\}_{i=1}^N$. Then, these can be simply translated to the Cartesian space as follows:

$$x_i = r_i \cos t_i \quad (4-20)$$

$$y_i = r_i \sin t_i \quad (4-21)$$

Transforming each measurement yields the set of Cartesian detection points $\{(x_i, y_i)\}_{i=1}^N$ that will be used for line fitting.

4-3-2 Line fitting algorithms

The final step in the quay wall detection scheme uses the detection points extracted from segmented beams in order to find the best fit for the line feature describing the quay wall.

This section will discuss existing algorithms and select the most suitable for Fleet Cleaner's application. From there, the quay wall detection system will be complete and will be tested by implementing algorithms on real data.

Common methods In the robotics field, there is much interest in the detection of line segments since environments can often be described by a collection of linear segments (such as walls, pathways, doors etc.) where line endpoints are also considered next to their normal distance ρ and orientation θ . For laser rangefinders, numerous methods have been developed such as Split and Merge or Expectation Maximization [65]. Luckily, as mentioned in Chapter 3, the quay can be described by an infinite line feature without endpoints, meaning only the problem of line fitting is relevant. To this end, the most widely used method is Total Least-Squares (TLS) according to [65], which is in essence a nonlinear cost minimization scheme to get the line where the sum of squared errors is at a minimum [44]. In the context of line detection in structured underwater environments with range measurements from sonar images, the Hough Transform [66, 67] proves to be an adequate choice e.g. in watertanks [68], dam walls [49, 51] and marinas [34]. However, in a watertank wall-following experiment with the help of an FLS, the RANdom SAMple Consensus (RANSAC) approach shows better performance when numerous outliers corrupt the sonar data [52]. It follows that three methods for line fitting stand out: the Hough Transform, RANSAC and TLS. These will be discussed in more details in the following sections.

4-3-3 Hough Transform

The Hough Transform (HT) is one of the most widely applied algorithms for line detection [66, 67] for optical images, and also very much so in the context of underwater wall detection with acoustic images [34, 49, 68, 48].

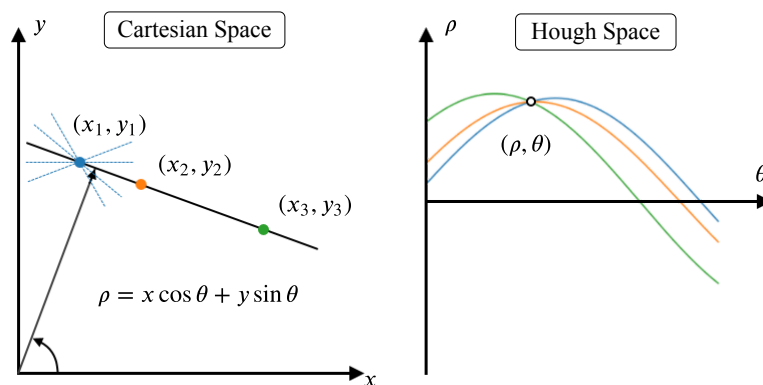


Figure 4-9: Illustration of the working principle of the Hough Transform. For each point (x_i, y_i) in the Cartesian space, the set of lines (ρ_i, θ_i) through that point are determined according to $\rho = x \cos \theta + y \sin \theta$, yielding a sinusoidal curve in the Hough space. Through this transformation, the line through all points in the Cartesian space becomes the intersection of the curves in the Hough space.

Working principle The Hough Transform is a method that detects lines in binary images and is illustrated in Figure 4-9. It is a voting scheme where, for each selected pixel p_i at

image location (x_i, y_i) , the set of all lines through p_i are determined by computing the normal distance ρ of the lines to the origin for all quantized angles θ_j in the Hough space (ρ, θ) :

$$\rho_{ij} = x_i \cos \theta_j + y_i \sin \theta_j \quad (4-22)$$

For each found line (ρ_{ij}, θ_{ij}) , the corresponding bin in the Hough space *accumulator* receives a vote by incrementing its value. This produces a sinusoidal curve in the Hough space for each pixel, where an intersection of curves corresponds to a line through these pixels.

Properties The HT is reported to be resistant to noise compared to others methods [69] and also doesn't require full lines for detection, .e.g. dotted lines are also effectively detected [70]. Nonetheless, the HT's voting scheme can be seen as a brute-force method with high computational cost depending on the size of the parameter space. By extension, choosing the adequate angle resolution $d\theta$ and $d\rho$ of the Hough space is problematic, where a trade-off between accuracy and computational speed has to be considered. However, this issue can be alleviated with a faster formulation of the Hough Transform, called the Fast Hough Transform (FHT) [71].

Considerations Conventionally, the HT is executed across the pixels selected after segmentation. However, given the sonar images are used as rangefinders, the HT is only used across the detection points instead of all the pixels.

4-3-4 RANdom SAMple Consensus

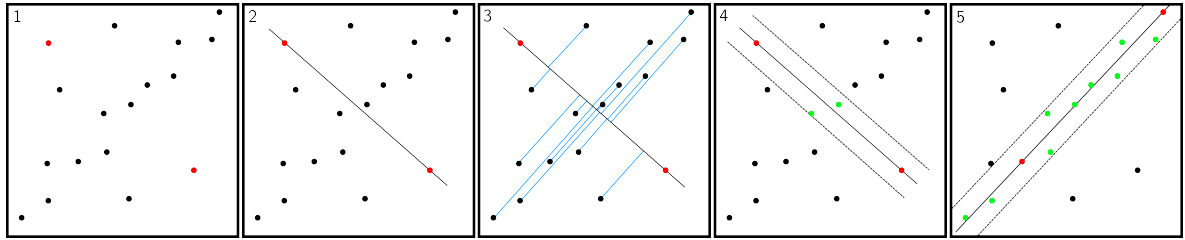


Figure 4-10: Illustration of RANSAC algorithm. (1) Sample selection. (2) Line model proposal. (3) Line model fitting. (4) Consensus selection. (5) Best fit selection.

Working principle The RANSAC algorithm [72] is a widely implemented method for line detection. It is particularly known for its ability to accurately fit line models with significantly noisy data (many outliers). It consists of five steps (illustrated in Figure 4-10):

1. **Sample selection:** Select two random points, these are the *hypothetical inliers*.
2. **Model proposal:** These points are used to propose a line model
3. **Model fitting:** The other data points are fitted on this line model, i.e. their normal distance to the hypothesis line computed in order to determine the fit

4. **Consensus selection:** Detection points are labeled as *inliers* when these fall within a distance d_{inlier} from the hypothesis line, representing the *consensus set*. Afterwards, Principal Component Analysis (PCA) is performed on the consensus set in order to improve the hypothesis model.
5. **Best fit selection:** After N_{RANSAC} iterations, the candidate line model with the largest consensus set (exceeding a minimal numbers of inliers) is selected as the best solution.

According to the authors of [72], the threshold for the number of inliers $T_{inliers}$ and error tolerance d_{RANSAC} should be determined experimentally, while a statistical approach can be used to select the number of iterations N_{RANSAC} . By considering the reciprocal of the probability p that the algorithm exclusively selects inliers after N_{RANSAC} iterations defined by

$$1 - p = (1 - \omega^2)^{N_{RANSAC}} \quad (4-23)$$

The number of iterations can be determined by:

$$N_{RANSAC} = \frac{\log(1 - p)}{\log(1 - \omega^2)} \quad (4-24)$$

where ω represents the fraction between the number of inliers and all points in the data set.

Properties RANSAC is known for its robustness to outliers caused by noisy data, showing accurate results even when outliers represent a large proportion of the data. However, RANSAC has no upper bound on the required time to compute the solution (except for exhaustion). That is, there is a trade-off between the number of iterations N_{RANSAC} and the optimality of the solution, where a higher number of iterations increases the chance of a satisfying solution at the price of longer computation times and vice versa. As a consequence, with this method an optimal solution is not guaranteed.

4-3-5 Total Least-Squares

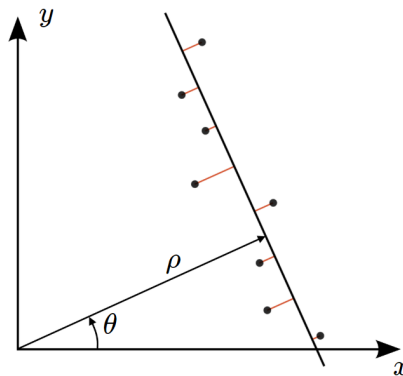


Figure 4-11: Illustration of the TLS method, with candidate line (ρ, θ) , detection points and their respective orthogonal distance to the line (red lines).

Working principle One the most widely implemented line fitting algorithms is the TLS, which is a formulation of the Least-Squares method which seeks to minimize the sum of the squared orthogonal distance of each detection point (x_i, y_i) to the candidate line (ρ, θ) [44], as illustrated in Figure 4-11:

$$E_{fit} = \sum_{i=1}^n (x_i \cos \theta + y_i \sin \theta - \rho)^2 \quad (4-25)$$

Where E_{fit} is the cost function and ρ and θ are the parameters to be optimized. Conveniently, a closed-form solution is available:

$$\theta = \frac{1}{2} \cdot \text{atan} \left(\frac{-2\sigma_{xy}}{\sigma_y^2 - \sigma_x^2} \right) \quad (4-26)$$

$$\rho = \bar{x} \cos \theta + \bar{y} \sin \theta \quad (4-27)$$

where $\bar{x}$ and $\bar{y}$ are the mean values of the x - and y -coordinates of the Cartesian detection points $p_i = (x_i, y_i)$:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i, \quad \bar{y} = \frac{1}{N} \sum_{i=1}^N y_i \quad (4-28)$$

And σ_x^2 , σ_y^2 and σ_{xy} are the covariances:

$$\sigma_x^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2 \quad (4-29)$$

$$\sigma_y^2 = \frac{1}{N} \sum_{i=1}^N (y_i - \bar{y})^2 \quad (4-30)$$

$$\sigma_{xy} = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y}) \quad (4-31)$$

Properties The least-squares method corresponds to the maximum likelihood solution. Thanks to its closed form solution, it is computationally efficient and yields reproducible results in contrast with iterative methods (such as RANSAC, explained in next section). Furthermore, the absence of adaptable parameters means no tuning is required. On the other hand, line fitting with TLS is known to lack robustness against outliers. Indeed, it tends to overcompensate outlier data. Unfortunately, no parameters can be tuned in order to better adapt the algorithm in its simplest form.

4-4 Implementation

In the previous section, the conceptual design of the quay wall detector was executed by identifying the required subfunctions: segmentation, range extraction and line fitting. For each of these subfunctions, a selection of the most adequate algorithms from relevant literature was given, along with the tunable parameters. In this section, the parameters of the algorithms

will be tuned and their respective line fitting performance will be tested w.r.t. to manually determined ground truth measurements from sonar data. Finally, the set of algorithms and corresponding parameters yielding the best performance results (w.r.t the criteria given in Chapter 3) will be chosen for the final design of the quay wall detector.

4-4-1 Setup

Data sets As explained in Chapter 1, the implementation is performed with recorded data from past cleaning operations. Ten sets recorded on six different ships (labeled A, B, C, D, E and F) were gathered, each consisting of 110 consecutive images. The tuning set consists of the first five sets originating from the first three ships (A1, A2, A3, B1, C1), and the validation sets consists of the last five sets originating from the last three ships (D1, D2, D3, E1, F1)⁴. Although limited, the data shows a variation of configurations in terms of distances, orientations, frame rates, viewing range and even quay wall appearance (see Table A-4 for an overview). Figure 4-12 shows the first image of each set.

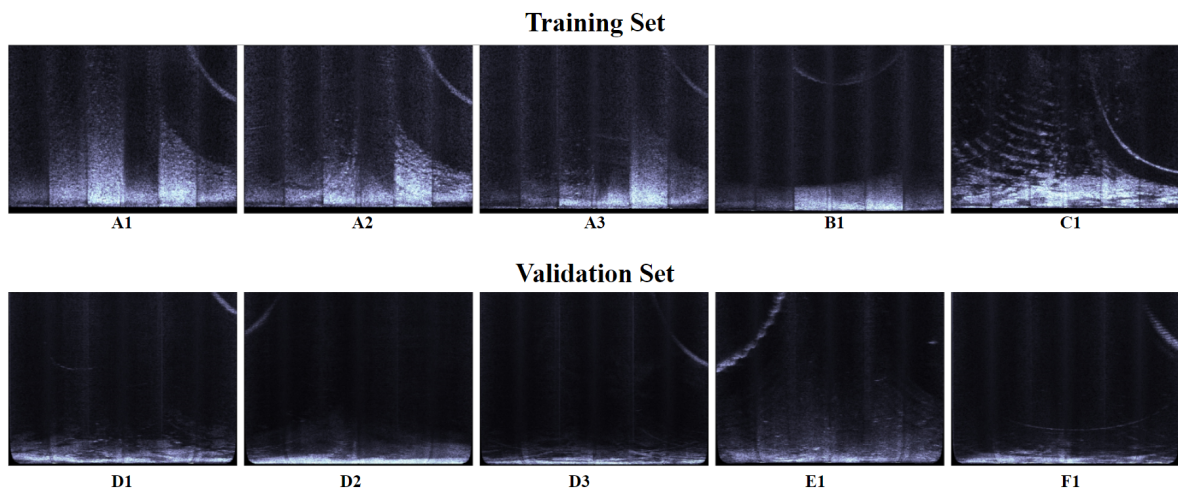


Figure 4-12: Illustration of the sonar data used for tuning and validation, where each image is the first image of the indicated set. The training set (upper row) was recorded on three different ships (Ship A, B and C). The validation set was recorded on three other ships (Ship D, E and F).

Ground truth The ground truth was defined manually by digitally drawing the line that visually best fits the quay wall in the Cartesian image for each image set. The choice is justified by the fact that drawing lines in the Cartesian space is more intuitive than in the polar space, where lines are curved (as can be witnessed in Figure 4-12). This process was executed twice by a single test subject (the author) and the resulting ground truth line parameters were averaged. Given the saccadic nature of the consecutive ground truth values, a Gaussian smoothing filter was applied for ρ_{gt} and θ_{gt} with values of $\sigma_\rho = 1.0$ [m] and $\sigma_\theta = 30$ [deg] which were defined heuristically. The resulting lines were observed to fit the underlying quay wall lines while reducing the previously apparent measurement noise.

⁴It should be noted that the tuning data was recorded by a worn FLS device in contrast with the validation data, which was recorded in a later phase with a newer device with clearly better image quality.

Table 4-1: Overview of the parameters to be tuned for the selected algorithms in the conceptual design phase of the quay wall detector, along with start values. These were determined heuristically. It should be noted that TLS has no parameters requiring tuning.

Function	Parameter	Variable	Unit
Parameter space	Range margin	$d\rho_{prior} = 0.5$	[m]
	Orientation margin	$d\theta_{prior} = 2.5$	[deg]
Region of Interest	Incidence angle	$\beta_{max} = 35$	[deg]
	Image margin	$k_m = 6$	pixels
	ROI margin	$k_r = 25$	pixels
	ROI margin	$k_t = 25$	pixels
	Mask size	$k_{mask} = 20$	pixels
Filtering	Kernel size (Gaussian)	$F_g = 3 \times 3$	pixels
	Kernel size (median)	$F_m = 3 \times 3$	pixels
Contrast	Proportional gain	$c = 0.1$	n/a
	Power gain	$r = 1.7$	n/a
Range Extraction	Standard deviation	$\sigma_r = 10.0$	n/a
Hough Transform	Range resolution	$d\rho = 0.05$	[m]
	Angle resolution	$d\theta = 0.10$	[deg]
RANSAC	Error tolerance	$d_{RANSAC} = 0.01$	[m]
	Fraction of inliers	$\omega = 0.20$	n/a
	Probability	$p = 0.99$	n/a

4-4-2 Tuning procedure

The image segmentation subfunctions along with the line fitting algorithms discussed earlier in this chapter rely on parameters that can be adapted to improve performance depending on the application. Therefore, the parameters for each subfunction and algorithm should be tuned first, with the focus on minimizing the resulting Root-Mean-Square Errors (RMSEs) and maximizing robustness score ACC (as described in Section 3-2-3). Then, the optimal solution will be selected by comparing the results.

Approach The tuning procedure follows a bottom-up approach and contains two methodically distinctive parts. The first part consists of choosing adequate parameters for the prior line's parameter space along with the ROI in order to make the system functional i.e. capable to track a line; this is done heuristically. The second part consists of improving the accuracy of the line estimations w.r.t to ground truths by first tuning the noise filtering, next the contrast enhancement and then line fitting algorithms. As such, performance is improved after each tuning step.

Starting values Before initiating the tuning procedure, the starting values and range of parameters to be explored can be reduced based on heuristic methods. The list of parameters for each of the subfunctions of the quay wall tracking system is shown in Table 4-1. These values were determined as follows:

- The uncertainties of the parameter space were set to $d\rho_{prior} = 0.5$ [m] and $d\theta_{prior} = 2.5$ [deg]. As these values are only used for rejecting outliers by the HT and RANSAC algorithms, these were chosen arbitrarily.
- In [34, 48], values as large as $\beta = 60$ [deg] are reported to return discernible reflection intensities. In this work, similar observations were made, however it was found that

tracking became unstable while including measurements around this value. This was solved by limiting the incidence angle threshold to $\beta_{max} = 50$ [deg].

- Initially, it was assumed that the ROI's margins should be based on the parameter space. However, given the variability of the sonar data in terms of time steps Δt between frames, choosing a fixed mask size in terms of fixed $d\rho_{prior}$ and $d\theta_{prior}$ values gave unstable results. However, a fixed line mask thickness value of $k_{mask} = 40$ pixels (or -20 and $+20$ pixels on each side of the line) was found to be sufficient to consistently enclose quay wall lines.
- As was mentioned in Section 2-1-1, series of noisy illuminated beams are observed at the left and right image boundaries and were found to destabilize measurements. Inspection of the sonar images concluded that this occurs at the first and last five pixel columns. Therefore, the beam rejection margin was chosen at value $k_m = 6$ pixels.
- From there it followed that the ROI margins k_r and k_t required to be larger than $k_{mask}/2$, as otherwise the boundaries of the ROI would cut through the line mask. Therefore, the values were chosen to be $k_r = 25$ and $k_t = 25$ pixels.
- In other works [40, 55], filters are generally kept relatively small, where kernel sizes of $F_{filter} = 3 \times 3$ show satisfactory results, while each of the candidate bilge detectors showed better performance without the use of noise filters in [18]. Therefore, filters no larger than $F_{filter} = 5 \times 5$ are considered.

4-4-3 Total Least-Squares

Table 4-2 shows the tuning results of each subsystem for TLS line fitting, where for each row the resulting performance of a tuned parameter is shown. At the start values for each parameter (Table 4-1) without a noise filter, the TLS algorithm yields $ACC = 92.1\%$ and $RMSE(\rho) = 0.078$ [m] and $RMSE(\theta) = 0.675$ [deg]. With filtering, a Gaussian filter with size $F_g = 5 \times 5$ offers the best improvement and is therefore selected. On the other hand, tuning the contrast enhancement subsystems offers much less significant gains, leading to slight modifications of Weng filter's parameters $c = 0.09$ and $r = 1.71$. Finally, a slight decrease of the range extraction parameter to $\sigma_r = 9.0$ yields the tuned TLS performance to $ACC = 94.2\%$ and $RMSE(\rho) = 0.065$ [m] and $RMSE(\theta) = 0.588$ [deg].

Table 4-2: Overview of the performance results after tuning the filtering and contrast subfunctions along with the TLS line fitting parameters.

Total Least-Squares

Subfunction	Parameter	ACC [%]	RMSE(ρ) [m]	RMSE(θ) [deg]
Filtering	No filter	92.09	0.078	0.675
	$F_g = 5 \times 5$	93.4	0.068	0.605
	$F_m = 3 \times 3$	93.3	0.072	0.652
Contrast	$c = 0.09$	93.5	0.067	0.600
	$r = 1.71$	93.8	0.067	0.599
Range Extraction	$\sigma_r = 9.0$	94.2	0.065	0.588

4-4-4 RANdom SAmple Consensus

The tuning results of the RANSAC line tracker parameters are shown in Table 4-3, starting at performance values of $ACC = 96.0\%$ and $RMSE(\rho) = 0.058$ [m] and $RMSE(\theta) = 0.537$ [deg], which is notably superior to TLS's tuned version. Furthermore, contrarily to TLS implementing filters showed no performance improvements and are therefore discarded. Similarly, the tuning the contrast subsystem yields negligible performance gain, nonetheless the proportional gain was modified to $c = 0.075$ whereas the power gain was held to its start value $r = 1.70$. Setting the range extraction constant to a lower value $\sigma_r = 8.5$ was found to further improve the performance, while the error tolerance $d_{RANSAC} = 0.25$ [m] was kept to its start value since other values showed detectable gains leading to $ACC = 97.4\%$ and $RMSE(\rho) = 0.052$ [m] and $RMSE(\theta) = 0.497$ [deg].

Table 4-3: Overview of the performance results after tuning the filtering and contrast subfunctions along with the RANSAC line fitting parameters. The parameters returning the best results are boldfaced.

RANSAC

Subfunction	Parameter	ACC [%]	RMSE(ρ) [m]	RMSE(θ) [deg]
Filtering	No filter	96.0	0.058	0.537
	$F_g = 5 \times 5$	95.1	0.059	0.529
	$F_m = 3 \times 3$	95.4	0.061	0.550
Contrast	$c = 0.075$	96.2	0.056	0.525
	$r = 1.70$	96.4	0.056	0.524
Range Extraction	$\sigma_r = 8.5$	97.4	0.052	0.497
Line fitting	$d_{RANSAC} = 0.25$	97.4	0.052	0.497

4-4-5 Hough Transform

Before tuning, the Hough Transform's performance tuning procedure can be seen in Table 4-4, where the initial performance starts at $ACC = 90.3\%$ and $RMSE(\rho) = 0.074$ [m] and $RMSE(\theta) = 0.724$ [deg]. Similarly to RANSAC, filters offer no gain and are therefore not used. Yet again, contrast enhancement was only slightly modified, where this time the proportional gain was held to its start value $c = 0.10$ whereas $r = 1.69$ was decreased. The range extraction constant showed the best results at a reduced value of $\sigma_r = 9.0$. Notably, the accuracy of the system only benefited from the increase of the angle resolution $d\rho_{HT} = 0.15$ [deg] whereas the range resolution was kept at the initial value $d\rho_{HT} = 0.10$ [m]. As a result, the tuned HT-based line tracker performance improves to $ACC = 93.1\%$ and $RMSE(\rho) = 0.066$ [m] and $RMSE(\theta) = 0.646$ [deg].

Table 4-4: Overview of the performance results after tuning the filtering and contrast subfunctions along with the HT line fitting parameters. The discarded filters are struck through.

Hough Transform

Subfunction	Parameter	ACC [%]	RMSE(ρ) [m]	RMSE(θ) [deg]
Filtering	No filter	90.3	0.074	0.724
	$F_g = 5 \times 5$	88.55	0.076	0.746
	$F_m = 3 \times 3$	88.91	0.077	0.758
Contrast	$c = 0.100$	90.3	0.074	0.724
	$r = 1.69$	90.6	0.074	0.723
Range Extraction	$\sigma_r = 9.0$	90.9	0.072	0.700
Line Fitting	$d\rho_{HT} = 0.10$	90.9	0.072	0.700
	$d\theta_{HT} = 0.15$	93.1	0.066	0.646

4-5 Validation

In the previous section, the conceptualized candidate solutions for quay wall tracking were implemented. With the help of tuning set of the sonar data, the parameters of each system was tuned for improved performance w.r.t. the manually defined ground truths. In this section the systems will be tested with the validation set and evaluated based on the functional and non-functional requirements defined in Section 3-2-3. From there, an overview of the findings and the limitations along with recommendations will be given in Section 4-6.

4-5-1 Accuracy

Results Table 4-5 shows a comparison of the performance results of all three quay wall tracking systems (TLS, HT and RANSAC) per data set, based on the performance metrics presented in Section 3-2-3. In terms of average score, the TLS algorithm clearly ranks first by showing the lowest RMSE values for both the normal distance ρ^S (0.188 [m] for TLS, 0.235 [m] for RANSAC and 0.268 [m] for HT) and orientation estimations θ^S (0.738 [deg] for TLS versus 0.882 [deg] for RANSAC and 0.928 [deg] for HT) along with the highest accuracy scores (83.5% for TLS against 74.5% for RANSAC and 73.4% for HT). The same conclusion can be drawn for each individual data set, except for F1, where the HT shows superior performance on all metrics, although by a close margin. Moreover, it is interesting to notice the difference between data sets where, regardless of the chosen algorithm, the best performance results are observed for data set D1 whereas data set D2 returns the worst. Another interesting observation is that clearly better performance are returned for D1 compared to D3, although these are circumstantially identical except for viewing range ($r_{stop} = 39.2$ [m] vs. 62.1 [m] respectively, see Table A-4 for more details), causing the line to appear much longer in images of the latter.

Table 4-5: Performance results of the TLS, HT and RANSAC quay wall trackers per data set. For almost all the data sets (except for F1), the TLS-based candidate returns the best results in terms of all three performance metrics, although by a close margin compared to TLS. Furthermore, by the same criteria it shows that RANSAC ranks second and HT shows the lowest overall performance of all three candidates.

			Performance per Data Set						
Variable	[Unit]	Algorithm	D1	D2	D3	E1	F1	Average	
$RMSE(\rho)$	[m]	TLS	0.115	0.268	0.228	0.121	0.210	0.188	
		HT	0.189	0.345	0.330	0.281	0.194	0.268	
		RANSAC	0.129	0.332	0.265	0.236	0.214	0.235	
$RMSE(\theta)$	[deg]	TLS	0.568	0.995	0.711	0.700	0.714	0.738	
		HT	0.749	1.090	0.962	1.137	0.700	0.928	
		RANSAC	0.630	1.201	0.803	1.011	0.765	0.882	
ACC	[%]	TLS	95.0	70.9	79.1	90.9	81.8	83.5	
		HT	90.9	66.8	65.4	60.0	84.1	73.4	
		RANSAC	91.8	60.5	71.8	67.3	81.4	74.5	

System output In Figure 4-13, a visualization of the TLS-based quay wall tracker's output is shown in two parts (TLS was chosen above others for its superior performance). First the estimated quay wall line parameters (ρ^S, θ^S) are compared to the manually defined ground truths ($\rho_{gt}^S, \theta_{gt}^S$), along with the estimated normal distance errors $\varepsilon_\rho = \rho^S - \rho_{gt}^S$ and orientation errors $\varepsilon_\theta = \theta^S - \theta_{gt}^S$ for each of the validation data sets. In the second part, a single example of quay wall line (green line) by the algorithm defined by the normal line (orange line) from the sonar origin (red dot) is shown for each data set. Furthermore, it should be noted that depending on the data set large performance ranges are observed; the RMSE scores range between $0.115 < RMSE(\rho) < 0.268$ [m] and $0.568 < RMSE(\theta) < 0.995$ [deg] and the robustness between $70.9\% < ACC < 95.0\%$. Further inspection of the output data shows that the average mean errors are relatively small, presenting values of $\bar{\varepsilon}_\rho < 0.01$ [m] and $\bar{\varepsilon}_\theta < 0.1$ [deg] respectively. This suggests that the observed RMSE values originate more from variance noise rather than bias in the estimates. Indeed, in the plots of Figure 4-13 it can be seen that consecutive estimates can jump as much as 0.2 [m] apart (data set D3) at intervals of approximately 1 [s]. This noisiness becomes clearly apparent when viewing the tracked line for consecutive sonar frames in real-time (Figure 4-13 shows example images), as the estimated line appears to "vibrate" along the normal axis (orange line). Nonetheless, it is clear that the quay wall line is successfully tracked.

4-5-2 Computation time

The implementation of the algorithms was performed with self-written Python software (the codes can be found in Appendix B) on a personal laptop equipped with an Intel Core i7-8750H CPU running at a clock speed of 2.20 GHz and up to 15.8 Gigabytes of usable RAM. The results of the validation are shown in Table 4-6, where computation times for each of the underlying subfunctions of the tested quay wall tracking algorithms. These are obtained by computing the average processing time for all the images in the validation set.



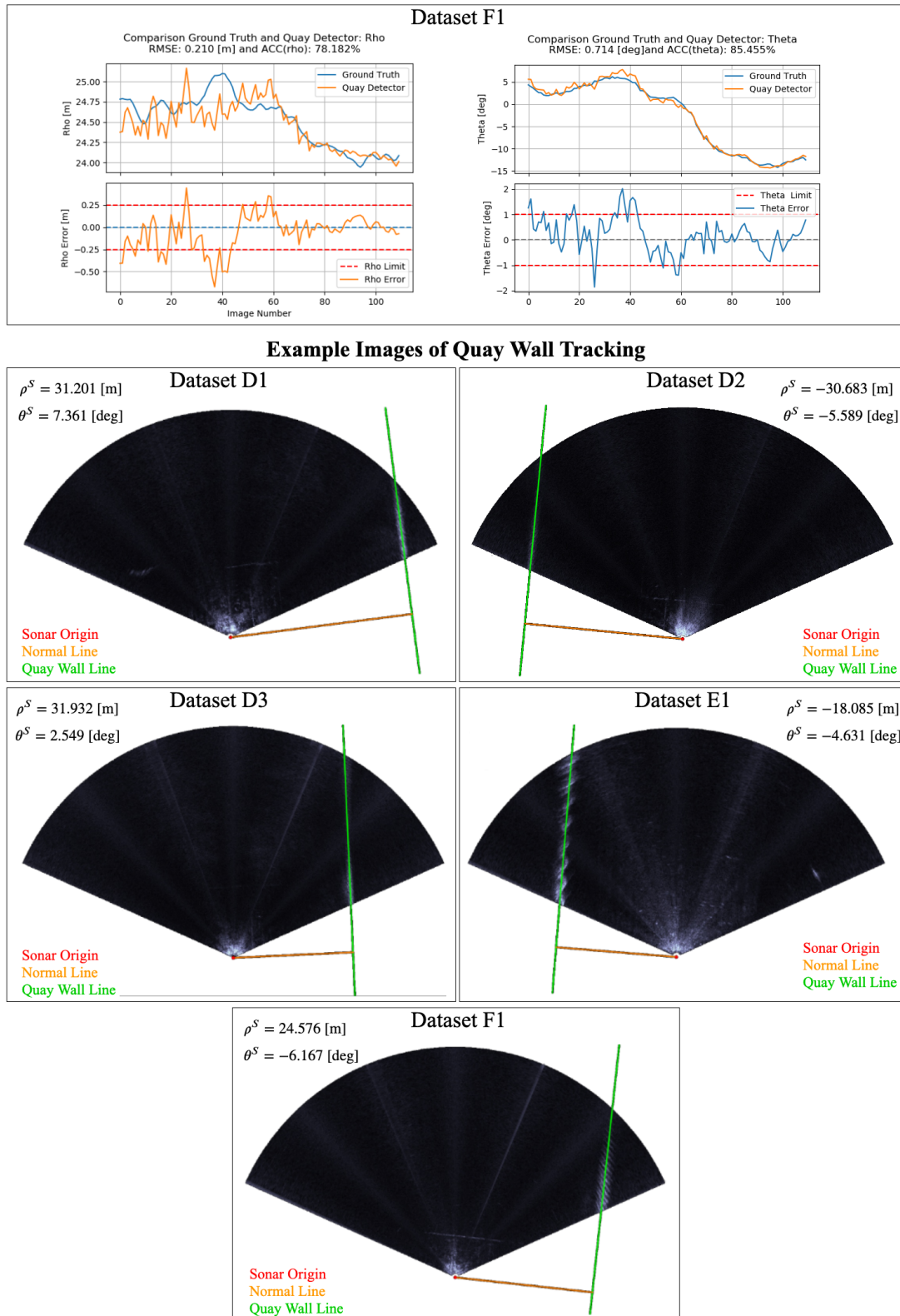


Figure 4-13: Results of the validation test for the TLS-based quay wall tracker, with plots of the estimated quay wall parameters along with ground truths and the corresponding errors. Furthermore, example images of line tracking for each data set are shown for visual purposes.

Table 4-6: Overview of the computation time for each of the subfunctions of each of the quay wall tracker candidates. Extracting the ROI, filtering the image and applying the threshold for segmentation all have negligible computation times ($t_c < 0.001$ [s]), indicated by the hyphens.

Computation Times per Subfunction [s]

Subfunction	TLS	RANSAC	HT
Region of Interest	-	-	-
Filtering	-	-	-
Contrast	0.008	0.008	0.007
Thresholding	-	-	-
Range Extraction	0.007	0.007	0.007
Line Fitting	0.012	0.014	0.048
Total	0.028	0.031	0.064

The results presented in the table show that defining the ROI, filtering the reduced image and applying the threshold can be individually done within a negligible time window ($t_c < 0.001$ [s]). Moreover, the other subfunction preceding line fitting show similar results, which is to be expected since all three candidates only differ in the line fitting methods they use. Indeed, this is where the main computation time difference lies, where the fastest is performed by TLS with an average total of $t_c = 0.028$ [s], followed closely by RANSAC with $t_c = 0.031$ [s], both well under the $t_c < 0.040$ [s] requirement. In contrast, the HT method at $t_c = 0.064$ [s] requires more than twice the time to return an estimate of the quay wall line, violating the computation time requirement by more than 50%.

4-6 Discussion

In the previous sections, the conceptual design of the quay wall tracking system was laid out. Three alternative solutions were proposed and implemented, where the candidate with the best accuracy performance was identified as a proof of concept. From there, this section will discuss the main findings and the limitations along with recommendations for further investigation.

4-6-1 Findings

The results of the validation tests in Section 4-5 along with the visualization demonstrated that tracking the quay wall can be done by all three candidates (TLS, RANSAC and HT). Most notably, the quay wall line tracker using Gaussian noise filtering and TLS line fitting gave the best performance, in terms of accuracy, robustness and computation time. Indeed, the computation time requirement was satisfied by a large margin $t_c = 0.028 < 0.040$ [s]. In terms of accuracy however, it turns out that the quay wall tracker does not meet the accuracy requirements with values at $RMSE(\rho) = 0.188 > 0.125$ [m] and $RMSE(\theta) = 0.738 > 0.500$ [deg]. The same is true for the robustness score, which is over 10% short of the required score $ACC = 83.5\% < 95\%$. Nonetheless, despite the requirements not being met, the system shows to be capable to track the quay wall for varying distances, orientations, frame rates, viewing ranges and appearance of the quay wall. It is believed that this success can be attributed mainly to the benefits of the ROI:

- The system capitalizes on the robot's slow dynamics and the assumption that two consecutive measurements should have similar values, making the choice for the ROI straightforward while keeping the guaranty that the quay wall line can still be found inside that same region. As such, the majority of disturbances and irrelevant objects present in sonar images are easily isolated.
- It also capitalizes on the assumption that no other strong reflections will be found in the neighbourhood of the quay wall, leaving only two classes of pixels (quay wall or background). In conjunction with the reflection amplification from Weng's filter, this plays along the strength of Otsu's method for selecting an adequate threshold to separate two classes of pixel intensities. Also, this method conveniently rejects less bright pixels at higher ranges and incidence angles, lowering the risk for noisy measurements. As such, there is no need for more complex and computationally heavy algorithms to consistently extract the entire line feature of the quay wall.

4-6-2 Limitations

Although the results are encouraging, the proposed system and the findings have some limitations that need to be mentioned.

Limited Data As mentioned in Section 2-4, the amount of data that was used for tuning and testing was limited and was far from uniform (mainly in terms of frame rate, viewing range and distance to the wall). Although it was shown that the system functions at varying circumstances, discrepancies in the performance scores suggest that these have an influence on the system's performance. However, it is hard to draw strong conclusions about the operational characteristics of the proposed system, as only a small sample of possible configurations were captured. For example, given that ships can be up to 60 [m] in breadth, the currently available data does not tell whether the system is applicable on the entire Flat Bottom (FB) surface of such large ships.

Tuning The proposed system was tuned with noisy data from a worn FLS device but validated on images from a new device, returning clearly better imagery. Although this means that the system is tuned to be robust against heavy noise, the found setting is probably not the optimal setting for the less noisy data from the newer device. That is, it is believed that performing the tuning procedure on sonar data that is more representative of the current situation would yield more adequate settings for the quay wall tracking algorithm, with better performance scores as a result.

Wall detection As explained in Section 4-2-1, the system depends on an initial estimate, without which it would be incapable giving an estimate of the quay wall line parameters. Furthermore, it assumes that a quay wall line will be present in the image and that its quality is sufficient for tracking. With Fleet Cleaner's vision for an autonomous robot in mind, a system capable to assess the quality of an image (e.g. similarity [73]), detect the presence and give a prior estimate of the quay wall line to launch the proposed tracking system by itself would be of great interest. Using the Radon Transform (RT) from the bilge detector in [18] is believed to be a good starting point.

4-6-3 Recommendations

From the limitations discussed in the previous paragraph, two main suggestions for future work can be recommended:

Weighted Least-Squares As was mentioned in Section 4-6-1, the segmentation subfunction naturally rejects pixels at higher ranges which are likely to be more noisy. Other than that, the current TLS-based algorithm omits the effect range might have on the noise of range measurements and the corresponding fitted line. Careful analysis of the residuals of the range measurements with respect to the ground truth measurements could provide insights to that relationship with uncertainties. That is, weights depending on the range could be assigned for each scan point e.g. putting more "trust" in closer measurements. Then, the best line fit could be found through a weighted formulation of the TLS, called Weighted Least-Squares (WLS) [74, 75].

Filtering Although the proposed system successfully tracks the quay wall line, it does present noisy measurements causing the accuracy to be insufficient. Given the robot's maximal forward speed of $v = 0.15$ [m/s], the large jumps between consecutive measurements of ρ^S (as large as 0.20 [m] between intervals of 1.0 [s]) suggests the robot's trajectory makes larger lateral jumps than it can move forward, a clear misrepresentation of reality. With the help of models and complementary sensors, predictions of the robot motion and ensuing sensory outputs could be used to filter noise and in turn improve the accuracy of the quay wall tracker's measurements. This method known as sensor fusion, is often implemented in the setting of robot localization [76], where the Kalman filter or its many variants [23] are often used.

4-7 Conclusion

The objective of this chapter was to *provide a proof of concept for an FLS-based algorithm measuring the quay wall line parameters*.

Based on the considerations from the operating conditions (Chapter 2) and requirements (Chapter 3), it was first determined that the FLS could be best used as a rangefinder for wall tracking, with an FLS image and a prior line estimate (ρ^S, θ^S) as inputs. Then, the required image processing steps to achieve this were identified: determining the Region of Interest, noise filtering, contrast enhancement, segmentation, range extraction and line fitting. A selection of the most commonly used paradigms from the state-of-the-art was chosen to propose three candidate solutions (TLS, RANSAC and HT). Each of the candidates were implemented by first tuning their respective parameters to maximize accuracy performance across the tuning data set w.r.t. manually defined ground truth line measurements. Afterwards, the algorithms were validated with a separate set, where the TLS solution arose as a proof of concept with the best performance. Its computation time is well within the limits ($t_c = 0.028 < 0.040$ [s]), however the required accuracy and robustness was not achieved ($RMSE(\rho) = 0.188 > 0.125$ [m], $RMSE(\theta) = 0.738 < 0.500$ [deg] and $ACC = 84.0 < 95.0\%$). This partial success is linked with the use of an ROI conveniently rejecting noise

All in all, the objective of this chapter was only partially achieved in that a fast enough proof-of-concept for FLS-based quay wall measurements was delivered, however not with the required accuracy performance. This is believed to be linked to the low quality of the data used for tuning, and further solvable by means of filters.

Drift-free Localization

The previous chapter has shown the design of the quay wall tracker, which serves as the front-end the localization system by delivering absolute measurements of the robot's pose. From there, the objective of this chapter is to *provide a proof of concept fusing Forward Looking Sonar (FLS) measurements with auxiliary sensors to enable drift-free localization on flat bottoms*. To achieve this, the same approach as in Chapter 4 will be adopted. First, a conceptual design with a selection of required subfunctions will be proposed. Then, the system will be implemented and validated, followed by a discussion of the results.

5-1 Subfunctions for localization

The previous section has shown the implementation and validation of a quay wall tracking system based on FLS images, which outputs the normal distance and orientation of the quay wall line w.r.t. to the sonar frame. With the goal of performing accurate and absolute localization of the robot in mind, translating measurements to the robot frame w.r.t. the ship frame is an essential first step. Furthermore, as was shown in Chapter 4, the measurements of the quay wall line feature are not perfectly accurate as they do not give a realistic representation of the robot trajectory. This problem can be alleviated by means sensor fusion [14]. As filters rely on models that describe the robot's motion and predict the sensor output from a previous state, such models should be formulated for the Fleet Cleaner robot. All in all, to achieve the sought after goal two tasks require to be fulfilled:

- **State estimation:** to translate the sensor measurements into motion estimates, make predictions of future poses and sensors outputs, all to estimate the pose of the robot.
- **Filtering:** to improve the accuracy of pose estimations from the selected sensors based on the constructed models.

5-2 State estimation

As mentioned in Section 1-1-2, fundamental aspects of localization are the ability to perceive the environment and the estimating motion. Indeed, both information channels can be combined to estimate the state of the robot i.e. its pose and eventual derivatives (e.g. speed). This is done with the help of models, which will serve later on for the filtering step in Section 5-3.

Models According to [24], when considering the state of a mobile robot moving through an environment, the following quantities require to be defined for each time instant k :

- $\mathbf{x}_k$: the state vector describing the pose of the vehicle i.e. (x, y, ψ)
- $\mathbf{u}_k$: the control input at time $k - 1$ which drives the vehicle to state $\mathbf{x}_k$ at time k .
- $\mathbf{z}_k$: measurements taken from the vehicle at time k .
- $\mathbf{w}_k$: the motion disturbance at time k (acting on $\mathbf{u}_k$)
- $\mathbf{v}_k$: the measurement noise at time k (acting on $\mathbf{z}_k$)

These form the basis for constructing the motion model (also called process model) and an observation model (also called sensor model) to incrementally describe the state and the corresponding sensor output of the robot:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \quad (5-1)$$

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{v}_k) \quad (5-2)$$

where $\mathbf{f}(\cdot)$ is the function for the motion model and $\mathbf{h}(\cdot)$ is the function for the observation model. The rest of this section will give detailed description of these models. This will serve as a basis for the filtering step later on (Section 5-3).

5-2-1 Motion model

Prior to this thesis, K. Cassee proposed a motion model based on odometry from the front wheel [17]. However, due to the unpredictability and high occurrence of lateral slip at the front wheel in practice¹ [5], an alternative model describing motion at the centre of the rear-axle is preferred in this work.

¹Currently, the robot does not perform differential steering. That is, all three wheels apply equal torque regardless of the corner radius (or conversely the steering angle), leading to oftentimes unpredictable behaviour e.g. driving forward while steering left or right.

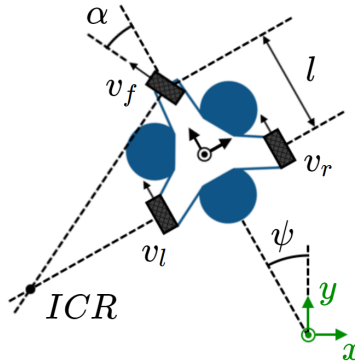


Figure 5-1: Schematic of the robot's motion model, which depends on steering angle α , forward speed v (defined as the minimum between front, right and left wheel speeds v_f , v_r , v_l as defined in Equation (2-6) and wheelbase l). The change in heading depends on the instantaneous centre of rotation ICR and the robot's forward speed.

Rear axle model The advantage of choosing the rear-axle model is that translations and rotations are kinematically decoupled. In other words, the linear displacement of the rear-axle in the x - and y directions is independent of the steering angle α . Also, considering that the force from rear wheel torque is directed in the longitudinal direction (local x -axis) and most slip occurs at the front wheel in practice (in other words, the front wheel has the least grip meaning lateral forces arising from steering are negligible), it is fair to assume that no lateral slip is to be expected at the rear wheels. Then, a simple coordinate transform can be executed to obtain the coordinates of the robot origin instead of the rear axle:

$$\begin{bmatrix} x_{\mathcal{R}} \\ y_{\mathcal{R}} \\ \psi_{\mathcal{R}} \end{bmatrix} = \begin{bmatrix} x_{ra} + l_{ra} \cos(\psi_{ra}) \\ y_{ra} + l_{ra} \sin(\psi_{ra}) \\ \psi_{ra} \end{bmatrix} \quad (5-3)$$

where l_{ra} is the distance of the rear axle to the robot origin $\mathcal{R}$. Inspection of the robot's kinematics in Figure 5-1 shows that the robot's position change depends on the forward speed and orientation, while the change of heading is related to the forward speed and steering angle. Assuming no lateral slip, this leads to the following nonlinear transition model:

$$\dot{\mathbf{x}}_k = \begin{bmatrix} \dot{x}_k \\ \dot{y}_k \\ \dot{\psi}_k \end{bmatrix} = \begin{bmatrix} v_k \cdot \cos \psi_k \\ v_k \cdot \sin \psi_k \\ v_k \cdot \frac{\sin \alpha_k}{l} \end{bmatrix} \quad (5-4)$$

According to [77], a discrete-time first-order Taylor approximation (of the form $\mathbf{x}_k = \mathbf{x}_{k-1} + \dot{\mathbf{x}}_{k-1} \cdot dt$) incorporating the transition model is reported to a sufficiently accurate motion model:

$$\mathbf{x}_k = \begin{bmatrix} x_k \\ y_k \\ \psi_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + v_{k-1} \cdot \cos(\psi_{k-1}) \cdot dt \\ y_{k-1} + v_{k-1} \cdot \sin(\psi_{k-1}) \cdot dt \\ \psi_{k-1} + v_{k-1} \cdot \frac{\sin(\alpha_{k-1})}{l} \cdot dt \end{bmatrix} \quad (5-5)$$

where l is the wheel base, dt is the time step, $\mathbf{x}_k = [x_k, y_k, \psi_k]^\top$ is the robot pose at time instance k , $\mathbf{u}_k = [v_k, \alpha_k]^\top$ is the control input consisting of the longitudinal speed v_k at the rear-axle driving the robot forward and the effective steering angle α_k causing the change in heading.

Uncertainty However, as explained earlier, it follows that the slip at the wheels cause disturbances acting on control input $\mathbf{u}_k$ and in turn in the robot's motion towards its new state. It is assumed that both inputs are independently disturbed by $\mathbf{w}_k = [\varepsilon_v, \varepsilon_\alpha]^\top$, a zero-mean Gaussian noise with covariance $\mathbf{Q}_{k-1}$:

$$\mathbf{w}_k = \begin{bmatrix} \varepsilon_v \\ \varepsilon_\alpha \end{bmatrix} \sim \mathcal{N}(0, \mathbf{Q}_{k-1}), \quad \mathbf{Q}_{k-1} = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_\alpha^2 \end{bmatrix} \quad (5-6)$$

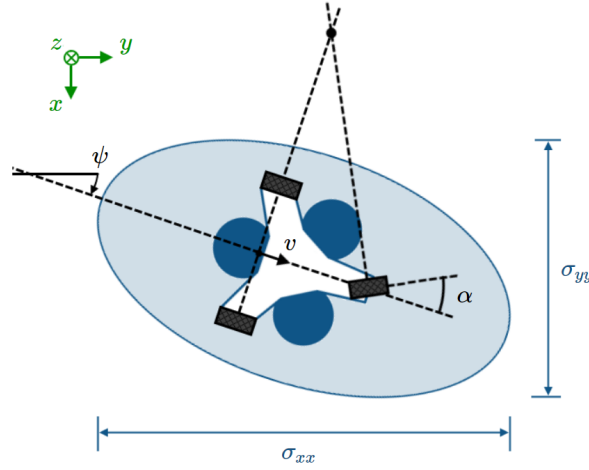


Figure 5-2: Illustration of the local position (x, y) covariance of the robot. Clearly, the global uncertainties σ_{xx} and σ_{yy} depend on the local uncertainty leads of the robot's displacement and its orientation ψ .

Where σ_v^2 and σ_α^2 are the variances of the speed and steering angle disturbances. From the inspection of Figure 5-2, it is clear that noise on v creates an uncertainty in the global pose of the robot, for which the x - and y -components vary w.r.t. the robot's heading. That is, the noise is said to be state dependent (or non-additive) [78]. In order to translate the state dependent uncertainty from input $\mathbf{u}_k$ that propagates into the robot state, the following transformation is applied:

$$\mathbf{P}_{u,k-1} = \mathbf{L}_{k-1} \mathbf{Q}_{k-1} \mathbf{L}_{k-1}^\top \quad (5-7)$$

where $\mathbf{L}_{k-1}$ is the Jacobian of the motion model in Equation (5-5) w.r.t. the input noise:

$$\mathbf{L}_{k-1} = \left. \frac{\partial \mathbf{f}}{\partial \mathbf{w}} \right|_{\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}} = \begin{bmatrix} \cos(\psi_{k-1}) \cdot dt & 0 \\ \sin(\psi_{k-1}) \cdot dt & 0 \\ \frac{\sin(\alpha_{k-1})}{l} \cdot dt & v_{k-1} \frac{\cos(\alpha_{k-1})}{l} \cdot dt \end{bmatrix} \quad (5-8)$$

5-2-2 Sensor model

Similarly to motion models, with proper knowledge of the sensor characteristics it is possible to predict the output of sensors along with their uncertainties, which will be useful for filtering later on. Then, the quay wall tracker output will be translated such that it directly relates to ship-relative localization of the robot and from there the sensor model will be defined.

Local to global mapping

As explained in Section 3-2-2, the quay wall tracking system expresses the line parameters of the quay-wall relative to the local sonar frame $\mathcal{S}$. However, localization of the robot requires to relate local robot frame $\mathcal{R}$ to the global ship frame $\mathcal{B}$. This means that the line tracker measurements require to be converted to global measurements, this is illustrated in Figure 5-3. Since the quay wall only represents a lateral and heading reference as described in Section 3-3-1, the longitudinal x -coordinate is not considered.

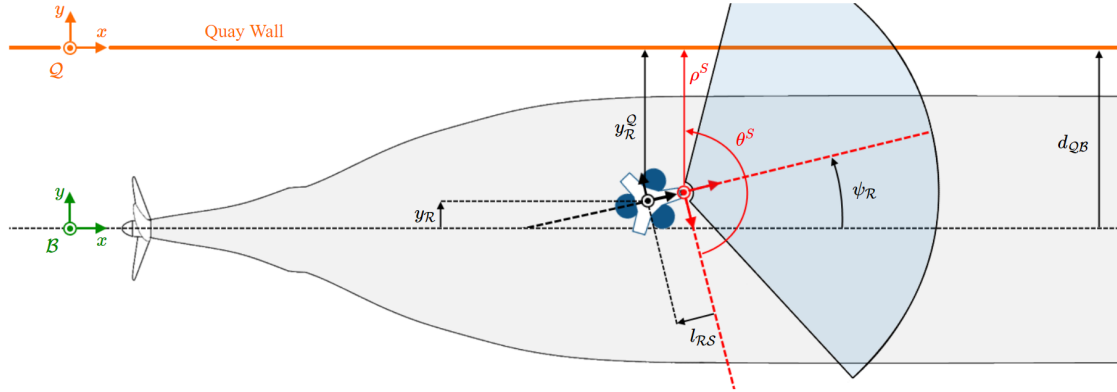


Figure 5-3: Illustration of localization using quay wall tracking. Clearly, the measurements ρ^S and θ^S require some conversion before obtaining the robot lateral and heading position y_R and ψ_R . First, measurements from the sonar are related to the quay-relative frame $\mathcal{Q}$. Then, using the offset d_{QB} , these can be related to the ship frame $\mathcal{B}$. Finally, measurements from the sonar frame are translated to the robot frame by using the longitudinal offset l_{RS} .

Conversions Figure 5-3 shows the geometry of the localization problem when the quay wall is situated on the left of the Flat Bottom (FB). Based on inspection of the figure, the first conversion transforms the sonar observations of the quay wall to a quay-wall-relative pose of the sonar as follows:

$$(R) : \begin{cases} y_S^Q = \rho^S \\ \psi_S^Q = -\theta^S \end{cases} \quad (L) : \begin{cases} y_S^Q = -\rho^S \\ \psi_S^Q = \pi - \theta^S \end{cases} \quad (5-9)$$

Where (R) and (L) correspond to the case when the quay wall is situated on the right and the left, respectively. The second transformation shifts the pose estimation of the sonar frame $\mathcal{S}$ from the quay-wall-relative frame $\mathcal{Q}$ to the ship-relative frame $\mathcal{B}$, using the offset d_{QB} between the two frames as described in Section 3-3-1:

$$(R) : \begin{cases} y_S = \rho^S - d_{QB} \\ \psi_S = -\theta^S \end{cases} \quad (L) : \begin{cases} y_S = -(\rho^S - d_{QB}) \\ \psi_S = \pi - \theta^S \end{cases} \quad (5-10)$$

Then, a simple coordinate transform from sonar frame $\mathcal{S}$ to robot frame $\mathcal{R}$ can be obtained using a-priori known dimensions of the robot:

$$x_{\mathcal{R}} = x_{\mathcal{S}} - l_{\mathcal{RS}} \cos \psi_{\mathcal{S}} \quad (5-11)$$

$$y_{\mathcal{R}} = y_{\mathcal{S}} - l_{\mathcal{RS}} \sin \psi_{\mathcal{S}} \quad (5-12)$$

$$\psi_{\mathcal{R}} = \psi_{\mathcal{S}} \quad (5-13)$$

where $l_{\mathcal{RS}}$ is the distance from the sonar to the robot frame in the local x -direction. Filling in the equations from (5-10) for $y_{\mathcal{R}}$ and $\psi_{\mathcal{R}}$ makes it possible to relate quay wall measurements $(\rho^{\mathcal{S}}, \theta^{\mathcal{S}})$ to the robot's lateral and heading pose for both quay wall cases:

$$(R) : \begin{cases} y_{\mathcal{R}} = \rho^{\mathcal{S}} - d_{QB} + l_{\mathcal{RS}} \sin(\theta^{\mathcal{S}}) \\ \psi_{\mathcal{R}} = -\theta^{\mathcal{S}} \end{cases} \quad (5-14)$$

$$(L) : \begin{cases} y_{\mathcal{R}} = -(\rho^{\mathcal{S}} - d_{QB}) - l_{\mathcal{RS}} \sin(\theta^{\mathcal{S}}) \\ \psi_{\mathcal{R}} = \pi - \theta^{\mathcal{S}} \end{cases} \quad (5-15)$$

Where identities $\sin(-\theta) = -\sin(\theta)$ and $\sin(\pi - \theta) = \sin(\theta)$ were used for simplification. To create a single equation for both cases, a constant k_Q is defined in order to take the quay wall position into account:

$$k_Q = \begin{cases} -1, & \text{if } \theta_q = \frac{\pi}{2} \quad (\text{left}) \\ 1, & \text{if } \theta_q = -\frac{\pi}{2} \quad (\text{right}) \end{cases} \quad (5-16)$$

Then, the equations for the global positions measurements from the quay wall tracker becomes:

$$y_{\mathcal{R}} = k_Q (\rho^{\mathcal{S}} - d_{QB} + l_{\mathcal{RS}} \sin \theta^{\mathcal{S}}) \quad (5-17)$$

$$\psi_{\mathcal{R}} = (1 - k_Q) \frac{\pi}{2} - \theta^{\mathcal{S}} \quad (5-18)$$

Measurement model

From the motion model in Equation (5-5), it follows that the state of the robot can be estimated from the measurements of forward speed v and steering angle α , captured by the wheel encoders filled in Equation (2-6) and the steering encoder respectively. Still, as was explained in Section 2-3, this method (also known as odometry) is insufficient due to the build up of errors from slip over time. This justifies the interest in incorporating the quay wall tracker to obtain absolute measurements of y_k and ψ_k , while longitudinal position x_k remains unreferenced nonetheless. Since it is beyond the scope of this thesis, no attempt to obtain absolute measurements for x_k will be made. In contrast with the motion model, the sensor noise is assumed to be independent of the robot state, resulting in the following model with additive noise:

$$\mathbf{z}_k = \begin{bmatrix} z_{y,k} \\ z_{\psi,k} \\ z_{v,k} \\ z_{\alpha,k} \end{bmatrix} = \begin{bmatrix} y_k \\ \psi_k \\ v_k \\ \alpha_k \end{bmatrix} + \mathbf{v}_k \quad (5-19)$$

Uncertainties

The uncertainties corrupting the sensor measurements is defined by $\mathbf{v}_k$, a zero-mean Gaussian noise with covariance matrix $\mathbf{R}_k$, composed of the quay wall tracker uncertainty $\mathbf{R}_{y\psi}$ and uncertainty from wheel and steering encoders $\mathbf{R}_{v\alpha}$:

$$\mathbf{v}_k = \begin{bmatrix} \varepsilon_y \\ \varepsilon_\psi \\ \varepsilon_v \\ \varepsilon_\alpha \end{bmatrix} \sim \mathcal{N}(0, \mathbf{R}_k), \quad \mathbf{R}_k = \begin{bmatrix} \mathbf{R}_{y\psi} & 0 \\ 0 & \mathbf{R}_{v\alpha} \end{bmatrix} \quad (5-20)$$

Encoder uncertainty Since the uncertainties of forward speed and steering inputs $\mathbf{u}_k = [v, \alpha]^\top$ are considered independent from each other in the motion model, the same can be assumed for their counterpart in the sensor variances σ_{zv}^2 and $\sigma_{z\alpha}^2$:

$$\mathbf{R}_{v\alpha} = \begin{bmatrix} \sigma_{zv}^2 & 0 \\ 0 & \sigma_{z\alpha}^2 \end{bmatrix} \quad (5-21)$$

Quay wall tracker uncertainty If we recall from equations (4-26) through (4-31) in Chapter 4, the quay wall tracker's measurements (ρ^S, θ^S) are determined by applying Total Least-Squares (TLS) line fitting on Cartesian scan points (x_i, y_i) translated from polar scan points (r_i, t_i) for each beam. The presence of θ^S in Equation (4-27) for estimating normal distance $\rho^S = \bar{x} \cos \theta^S + \bar{y} \sin \theta^S$ clearly suggests that there is a nonlinear relationship between errors in ρ^S and θ^S , meaning that assuming independent uncertainties would be ill-defined. This is also clear from a visual stand point, where the sign of the angle error yields different magnitudes of normal distance errors shown in Figure 5-4.

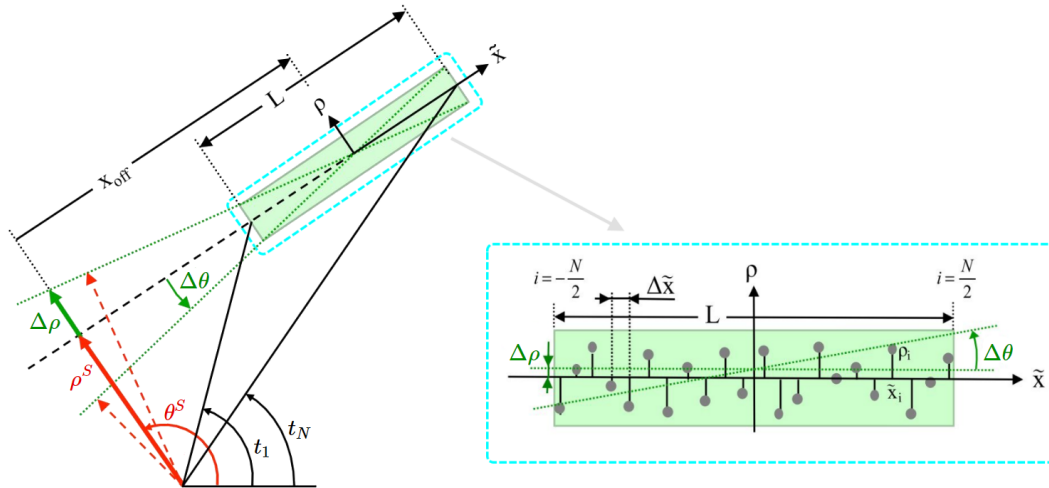


Figure 5-4: Illustration of the line uncertainty model as defined in [75]. Clearly, depending on the sign of orientation error $\Delta\theta$ (green dotted lines), errors in ρ^S with different magnitudes arise (red dotted arrows). Furthermore, the individual range discrepancies $\rho_i - \rho^S$ for each scan point also contribute in that error.

More realistically, the line uncertainty should be estimated from known range and bearing uncertainties σ_r and σ_t for each scan point. With knowledge of the geometry of scan points the uncertainty matrix for the line parameters can be determined analytically, as was demonstrated in [74] for a Maximum Likelihood (ML) scheme for line fitting. A slightly simpler solution proposed in [79] obtains the uncertainty matrix $\mathbf{R}_{\rho\theta}$, while only requiring the range uncertainty σ_r . However, neither the range nor bearing uncertainties are specified in the documentation of the FLS used in this work. Nonetheless, in [75] it is shown that it is possible to approximate this line covariance without knowledge of σ_r and σ_θ for each scan point. Using an assumption of N equally-spaced points with uncorrelated errors, the uncertainty covariance matrix can be simplified to the following closed-form error model:

$$\mathbf{R}_{\rho\theta} = \begin{bmatrix} S_{\rho\rho} & S_{\rho\theta} \\ S_{\theta\rho} & S_{\theta\theta} \end{bmatrix} \approx \sigma_\rho^2 \cdot \begin{bmatrix} \frac{12 \cdot x_{off}^2}{L^2 \cdot N} + \frac{1}{N} & \frac{-12 \cdot x_{off}}{L^2 \cdot N} \\ \frac{-12 \cdot x_{off}}{L^2 \cdot N} & \frac{12}{L^2 \cdot N} \end{bmatrix} \quad (5-22)$$

where σ_ρ^2 is the variance of the scan points' ranges w.r.t. the estimated line, the length L of the line segment formed by the scan points and the offset x_{off} between its centre and the normal line (as shown in Figure 5-4). These are in turn defined by:

$$\sigma_\rho^2 = \frac{1}{N} \sum_{i=1}^N (\rho_i - \rho^S) \quad (5-23)$$

$$= \frac{1}{N} \sum_{i=1}^N (x_i \cos \theta^S + y_i \sin \theta^S - \rho^S) \quad (5-24)$$

$$x_{off} = \frac{\rho^S}{N} \cdot \sum_{i=1}^N \tan(\theta^S - \theta_i) \quad (5-25)$$

$$L = \rho^S \cdot |\tan(\theta^S - t_N) - \tan(\theta^S - t_1)| \quad (5-26)$$

where t_1 and t_N are the first and last beam angles of the considered scan points. Since the uncertainty $\mathbf{R}_{\rho\theta}$ defines the uncertainty of local sonar measurements ρ^S and θ^S , it requires to be translated to global uncertainties of y and ψ . In order to do so, first the Jacobian describing the partial derivatives w.r.t. ρ^S and θ^S needs to be defined:

$$\mathbf{J}_{y\psi} = \begin{bmatrix} \frac{\partial y}{\partial \rho} & \frac{\partial y}{\partial \theta} \\ \frac{\partial \psi}{\partial \rho} & \frac{\partial \psi}{\partial \theta} \end{bmatrix} = \begin{bmatrix} k_Q & k_Q \cdot l_{RS} \cdot \cos \theta^S \\ 0 & -1 \end{bmatrix} \quad (5-27)$$

From there, the uncertainty $\mathbf{R}_{y\psi}$ can be defined using the Jacobian from the previous equation:

$$\mathbf{R}_{y\psi} = \mathbf{J}_{y\psi} \cdot \mathbf{R}_{\rho\theta} \cdot \mathbf{J}_{y\psi}^\top = \begin{bmatrix} \sigma_{yy}^2 & \sigma_{y\psi}^2 \\ \sigma_{\psi y}^2 & \sigma_{\psi\psi}^2 \end{bmatrix} \quad (5-28)$$

where σ_{yy}^2 and $\sigma_{\psi\psi}^2$ are the positional measurement uncertainties and $\sigma_{y\psi}^2 = \sigma_{\psi y}^2$ are the covariances between the positional uncertainties arising from the transformed uncertainty $\mathbf{R}_{\rho\theta}$.

5-3 Filtering

Thanks to the previously defined models, the addition of absolute measurements (quay wall tracker) next to relative measurements (odometry) for localization is enabled. However, the measurements from the quay wall tracker suggests trajectories that would entail large lateral displacements within short time windows, violating the no-lateral-slip assumption. Although the tracker's measurements are accurate over long time periods, this violation suggests that its short-term accuracy is poor. As mentioned in Section 1-1-2, the short-term accuracy of relative measurements can be combined with the long-term accuracy of absolute measurements by means of sensor fusion to obtain better estimates. This section will first present common methods identified in the literature survey prior to this thesis [5] and then select the most adequate solution for the given use case. Then, a more in depth description of the selected algorithm will be given and from there, the proposed system will be implemented in the next section.

5-3-1 Methods

Within the large body of research of mobile robot localization, according to [13, 80, 17, 77] the most widely applied sensor fusion techniques are the Extended Kalman Filter (EKF) [81], the Unscented Kalman Filter (UKF) [82] and the Particle Filter (PF) [83]. In this section, the general principles and properties of each method in terms of accuracy and computation will be discussed, from which the most adequate solution for this application will be selected.

Extended Kalman Filter

For linear Markovian processes² with Additive White Gaussian Noise (AWGN) uncertainties, the Kalman filter is the optimal Bayesian estimator [22] and since its introduction many variants have been developed [23]. For mobile robot with nonlinear dynamics like the Fleet Cleaner, the "EKF is undoubtedly the most widely used nonlinear state estimation technique that has been applied in the past few decades" according to [78].

Working principle The main principle of the EKF is to estimate the mean and covariance of the state by linearizing the nonlinear motion and sensor model from a single prior estimate. The algorithm consists of two steps; first, the new state and the uncertainty covariance is predicted from the previously estimated state and control input (prediction step). Then, a least mean square approach fusing the prediction and measurement is applied to obtain a statistically optimized estimate with reduced uncertainty. This is illustrated in Figure 5-5. Furthermore, the interested reader is referred to [81] for derivations of the algorithm.

²A process in which the present state depends only on the previous one.

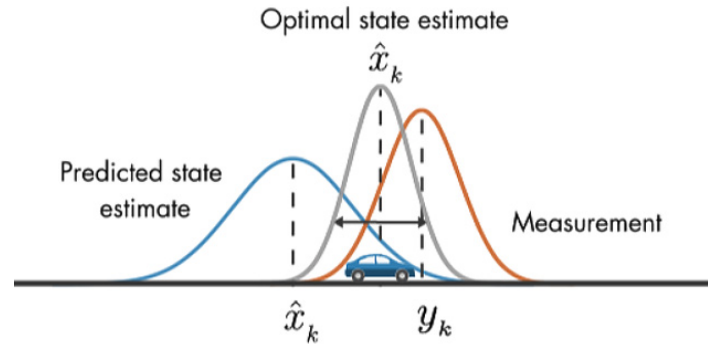


Figure 5-5: Illustration of the Kalman filter's optimal estimation in the one-dimensional case. First, a state estimate $\hat{x}_k$ is predicted from a previous estimate and control input. Then, the measurement is used to optimize the state estimate. Figure from <https://www.mathworks.com/videos/understanding-kalman-filters-part-3-optimal-state-estimator-1490710645421.html>, accessed on 24 February 2020.

Properties The main weakness of the EKF is that linearization can lead to poor representations of the nonlinear functions and of the associated probability distributions [84] (this is illustrated in Figure 5-6). It has the tendency to underestimate noise, with the risk of divergence as a result. Furthermore, one should be careful when defining the initial state estimate and motion model; if modeled incorrectly, the filter will also diverge. This can be prevented however, for example by adapting the EKF gain using an adaptive fuzzy logic system [85, 86]. Nevertheless, the EKF has been applied with great success in state estimation problems that violate the underlying AWGN assumption [6]. It is reported to be accurate to the first order [77] and in comparison with Unscented Kalman Filter and Particle Filter it carries considerably less computational complexity [87].

Unscented Kalman Filter

The UKF [82] has been developed in recent years to address the main problem of the EKF, i.e. the poor approximating properties of the linear approximation and the ensuing proneness to divergence for highly nonlinear and noisy systems.

Working principle The basic idea behind the UKF is that of finding a transformation that allows to approximate the mean and covariance of a random vector of length n when it is transformed by a nonlinear map. Rather than using a single point like the EKF, this is done by computing a set of $2n + 1$ weighted points, called sigma-points, on the basis of the mean and covariance of the original vector, propagating these points through the nonlinear map and then approximating the mean and variance of the transformed vector from the propagated sigma-points (this is illustrated in Figure 5-6). The interested reader is referred to [82] for the theoretical aspects.

Properties Unlike the EKF, thanks to the use of sigma-points the UKF does not require the computation of partial derivatives (or Jacobians), meaning that it is also suitable for

non-differentiable functions [82]. Furthermore, its superior approximation of propagated uncertainties makes it more suitable for systems with highly nonlinear dynamics and as a result its state estimates are accurate up to the third order (versus first order for the UKF) [77]. The main drawback of the UKF however is the added computational complexity due to the sampling of sigma-points [17].

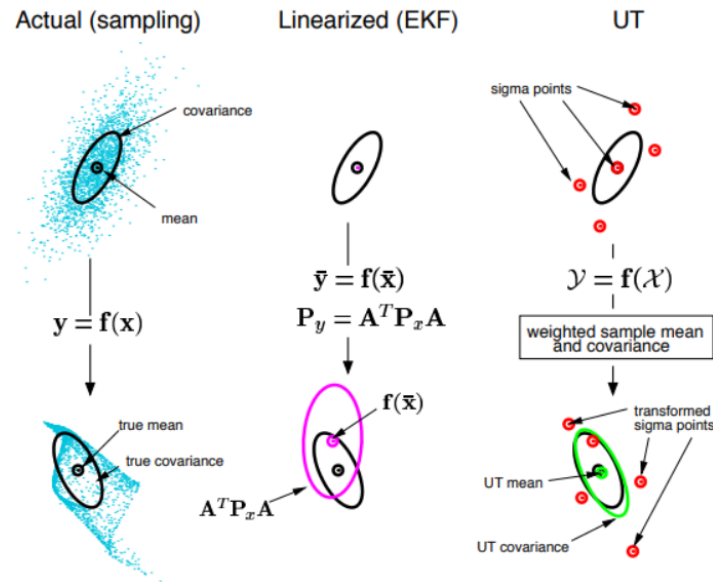


Figure 5-6: Illustration of the uncertainty propagation of the UKF in comparison to an actual sampling and the EKF. Clearly, a set of Gaussian distributed particles (blue points) with given mean and covariance do not yield a Gaussian distribution after a nonlinear transformation $y = f(x)$. The EKF however, applies a linearization $P_y = A^T P_x A$ around a single point (magenta point, i.e. the mean of the prior estimate) yielding another Gaussian distribution, a poor estimate of the true posterior mean and covariance. The UKF on the other hand, uses multiple (sigma-)points (red points) which are propagated through the nonlinear map, resulting in a better approximation (green ellipse). Figure from [82].

Particle Filter

Also in response to the inadequacy of EKF's, the interest in a filter capable to handle both non-Gaussian and nonlinear processes gave rise to the Particle Filter (originally called the *bootstrap filter* [83]), becoming one of the most widely used filters ever since [88].

Working principle The main principle of the PF is that the distribution of the state vector is represented by a set of random state samples with associated weights (or particles), instead of an assumed Gaussian distribution described by its mean and covariance. To accommodate this, the generic algorithm consists of three steps (illustrated in Figure 5-7). First, the set of samples from the previously estimated state is propagated through the motion model to create a proposal distribution (prediction). Next, upon arrival of a new measurement, the proposed samples are assigned weights to reflect the probability of any hypothesized state, or particle, given the measurement (update). Finally, a new set is generated at random according to the updated distribution, in order to represent new state estimate (resampling).

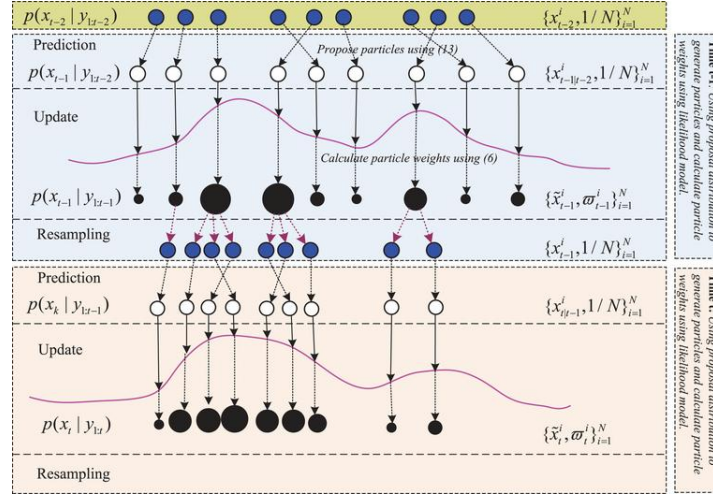


Figure 5-7: Illustration of two consecutive iterations of the particle filter working principle. Given a distribution of particles, a prediction for each particles next state is made. In the update phase the particles weights are updated using the likelihood model. In the last step the particles are resampled according to the weights of the updated particles, and so on.

Properties The greatest benefit from the sampling method is that PFs are not restricted to Gaussian uncertainties and prove to be particularly effective for highly non-linear systems [83]. However, they are significantly more computationally heavy due to the dependence on a large number of particles for accuracy [13]. Furthermore, PFs are particularly known to suffer from sample degeneracy and impoverishment. The former is the apparent widening of the particle distribution, while sample impoverishment can be viewed as particles being overly concentrated, both of which cause the particle distribution to become inaccurate [89].

Filter choice

Accuracy In [17], the choice of the EKF was justified by the insignificant advantages of the UKF and PF in dealing with system non-linearity and the non-Gaussian distribution of the robot. Indeed, given the motion model based on a first order Taylor approximation, implementing the UKF for its third order accuracy would be superfluous. Moreover, as was mentioned in Section 2-3, the motion of the robot is mostly close to linear, suggesting that implementing UKF or PF is unlikely to present significant gains over the EKF.

Computation time As mentioned previously, in comparison with other methods the EKF is computationally speaking the most straightforward method given differentiable models, as is the case here. Although the developers of the UKF report similar complexity [82], it is often deemed more computationally heavy, even more so for the PF as it represents one of its main weaknesses [17].

Selection From these considerations, it follows that the EKF is chosen for its computational simplicity and the insignificant gain from more accurate filters. A description of the algorithm will be given in the next section.

5-3-2 Extended Kalman Filter

Predict The first step of the EKF predicts the new state and uncertainty covariance matrix using the previously defined state and uncertainty estimates $\hat{\mathbf{x}}_{k-1|k-1}$ and $\mathbf{P}_{k-1}$:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k) \quad (5-29)$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1} \mathbf{P}_{k-1} \mathbf{F}_{k-1}^\top + \mathbf{P}_{u,k-1} \quad (5-30)$$

The predicted uncertainty is derived from the sum of the motion uncertainty from the control input $\mathbf{P}_{u,k-1}$ as defined in Equation (5-7) and the propagation of the previous state uncertainty $\mathbf{P}_{k-1}$ through the linearized motion model $\mathbf{F}_{k-1}$:

$$\mathbf{F}_{k-1} = \left. \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right|_{\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k} = \begin{bmatrix} 1 & 0 & -v_k \sin(\hat{\psi}_{k-1}) \cdot dt \\ 0 & 1 & v_k \cos(\hat{\psi}_{k-1}) \cdot dt \\ 0 & 0 & 1 \end{bmatrix} \quad (5-31)$$

Before initiating the correction step, the sensor model is also linearized as $\mathbf{H}_k$ at the predicted operating point $\hat{\mathbf{x}}_{k|k-1}$:

$$\mathbf{H}_k = \left. \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \right|_{\hat{\mathbf{x}}_{k|k-1}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (5-32)$$

Update Upon a new observation $\mathbf{z}_k$, the measurement and the linearized sensor model are used to update the predictions as follows:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}) \quad (5-33)$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R}_k \quad (5-34)$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^\top \mathbf{S}_k^{-1} \quad (5-35)$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \quad (5-36)$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^\top \quad (5-37)$$

Where $\tilde{\mathbf{y}}_k$ is the measurement residual, $\mathbf{S}_k$ the ensuing residual covariance, $\mathbf{R}_k$ is the measurement covariance as defined in Equation (5-20) and $\mathbf{K}_k$ is the near-optimal Kalman gain to obtain the updated state and covariance estimates $\hat{\mathbf{x}}_{k|k}$ and $\mathbf{P}_{k|k}$.

5-4 Implementation

In the previous sections, the motion and sensor model describing the behaviour of the robot and its sensory outputs along with their respective uncertainties were constructed. These serve as the basis for the selected filter, the EKF, fusing the quay wall tracker with odometry sensors for improved state estimation. In this section, the localization system will be implemented; first, the filter will be tuned, then its performance will be tested. This will be done by first comparing the performance results for the upgraded system to the unfiltered quay wall tracker, and then by estimating the trajectories from data sets to evaluate the drift. From there, an overall discussion of the results and their limitations will be laid out.

5-4-1 Setup

Data sets As was mentioned in Section 2-4, due to issues with the navigational data, only a limited amount are useful for implementation in combination with the available sonar images (for more details on the data refer to Table A-4). As a result, only images from data sets D1, D2, D3 and E1 from the validation are used for implementation. For performance testing, the first 110 images from each set will be used in the same manner as in the previous chapter. Then the full set of images is used (varying between 200 to over 1000 images depending on the set) to estimate the trajectory from quay wall measurements. Furthermore, as sonar and navigational data are not recorded synchronously, a linear interpolation was applied to tackle the issue.

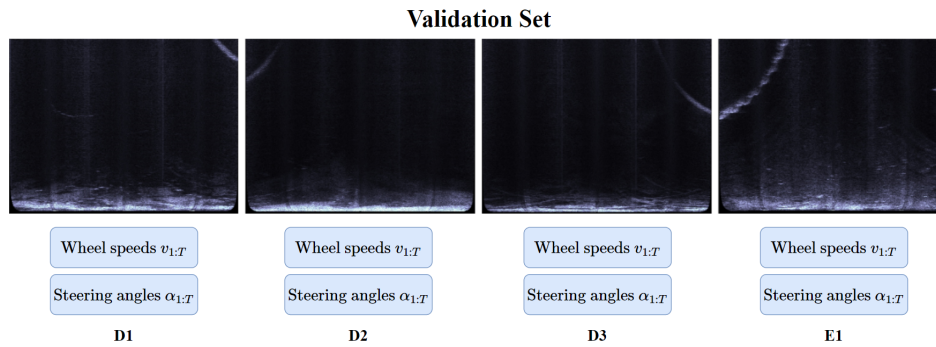


Figure 5-8: Illustration of the used data set for validation. The available navigational data is composed of the wheel speeds and steering angles that were measured throughout the recordings of data sets D1, D2, D3 and E1

Ground truth Given the size of some of the data sets (over 1000 images) and the labour intensive nature of defining ground truths, the manually defined ground truths were not further extended for the entire sets. That is, the same ground truth measurements as in Section 4-4-1 are used for the corresponding data sets. However, to enable the evaluation of drift, ground truth measurements of the final pose of the robot w.r.t. quay wall were manually defined. This was done by taking the average of ten measurements at the final image for each set.

5-4-2 Tuning

From the explanation of the EKF algorithm in Section 5-3-2, it is clear that the filtering procedure depends on motion and sensor uncertainties $\mathbf{Q}_k$ and $\mathbf{R}_k$, both of which can be tuned in order to better reflect the system dynamics, as such improving pose tracking performance. Roughly speaking, the former is a measure of the "trust" in the prediction, whereas the latter is a measure of the "trust" in the sensor measurement.

Approach In contrast with the tuning procedure of the quay wall tracker, the tuning procedure here will not go further than the heuristic approach. The reason for this is that it was found from preliminary trials that little gain is to be made w.r.t. the manually determined

ground truths, while it also became apparent that the ground truth's accuracy is limited (this will be discussed further in Section 5-6). It follows that an elaborate tuning procedure to better fit imperfect data would be superfluous.

Motion model In Section 5-2-1, the motion model was shown to be driven by forward speed v and steering angle α inputs, with an uncertainty covariance modeled by $\mathbf{Q}_k$. For the speed uncertainty, a values below $\sigma_v = 0.001$ [m/s] were found to return stable results. Indeed, higher values cause the estimator to back track longitudinal positions, which is clearly not possible for a forward moving robot. Next, given the assumption that the front wheel often suffers from slip, a high uncertainty is excepted for the input steering angle α . With maximum steering angles at $\alpha_{max} = \pm 34$ [deg], disturbances above that value are very unlikely. On this basis, it is assumed that the uncertainty must fall within three standard deviations $3\sigma_\alpha < \alpha_{max}$ (corresponding to 99.7% of disturbances ε_α falling within $\pm\alpha_{max}$). This yields a standard deviation for steering inputs with value $\sigma_\alpha < 34/3 \approx 11.3$ [deg]. This very pessimistic value was chosen in order to capture the eventuality of the robot driving straight despite steering at α_{max} , as is sometimes the case in practice.

Sensor model As explained in Section 5-2-2, the sensor model is divided in two parts: the quay wall tracker and the encoders. The uncertainties of the quay wall tracker require no tuning since these are determined from the set of scan points (x_i, y_i) for each incoming sonar image, as defined by Equation (5-28). Therefore, only the uncertainty of encoders measurements require tuning. In [17], values at stationary were found to be $\sigma_{z\alpha} = 0.023$ [deg] and $\sigma_{zv} = 0.006$ [m/s]. Although it was pointed out that it might not reflect the operational conditions in [17], the work in [81] reports that poor models with EKF's can be compensated by injecting enough uncertainty. On top of that, additional disturbances are to be expected while navigating compared to stationary conditions, for that reason the values were doubled to take this consideration into account. It should be noted however, that no significant differences are observed for different values.

Tuned parameters Table 5-1 shows an overview of the resulting motion and sensor model parameters for the EKF sensor fusion scheme. This presents the final configuration of the localization system will be validated in the following section by testing its performance w.r.t. ground truth measurements and its robustness against drift compared to other methods.

Table 5-1: Overview of the parameters to be tuned for EKF motion and sensor models. Note that the first three parameters of the sensor models have no definite value since these are defined from scan points by Equation (5-28), which change for each incoming sonar image.

Function	Parameter	Variable	Unit
Motion model	Forward speed	$\sigma_v =$	0.001 [m/s]
	Steering angle	$\sigma_\alpha =$	11.30 [deg]
Sensor model	Normal distance	$\sigma_{yy} =$	n/a [m]
	Orientation	$\sigma_{\psi\psi} =$	n/a [deg]
	Covariance	$\sigma_{y\psi} =$	n/a [m · deg]
	Forward speed	$\sigma_{zv} =$	0.012 [m/s]
	Steering angle	$\sigma_{z\alpha} =$	0.046 [deg]

5-5 Validation

In the previous sections, an EKF-based sensor fusion scheme was chosen to improve the quay wall tracker's estimations of the robot's pose, which was tuned heuristically. In this section, the upgraded system will be validated based on the performance metrics defined in Section 3-3-3, that is, accuracy, drift and computation time.

5-5-1 Accuracy

Results The performance results of the EKF-based state estimator in comparison with the unfiltered quay wall tracker are compiled in Table 5-2. Clearly, the filtered state estimator shows superior performance w.r.t. all performance metrics, with $RMSE(\rho) = 0.116 < 0.183$ [m] (36.7% error decrease), $RMSE(\theta) = 0.606 < 0.738$ [m] (18.5% error decrease) and $ACC = 93.9 > 84.0\%$ (9.9% accuracy increase). Clearly, the strongest improvement is observed for data set D3, where errors are almost halved and robustness increases significantly as well (-43.4% for $RMSE(\rho)$, -41.8% for $RMSE(\theta)$ and +18.6% for ACC). Also, it should be noted that there is a clear contrast of results between data sets, where yet again D2 shows significantly inferior performance compared to the rest. Furthermore, while D3 originally showed mediocre performance compared to D1, it now shows superior performance on two of the three performance metrics ($RMSE(\theta)$ and ACC), suggesting the filter effectively suppresses the noise disturbances described in the previous chapter.

Table 5-2: Performance results of the unfiltered TLS quay wall trackers (labelled as TLS) and the EKF-based system (labelled as EKF), per data set. Clearly, the EKF-based system shows improvements on all metrics compared to the unfiltered quay wall tracker.

		Performance per Data Set					
Variable	[Unit]	Algorithm	D1	D2	D3	E1	Average
$RMSE(\rho)$	[m]	TLS	0.115	0.268	0.228	0.121	0.183
		EKF	0.102	0.153	0.129	0.079	0.116
$RMSE(\theta)$	[deg]	TLS	0.568	0.995	0.711	0.700	0.738
		EKF	0.478	0.922	0.449	0.574	0.606
ACC	[%]	TLS	95.0	70.9	79.1	90.9	84.0
		EKF	96.8	82.7	97.7	98.2	93.9

System output In Figure 5-9, the same visualization of the filtered quay wall tracker's output is shown as in the previous chapter. First, the state estimations of the filtered quay wall tracker are translated back to the local measurements (ρ^S, θ^S). Then the estimated quay wall line parameters (ρ^S, θ^S) are plotted and compared to the manually defined ground truths ($\rho_{gt}^S, \theta_{gt}^S$), along with the estimated normal distance errors $\varepsilon_\rho = \rho^S - \rho_{gt}^S$ and orientation errors $\varepsilon_\theta = \theta^S - \theta_{gt}^S$ for each of the validation data sets. The most obvious improvement compared to the unfiltered quay wall tracker in Figure 4-13, is that estimations are much more stable as can be witnessed by the smooth line instead of the previously noisy line. This reflects the higher ACC values suggesting that less outliers occur. Interestingly, the measurements are so stable that the ground truth measurements appear to be the noisy ones instead. Furthermore, viewing the filtered line estimates for each consecutive sonar frame in real-time clearly shows that the previously apparent "vibrating" noise mentioned in Section 4-5-1 is successfully suppressed when using the EKF filtering scheme, while still closely tracking the quay wall line.



Figure 5-9: Comparison between ground truth and EKF-filtered quay wall tracker estimation of ρ and θ . In comparison with the unfiltered measurements shown in Figure 4-13, one particularly interesting observation can be made: the ρ measurements are clearly less noisy, so much that the ground truth measurements are the ones appearing to be noisy.

5-5-2 Drift

In the previous section, the quay wall tracker upgraded with an EKF was evaluated based on the accuracy performance measures w.r.t. ground truths for the 110 first images of each data set. This section will show the estimated trajectories constructed from the entire set of images within each data. From there, the resulting drift w.r.t. to the ground truth end position will be evaluated.

Results In order to offer material for comparison, the drift from three different estimation methods are compared:

- Current system (as explained in Section 2-3)
- Odometry (filling in encoder data in the motion model in Equation (5-5))
- Quay wall tracking with EKF filtering

Table 5-3 compiles the resulting drift for each method and for each data set. Clearly, the filtered quay wall tracker has the lowest drift, with values at $EBU(y) = 0.130$ [m] and $EBU(\psi) = 0.459$ [deg] on average. In contrast, the disproportionately large error build-ups from odometry with values estimated at $EBU(y) = 16.975$ [m] and $EBU(\theta) = 150.799$ [deg] clearly show its inadequacy for drift free localization, as introduced in Section 1-2-1. Furthermore, the current system, although to a lesser extent, is also clearly inferior compared to the EKF-based tracking system with errors building up to $EBU(y) = 1.023$ [m] and $EBU(\psi) = 4.020$ [deg].

Table 5-3: Comparison of the drift lateral drift $EBU(y)$ and heading drift $EBU(\psi)$ between the different systems.

Variable	[Unit]	Algorithm	Drift per Data Set				Average
			D1	D2	D3	E1	
$EBU(y)$	[m]	EKF	0.172	0.084	0.242	0.021	0.130
		Current	2.199	0.220	0.804	0.868	1.023
		Odometry	20.261	12.766	31.105	3.769	16.975
$EBU(\psi)$	[deg]	EKF	0.391	0.642	0.336	0.468	0.459
		Current	2.499	6.205	1.567	5.808	4.020
		Odometry	222.099	195.587	171.463	14.049	150.799
Total Distance	[m]	All	52.099	22.949	52.875	44.424	43.087

Trajectories Figures 5-10 and 5-11 give an overview of the resulting trajectories for each of the considered methods, for each data set. In order to show the improvement from EKF filtering, the trajectory as estimated by the unfiltered quay wall tracker's range measurements is also included in the figure. Indeed, it becomes clear that the reduction in noise results in a much smoother estimated trajectory, which is a better representation of reality. Moreover, as the numbers have shown, it is obvious that odometry is very prone to drift, as showcased by its almost immediate deviation in three of the four cases (D1, D2 and D3). In contrast, the current system oversimplifies the trajectory of the robot, by drawing a line along a single direction which clearly contradicts the visual evidence from viewing sonar recordings suggesting that the trajectories are rather sinuous.

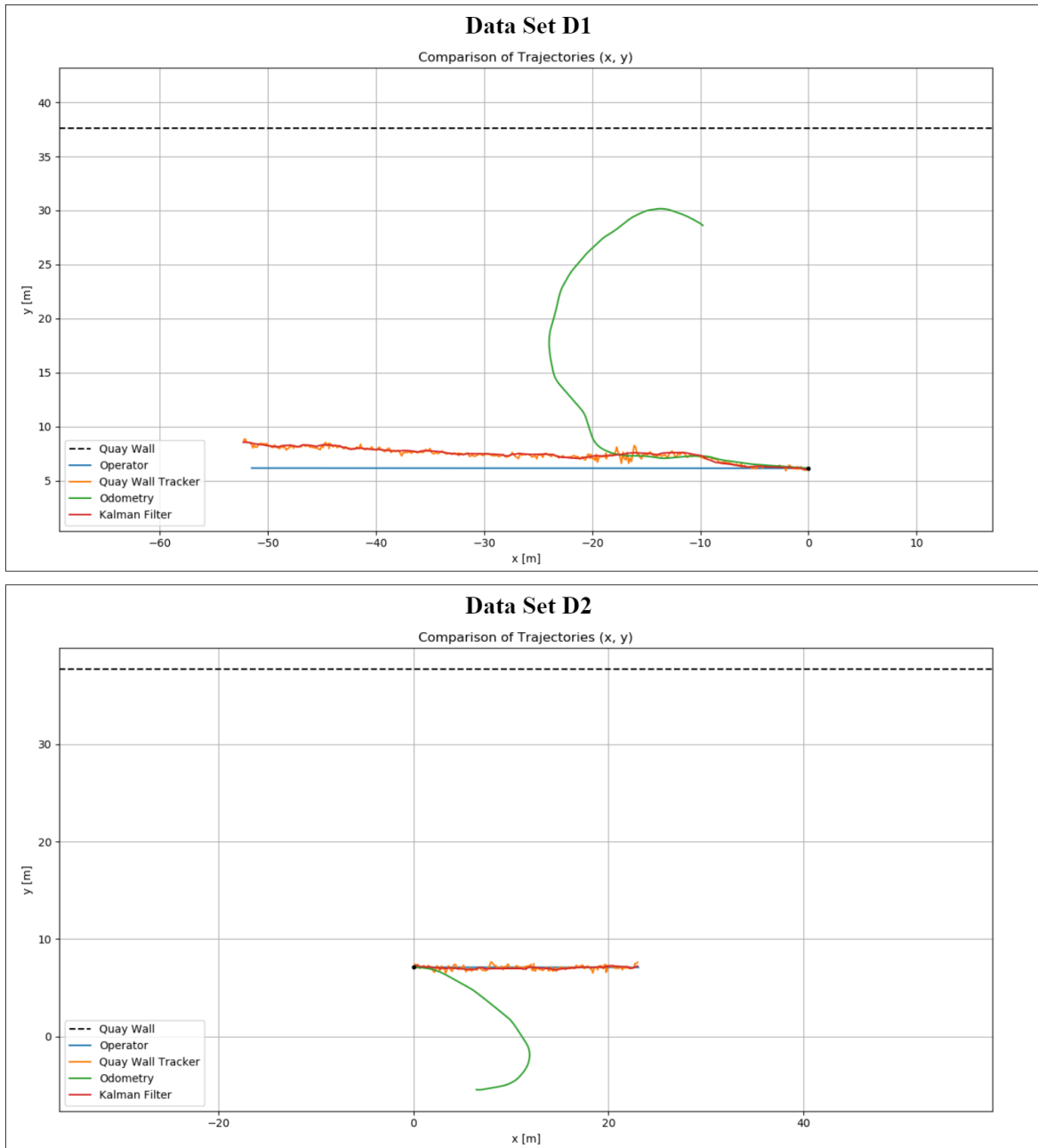


Figure 5-10: Trajectories of data sets D1 and D2 along with the estimated quay wall position (dotted black line), as estimated by the localization system ("Operator"), odometry ("Odometry"), the unfiltered quay wall tracker ("Quay Wall Tracker") and the one with an EKF filter ("Kalman Filter"). Clearly, odometry is a mediocre estimator for long term estimations (traveled distance beyond 20 [m] in this case) although it does show a "smooth" trajectory should be expected, while the current system does not reflect the imperfections of the robot's approximately straight trajectories. The quay wall tracker was shown to be accurately tracking the position overall, although its apparent noisiness suggests mediocre short-term accuracy. With the EKF, a much smoother result is obtained.

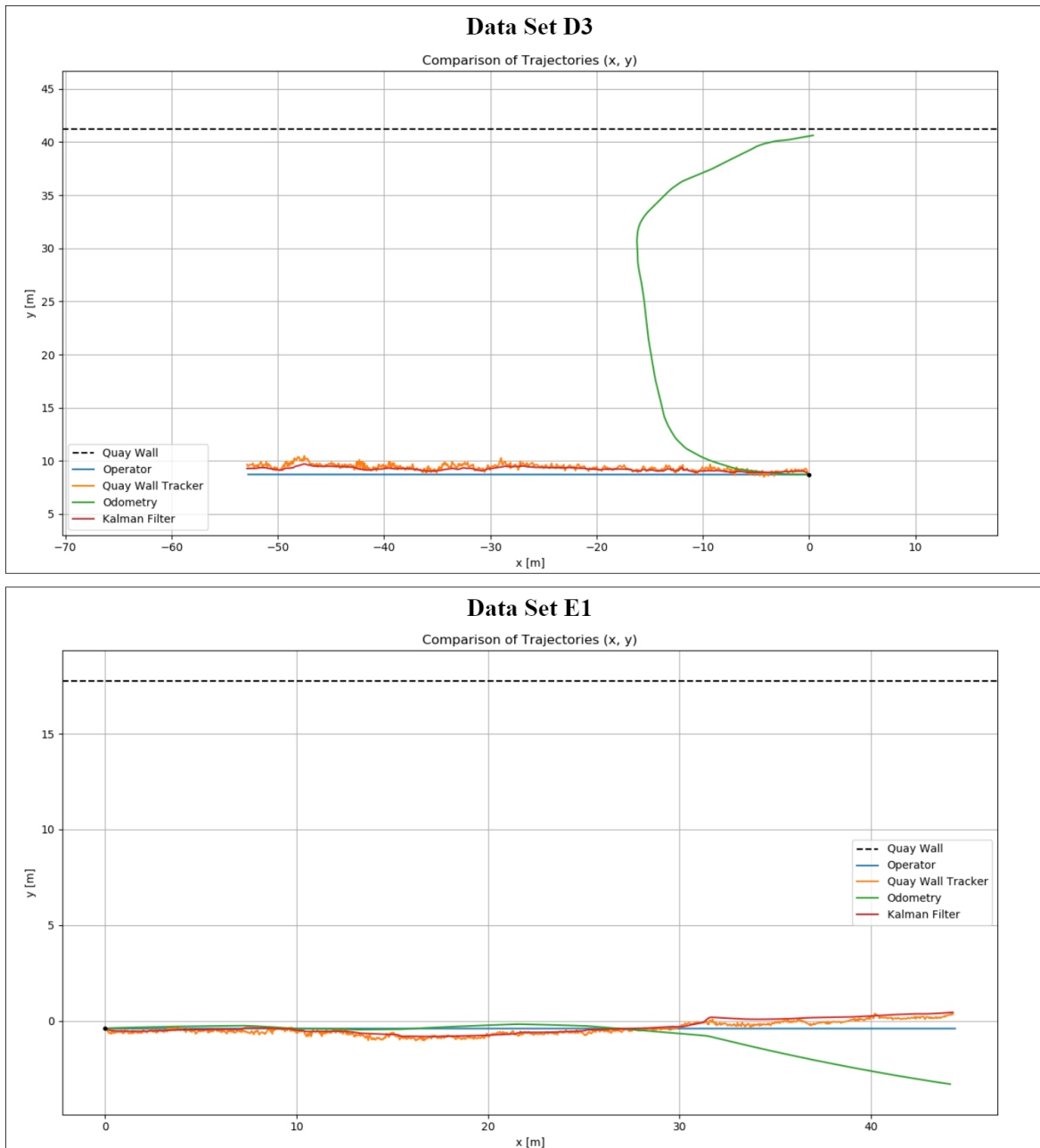


Figure 5-11: Trajectories of data sets D3 and E1 along with the estimated quay wall position (dotted black line), as estimated by the localization system ("Operator"), odometry ("Odometry"), the unfiltered quay wall tracker ("Quay Wall Tracker") and the one with an EKF filter ("Kalman Filter"). As was shown in the previous figure, odometry has very irregular results, immediately drifting with D3 while for E1 it remains fairly accurately up to 30 [m] from the starting point, while the current system idealizes the straightness of the path.

5-5-3 Computation time

In this section, the subsystems will not be considered again, since it was already shown in Section 4-5-2 that the quay wall tracking system they compose satisfied the computation requirement $t_c < 0.040$ [s]. Instead, it is investigate whether the addition of the EKF-based state estimation scheme still satisfies the computation time limit. Apart from that, the experiments are performed under the same conditions as described in Section 4-5-2, were the results are presented in Table 5-4. It turns out that the EKF adds no significant computation time, with an average total computation time of $t_c = 0.023 < 0.040$ [s].

Table 5-4: Computation times for the front-end and back-end of the localization system. On average, extracting quay wall tracker measurements requires $t_c = 0.023$ [s], leaving an margin average of 0.017 [s] for the EKF-based state estimator to compute the state of the robot. Conveniently, with a negligible computation time of $t_c < 0.001$ [s], it falls within this margin with ease.

Computation Times per Subfunction [s]

Subfunction	TLS
Quay Wall Tracking	0.023
Extended Kalman Filter	< 0.001
Total	0.023

5-6 Discussion

In the previous sections, the conceptual design of the localization system using the quay wall tracker was justified, where the EKF was chosen as a sensor fusion paradigm. It was afterwards implemented and validated by using recorded data from passed cleaning operations. Based on this data, the performance in terms of accuracy, computation time and drift were evaluated. From there, the main findings and limitations will be presented in this section along with recommendations for future research.

5-6-1 Findings

First and foremost, it was found that the implementation of an EKF drastically improved the performance of the quay wall tracker in terms of accuracy and robustness score. With these improvements, the required accuracy score for normal distance measurements is achieved with $RMSE(\rho) = 0.116 < 0.125$ [m], however the score for orientation errors remain larger than was desired at $RMSE(\theta) = 0.606 < 0.500$ [deg], neither was the robustness score at $ACC = 93.9 < 95.0\%$. It should be noted however, that on the individual data set level, D1 meets all requirements while D3 and E1 only show small Root-Mean-Square Error (RMSE) infringements. In contrast, none of the requirements are met for D2, suggesting that its conditions are worse than other data sets. On a more positive note, the computation time requirements are easily met thanks to the minimal computational complexity of the EKF with a total of $t_c = 0.023 < 0.040$ [s]. Nonetheless, the slight excess of RMSEs does not impede the system to successfully reduce localization drift while estimating the robot trajectory, with satisfying error build-up scores evaluated at $EBU(y) = 0.130 < 0.250$ [m]

and $EBU(\psi) = 0.459 < 1.000$ [deg], something the other methods were not capable of. It is believed that this partial success can be linked to one factor:

- As was mentioned in Section 5-3, the EKF is particularly well-suited for systems whose dynamics are close to linear, which is in general the case for the robot as it drives along approximately straight trajectories. Moreover, by using predictions from the models, constraints on the measurements can be set as such suppressing the noise. However, when the assumptions of the models become invalid, the performance could degrade quickly. This is believed to explain the inferior performance observed for data set D2, where the low image update rate in combination with the more sinuous nature of the trajectory invalidates the linearity assumption.

5-6-2 Limitations

Ground truth As was explained earlier, the ground truth measurements were defined manually and averaged to compensate for possible human error. For the more approximate unfiltered quay wall tracker, these measurements were helpful in indicating inaccuracies of the system. However, for the more precise EKF-based measurements, the imperfect ground truth potentially distorts its performance values by exaggerating the magnitude of the errors. In that sense, the found RMSE values might be an indication of the inferiority of the ground truths in comparison with the the proposed system's output. On top of that, it was explained in the discussion of the previous chapter that the quay wall tracker is not optimally tuned due to differences in training and tuning set (i.e. images from old device vs. new device). For these reasons it was considered superfluous to tune the filter parameters more elaborately for better accuracy performance.

Sensor selection Next to the FLS and the quay wall tracking scheme, only the odometry sensors were considered (measuring speed v and steering angle α). While in practice it is considered to predict the robot's forward motion sufficiently accurately, the prediction of the robot's heading change is particularly poor, as witnessed by the occurrences of severe drift in the estimated trajectories. However, the gyro included in the Inertial Measurement Unit (IMU) is capable to measure this change directly instead of being predicted from the two noisy odometry measurements and could therefore potentially be a better predictor³. As such, more accurate predictions would further improve the accuracy of the EKF-based estimations of the robot poses.

5-6-3 Recommendations

In the future, some additions could be considered in order to add to the autonomous capabilities of the robot.

³Given that the problem of magnetic disturbances mentioned in Section 2-3 are solved.

Flat Bottom detection In Section 2-3, it was mentioned that ensuring that the robot is situated on the Flat Bottom of the ship is among the tasks of operators, next to ensuring the presence of the quay wall in sonar images. Obviously, it is essential to perform this assessment correctly since the proposed system relies on the assumption of two-dimensional planar motion, which would justify the interest in its automation. Indeed, along with the quay wall detector recommended in the previous chapter, this system could ensure that the quay wall tracker is only launched once the assumptions upon which it rests are valid, as such creating a more robust autonomous system.

Path planning and control As mentioned earlier, the sometimes sinuous nature of trajectories and the ensuing invalidation of assumptions of linear motion are believed to contribute to the degradation of the EKF's accuracy. These are due mostly to the operator's inattention after long operations, with frequent steering corrections as a result. It follows that by implementing a path planning and control scheme capable to follow a more linear path would benefit the accuracy performance of the EKF. With the quay wall as a reference, this would amount to solving the wall-following problem [90].

5-7 Conclusion

The objective of this chapter was to *provide a proof of concept fusing FLS measurements with auxiliary sensors to enable drift-free localization on flat bottoms*.

To achieve this, the first step was to construct two models for estimating the state of the robot: a model describing the robot's motion according to given control inputs (forward speed and steering angle) and a model for corresponding sensory outputs including the quay wall tracker, along with uncertainties arising from motion and sensor disturbances modeled by zero-mean Gaussian noise. Afterwards, a selection of the most widely applied filters for improved localization was made, comprising the of the EKF, the UKF and the Particle Filter. Based on considerations on accuracy and computational complexity, the EKF was selected for its simplicity and the insignificant gain from the other more complex methods. Then, the filtering algorithm was laid out and related to the previously defined models. From there, the implementation phase was initiated by first heuristically tuning the model parameters for the filter instead of iteratively. Then, the proposed system was validated showed considerably better performance compared to the unfiltered counterpart presented in the previous chapter. Still, only the accuracy of normal distance measurements were satisfying ($RMSE(\rho) = 0.116 < 0.125$ [m]) while the orientation accuracy ($RMSE(\theta) = 0.606 > 0.500$ [deg]) nor the robustness ($ACC = 93.9 < 95.0\%$) were insufficient. However, this did not keep the system from fulfilling the requirements for reduced localization drift ($EBU(y) = 0.130 < 0.250$ [m] and ($EBU(\psi) = 0.459 < 1.000$ [deg]).

All in all, the goal of the chapter was achieved in that a proof-of-concept was delivered and validated, enabling reduced-drift localization of the Fleet Cleaner robot on FBs. The requirement for computation time was achieved, however the accuracy requirements were not all achieved; this is believed to be linked to the imperfection of both the manually defined ground truths and the tuning of the underlying quay wall tracker.

Chapter 6

Closing remarks

In this concluding chapter, the main takeaways of each chapters of this thesis will be summarized. In conjunction with the summary, the objectives that were formulated in the preliminary chapters will be evaluated. Based on these evaluations, recommendations for further research will be listed.

6-1 Summary

In order to successfully complete cleaning operations, the ability for the robot to keep track of its pose on the Vertical Sides (VSs) and the Flat Bottom (FB) of a ship hull is essential. While automated localization on the VS is possible, the current system for localization of the Fleet Cleaner robot on the Flat Bottom of a ship is insufficient. That is, the available sensors do not offer absolute pose measurements and therefore the system relies instead on the operator's interpretation of Forward Looking Sonar (FLS) data and his manual inputs to represent an oversimplified trajectory, with estimation errors building up over time as a result. In the future, Fleet Cleaner envisions to develop an automated localization system to alleviate the reliance on human input and eliminate drift. By formulating the goal to *develop a proof-of-concept for a sonar-based framework enabling drift-free localization of a ship-hull cleaning robot on flat bottoms*, this thesis aims to contribute in Fleet Cleaner's vision. This goal was divided in two objectives, the first aims to enable absolutely referenced sensing from sonar images by *providing a proof of concept for an FLS-based algorithm measuring the quay wall line parameters* from which the second objective can be achieved, i.e. to *provide a proof of concept fusing FLS measurements with auxiliary sensors to enable drift-free localization on flat bottoms*.

Chapter 2: Operating conditions In this chapter, the operating conditions the proposed system will subjected to are described in details. These practical constraints defining the problem setting of the thesis can be divided in four main parts. First, the preference for acoustic imagery from the FLS is justified by its superior range compared to the poor visibility

of cameras in turbid waters. Nonetheless, the FLS still presents limitations, mainly from transmission loss on one hand, and noise disturbances on the other. Then, the operating conditions for quay wall tracking are laid out by describing the operator tasks and the general appearance of the quay wall line in sonar images. Thirdly, the operating conditions for localization describes the current method in more details and explains its weaknesses along with those of motion sensors. Lastly, the limitations causing the volume of data useful for implementation experiments to be restrained are described.

Chapter 3: Requirements This chapter provides a list of functional and non-functional requirements for both of the thesis objectives. The functional requirements define the desired outputs given a specific set of inputs i.e. the line parameters (ρ^S, θ^S) of the quay wall line feature from FLS images and the pose of the robot (x, y, ψ) from the quay wall tracker measurements and auxiliary sensors. The non-functional requirements set specifications for the subsystems' outputs in terms of performance. Since both systems are related, they are subjected to the same requirements for computation time $t_c < 0.040$ [s] but also accuracy $RMSE(\rho) < 0.125$ [m] and $RMSE(\theta) = 0.500$ [deg] and robustness score $ACC > 95\%$ measuring the proportion of outlier measurements. However, as a final requirement, the latter system which performs localization is subjected to the additional requirement to reduce drift $EBU(y) < 0.250$ [m] and $EBU(\psi) < 1.000$ [deg]. Together with the operating conditions, this chapter gives a clear view of the problem setting and the objectives.

Chapter 4: Quay-wall tracking This chapter goes through the conceptual design phase, the implementation and the validation of the first subsystem of the proposal: the quay wall tracker. Motivated by examples in the literature, it was decided to treat the FLS as a rangefinder for each beam. From there, the required subfunctions to achieve quay wall tracking while alleviating the weaknesses of the FLS were identified. The first, image segmentation, processes the FLS image through filtering and contrast enhancement to separate quay-wall pixels from the background inside a predefined Region of Interest (ROI). The second draws range measurements for the selected pixels and the last fits a line through the found scan points. For each subfunction the literature was investigated in order to select adequate methods, leading to three candidate solutions mainly distinguished by their line fitting algorithm: Total Least-Squares (TLS), Random Sample Consensus (RANSAC) and the Hough Transform (HT). Before validation, the system parameters were tuned and then their performance were compared where the TLS-based solution with Gaussian filtering showed the best results. Although it satisfied the computation time requirement with $t_c = 0.028 < 0.040$ [s], it was not accurate ($RMSE(\rho) = 0.183 < 0.125$ [m] and $RMSE(\theta) = 0.738 > 0.500$ [deg]) nor robust enough ($ACC = 84.0 < 95.0\%$) w.r.t. the requirements.

Chapter 5: Drift-free localization This chapter lays out the design and implementation of the filtering scheme fusing the quay wall tracker with auxiliary sensors in order to perform absolute localization on the FB. Models describing the robot motion from control inputs (speed and steering control inputs v and α) and ensuing sensory outputs (odometry and quay wall tracker) along with uncertainties from disturbances were constructed for state estimation, where control inputs and odometry uncertainties are assumed to be zero-mean Gaussian. Unlike the rest, a closed-form formula was used to define the uncertainty, accounting for the

dependency of distance errors ε_ρ with orientation errors ε_θ . Additionally, a conversion from local sonar to global ship referenced measurements was defined. These models serve as a basis for the implementation of filters for sensor fusion, for which three proven solutions were initially selected: the Extended Kalman Filter (EKF), the Unscented Kalman Filter (UKF) and the Particle Filter (PF). Based on literature, considerations in terms of computation time and accuracy concluded that the EKF offers the most computational simplicity while other solutions do not show significant gains for the given use case. Tuning of the filter was performed heuristically and validation tests have shown great improvements in terms of all performance metrics compared to the unfiltered system ($RMSE(\rho) = 0.116$ [m] and $RMSE(\theta) = 0.606$ [deg] and $ACC = 93.9\%$) while remaining under the processing time limit $t_c < 0.023$ [s]. These improvements were not enough to satisfy all performance requirements however, this did not keep from reducing error build-up below the required limits ($EBU(y) = 0.130 < 0.250$ [m] and $EBU(\psi) = 0.459$ [deg]).

6-2 Main conclusions

Quay wall tracking The first of objective of this thesis is to *provide a proof of concept for an FLS-based algorithm measuring the quay wall line parameters*. This objective was partially achieved in that the proposed system showed to be capable to track the quay wall line feature within the processing time limit with $t_c = 0.028 < 0.040$ [s], however without fulfilling the desired accuracy specifications ($RMSE(\rho) = 0.183 < 0.125$ [m] and $RMSE(\theta) = 0.738 > 0.500$ [deg]) nor the robustness score ($ACC = 84.0 < 95.0\%$). The successful part was attributed to the use of a ROI. On the other hand, it is believed that the choice of data sets for tuning, which was not representative of the validation set (recordings on old vs. new FLS device), played an important role in the insufficient accuracy. It was pointed however, that besides improving the tuning procedure, the performance can also be improved by the implementation of a filter.

Drift-free localization The second objective of this thesis is to *provide a proof of concept fusing FLS measurements with auxiliary sensors to enable drift-free localization on flat bottoms*. Formally speaking, the objective was only achieved partially. The accuracy of the quay wall referenced measurements was sufficient for normal distance ($RMSE(\rho) = 0.116 < 0.125$ [m]) but for not the orientation ($RMSE(\theta) = 0.606 > 0.500$ [deg]) nor the robustness ($ACC = 93.9 < 95.0\%$). The main factors are believed be the sub-optimal tuning of the underlying quay wall tracker but also the human error in manually defined ground truths, suggesting larger inaccuracies than in reality. Also, the weakness of the underlying odometry model predicting the heading is believed to degrade the performance of the system.

Overall All in all, this thesis contributed in reaching the goal to *develop a proof-of-concept for a sonar-based framework enabling drift-free localization of a ship-hull cleaning robot on flat bottoms*. With the first objective, a proof of concept for the front-end of the system extracting absolutely referenced measurements from FLS data was provided, as such augmenting the sensory capabilities of the robot. With the second objective, a proof of concept for the

back-end processing the new FLS sensor together with auxiliary sensors for drift-free localization was provided. Although requirements for computational speed and drift reduction were satisfied, the requirements for accuracy and robustness were not all fulfilled.

6-3 Recommendations

Although the goal of the thesis is not formally achieved, it is believed that the requirements that were not satisfied in this thesis could be accomplished by addressing the weaknesses the proposed systems. Suggestions will be discussed in Section 6-3-1 through 6-3-3. In light of Fleet Cleaner's vision for an autonomous robot in the future, recommendations for new research directions in that respect will be given.

6-3-1 Both systems

Ground truths For this thesis the irregularity of manually defined ground truths from human error was taken care of by taking the average of measurements and smoothing the resulting consecutive measurements with a Gaussian filter. In the future, statistical analysis of the uncertainty of human measurements should be performed, which could be used to optimize the ground truths and in turn the estimation of errors for tuning and testing systems.

More data and live implementation The amount of recorded data for this thesis was limited due to various practical issues listed in Section 2-4. As a result, the current system was tuned on a sub-optimal tuning set. In the future, the overall system (both the TLS-based line tracker and the EKF for localization) should be tuned on additional and more representative data for better performance. Moreover, the currently available data sets are only a limited sample of the possible circumstances. In the future, the proposed system should be implemented in real-time during actual FB operations in order to gain more insights, e.g. the robustness depending on the distance to the quay wall.

Quay wall detection Currently, the proposed system relies on a previous estimate and the assumption that the quay wall line feature is present for each incoming sonar image. However, the system is incapable to assess whether this assumption holds, nor the quality of the images. In the eventuality of the quay wall's disappearance from the image, the system would likely collapse and require human intervention. With Fleet Cleaner's vision for an autonomous robot in mind, a system capable to detect the quay wall line feature to launch or abort quay wall tracking by itself would be of great interest. Steensma showed that the Radon Transform (RT) presents good performance for detecting the bilge, also a linear feature, on images from the same FLS model.

Flat Bottom detection In Section 2-3, it was mentioned that ensuring that the robot is situated on the Flat Bottom of the ship is among the tasks of operators, next to ensuring the presence of the quay wall in sonar images. Obviously, it is essential to perform this assessment correctly since the proposed system relies on the assumption of two-dimensional

planar motion, which would justify the interest in its automation. Indeed, along with the quay wall detector recommended in the previous chapter, this system could ensure that the quay wall tracker is only launched once the assumptions upon which it rests are valid, as such creating a more robust autonomous system.

6-3-2 Quay wall tracking

Weighted Least-Squares The current TLS-based algorithm for line fitting "trusts" all measurements equally, regardless of their spatial provenance. However, inspection of images suggested the decrease of reliability for increasing range measurements. Careful analysis of the residuals of the range measurements with respect to the ground truth measurements could provide insights with regards to uncertainties of scan points. That is, weights depending on the range could be assigned for each scan point e.g. putting more "trust" in closer measurements. Then, the best line fit could be found through a weighted formulation of the TLS, called Weighted Least-Squares (WLS) [74, 75].

6-3-3 Drift-free localization

Quay wall offset measurement The proposed system uses a prior pose estimate delivered by the operator to define the offset between quay wall frame and ship frame, which is in turn used to reference quay wall line measurements to the ship frame. Naturally, the accuracy of this offset depends on the operator input, which is unfortunately prone to errors. For that, it would be of interest to find method to accurately measure this offset from FLS measurements. For example, this could be done by using Steensma's bilge detector [18] to locate the bilge in combination with the quay wall tracker to locate the quay wall. With known ship dimensions, the offset can be determined.

Sensor selection Next to the FLS, only the encoders were considered (measuring speed v and steering angle α) since other sensors suffer from heavy disturbances. While in practice it is considered by Fleet Cleaner to predict the robot's forward motion sufficiently accurately, the prediction of the robot's heading change is particularly poor due to slip of the front wheel. To alleviate this problem, future work should consider using auxiliary sensors. For example, a gyro included in the Inertial Measurement Unit (IMU) is capable to measure this change directly instead of being predicted from noisy odometry measurements. As such, more accurate predictions would further improve the accuracy of the EKF-based estimations of the robot poses. One might even consider combining the two, a method called "gyrodometry" [91].

Path planning and control The rather sinuous nature of trajectories (due to human error) instead of perfectly straight ones are believed to cause the invalidation of linear motion assumptions, in turn degrading the EKF's accuracy. To eliminate irregular operator control inputs, implementing a path planning and control scheme capable to follow a more linear path would benefit the accuracy performance of the EKF. With the quay wall as a reference, this would amount to solving the wall-following problem [90].

6-3-4 Line-based Simultaneous Localization And Mapping (SLAM)

In this thesis, the quay wall was the only reference used for localization. However, the environment of the FB knows more features, such as the bilge (keels), fouling or weld lines which form a grid of lines. Just as the quay wall, weld lines appear as linear features (either parallel or perpendicular to the quay wall), although not as thick and bright. Nonetheless, these linear features represent useful longitudinal references in practice. Therefore, research in a system tailored to the detection of weld lines, where knowledge of heading ψ could help to refine the search space. Based on the detector, a line-based SLAM scheme such as in [41] could be proposed in order to create a map of weld lines from which drift in the x -direction could be reduced. As such, a completely referenced localization scheme would be created.

6-3-5 Trajectory optimization

The proposed localization system uses a filter, i.e. the EKF, to fuse sensor measurements together in order to improve the accuracy of state estimates. In essence, the EKF only considers current measurements and the previous state estimate to optimize the new estimate, instead of the entire trajectory of previous states and corresponding measurements. In the literature survey prior to this thesis, it was found that trajectory optimization is a commonly used paradigm to improve the localization of Hovering Autonomous Underwater Vehicles (HAUVs) around ship hulls [32, 33]. For future research, it would be interesting to investigate the potential benefits from the known solutions. For example, in [51] the trajectory is optimized by augmenting the state of the Kalman filter to include the last ten states. Another more widely used method in the ship survey setting is the pose-graph formulation [27, 28, 30, 32], where measurements between states are used as constraints for finding the trajectory best fitting the sensory data [92]. This method knows an incremental equivalent, which has also proved its worth in the same setting [93, 94].

Appendix A

Sensory systems

This chapter in the appendix gives an overview of the relevant sensory systems on the Fleet Cleaner robot that were used in this thesis: the Forward Looking Sonar (FLS) and odometry sensors, comprising of the wheel encoders and the steering angle encoder.

A-1 Forward Looking Sonar

Specifications The acoustic imagery used in this thesis are gathered by BlueView M-900-2250 FLS for which table A-1 gives the specifications. Recall that the high-range/low-resolution (900 kHz) mode of operation was chosen.

Property	Value
Field-of-View	130°
Max Range (900/2250)	100 m/10 m
Optimal Range (900/2250)	2 - 60 m/0.5 - 7 m
Beam Width (900/2250)	1°×20°/1°×20°
Beam Spacing	0.18°
Number of Beams	768
Range Resolution (900/2250)	1.3 cm / 0.6 cm
Update Rate*	Up to 25 Hz
Frequency	900 kHz - 2250 kHz

Table A-1: Specifications of FLS used by the Fleet Cleaner robot.

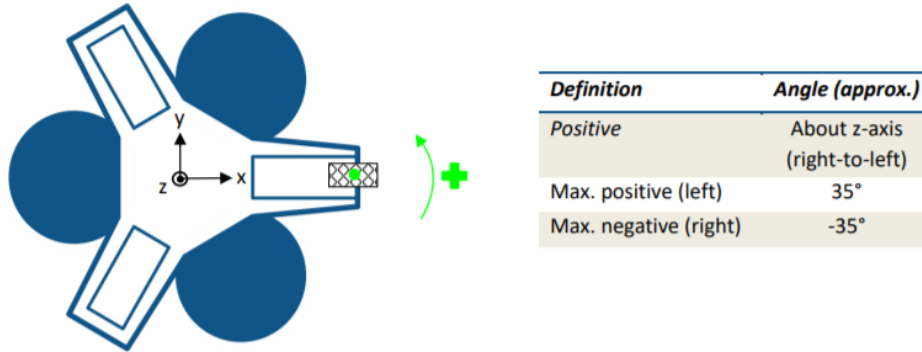
Dimensions The relevant robot dimensions that were used to translate the different frames of reference used in this thesis are listed in Table A-2 below:

Table A-2: Overview of the relevant robot dimensions.

Parameter	Value [m]	Unit	Description
l_{RS}	1.141	[m]	Robot origin to sonar (x -direction)
l_{ra}	0.388	[m]	Robot origin to rear axle ($-x$ -direction)
l	1.163	[m]	Wheel base

A-2 Odometry

Steering angle encoder Fleet Cleaner uses an electromagnetic absolute encoder to measure the angle of the steering wheel. It has a resolution of 4096 steps per full rotation, corresponding to steps of 0.088 [deg] [19]. Furthermore, the maximum steering angles the robot is capable to apply is $\alpha_{max} = \pm 35$ [deg], as illustrated in the figure below.

**Figure A-1:** Schematic of the steering angle encoder.

Wheel encoders The Fleet Cleaner robot has RLI200 bearingless wheel encoders installed on each of the three wheels [19]. With known wheel diameters $d_w = 0.30$ [m], the forward velocity assuming zero slip can be simply calculated with $v = d_w \cdot \omega$, where ω is the estimated rotational speed. Further specifications of the encoders is given in the table below:

Table A-3: Overview of specifications of the wheel encoders. The [ppr] unit stands for pulse per revolution.

Parameter	Value	Unit
Accuracy	0.3	[deg]
Pulse rate	2800	[ppr]
Input frequency	250	[kHz]

A-3 Recorded data

This section gives an overview of the recorded data from passed cleaning operations that was used in this thesis, namely FLS images and odometry data.

Forward Looking Sonar Table A-4 gives an overview of the characteristics of the recorded FLS data.

Table A-4: Overview of the data used for the tuning and validation tests for the quay wall detection system and the filtered state estimation. T is the total duration of the recording, Δt is the time step between frames and f is the corresponding frame rate, r_{max} is the max viewing range of the sonar image, dr is the range resolution, Q / S is the the quay wall position w.r.t. the sonar (L for left and R for right), Q / B is the the quay wall position w.r.t. the ship and ADS is the availability of navigational data.

Ship	Images	T [s]	Δt [s]	f [Hz]	r_{max} [m]	dr [m]	Q / S	Q / B	ADS
A1	942	73.31	0.078	12.82	14.00	0.0237	R	L	No
A2	724	54.81	0.076	13.16	14.00	0.0237	R	L	No
A3	723	67.72	0.093	10.75	18.00	0.0300	R	L	No
B1	810	130.51	0.161	6.21	32.40	0.0550	R / L	?	No
C1	7347	107.41	0.105	9.52	16.40	0.0270	R	?	No
D1	564	480.66	0.854	1.17	39.20	0.0663	R	L	Yes
D2	204	219.39	1.055	0.95	45.20	0.0763	L	L	Yes
D3	691	446.64	0.643	1.56	62.06	0.1050	R	L	Yes
E1	1068	389.26	0.363	2.75	34.80	0.0587	L	L	Yes
F1	1716	400.94	0.233	4.29	41.47	0.0701	R	R	No

Odometry The recorded data that was used in Chapter 5 for sensor fusion is presented in Figure A-2 and Figure A-3 (next pages). As shown in Table A-4, these were only available for data sets D1, D2, D3 and E1.

Odometry Data per Data Set

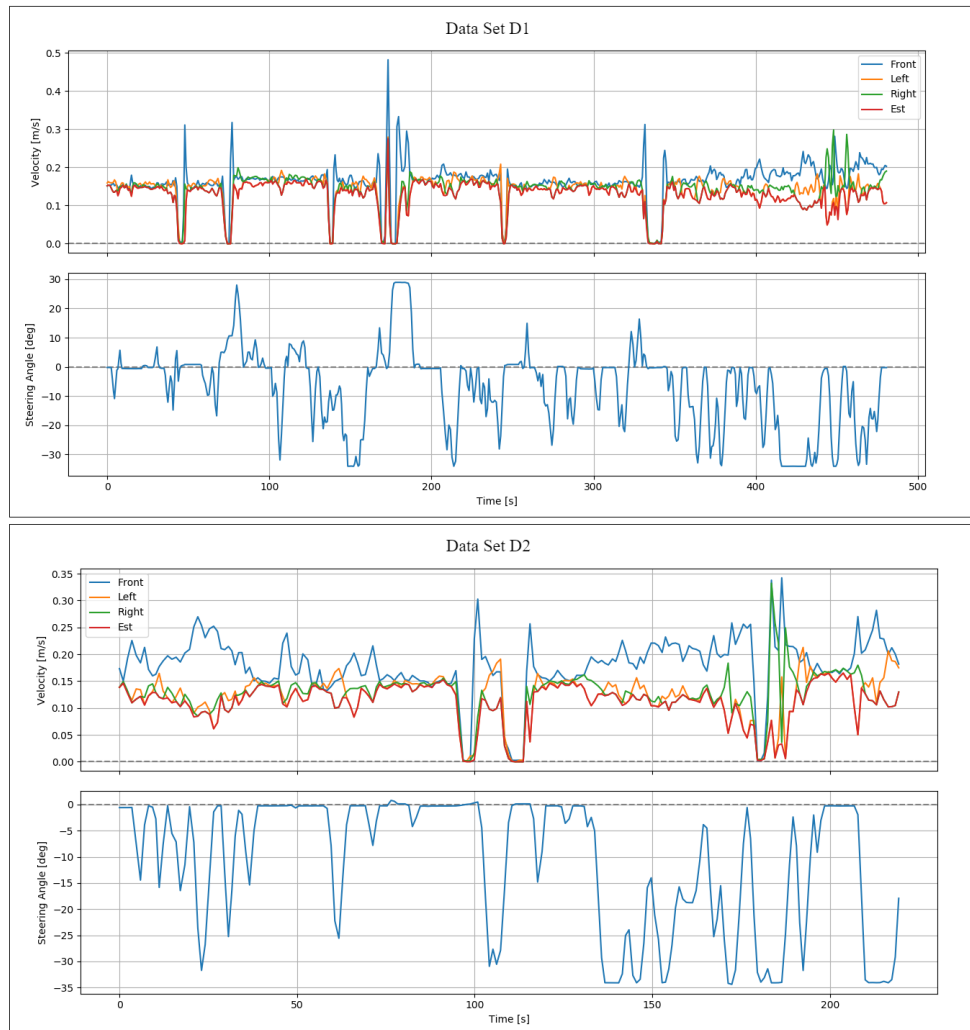


Figure A-2: Odometry data for data sets D3 and E1.

Odometry Data per Data Set

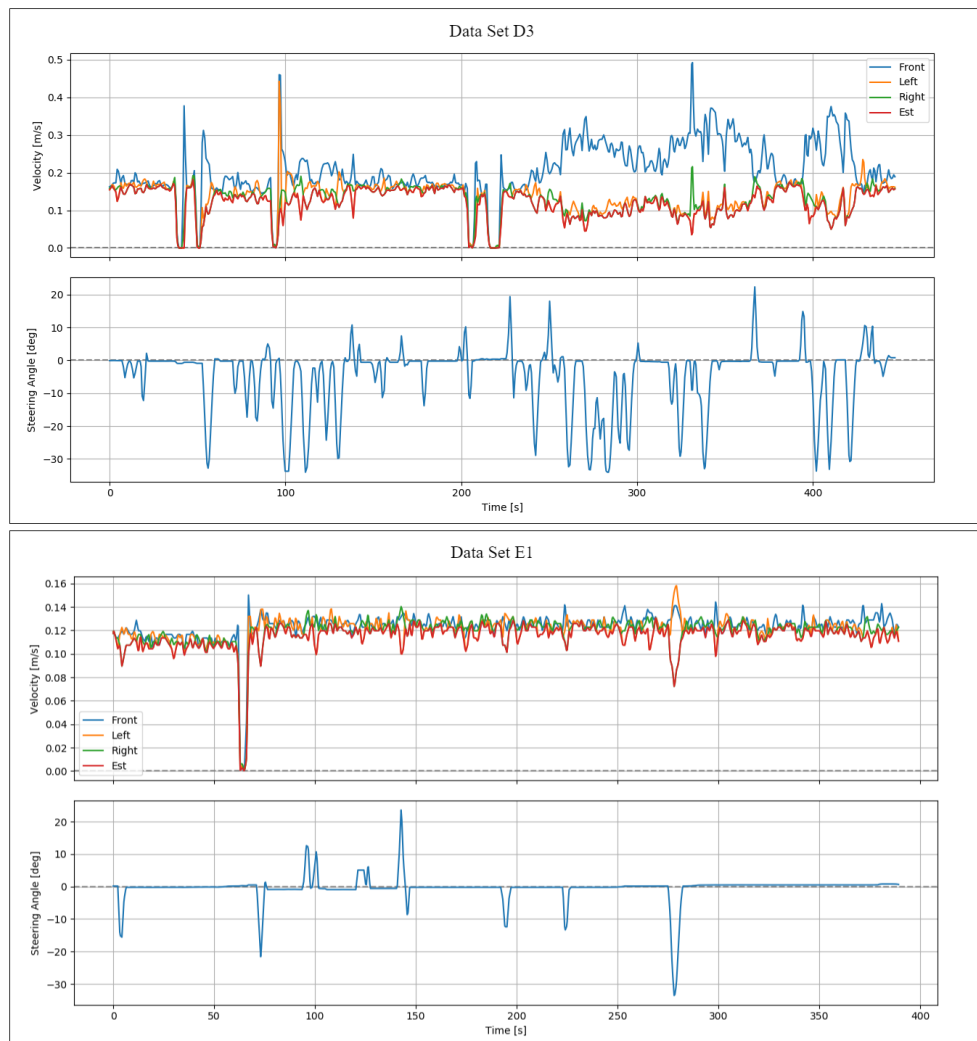


Figure A-3: Odometry data for data sets D3 and E1

Appendix B

Algorithms

This appendix gives the source code for the self-written algorithms which were used for implementation of the

B-1 Segmentation

Filtering For the implementation of filtering, the functions `MedianBlur()` and `GaussianBlur()` from OpenCV's library were used.

Contrast enhancement The method for enhancing contrast, Weng's filter [55], was implemented as follows.

```
1  C = 0.10
2  R = 1.7
3
4  def weng_filter(image):
5      """ Enhance contrast with Weng's method (2012) """
6      # Create white image and select minimum between white image and Weng transformed image
7      image_white = np.ones_like(image) * 255
8      return np.minimum(C * image ** R, image_white)
```

Thresholding The implementation of Otsu's method for threshold selection was performed using OpenCV's built-in function e.g. `cv2.threshold(image, 0, 255, cv2.THRESH_OTSU)`.

B-1-1 Range extraction

```
1  def range_measurements(image_segmented, roi_point_1, roi_point_2, polar):
2      """ This function gives the detection points of the line detector in sonar polar and cartesian
3          coordinates """
4      # Define beams to analyse
5      filtered_beams = filter_beams(image_segmented.copy(), roi_point_1[0], roi_point_2[0]) # 1D
6      # Get row and column indices for target hits
7      target_hits = []
8      points_rt = []
```

```

9     points_xy = []
10    for col in range(roi_point_1[0], roi_point_2[0]):
11        row = np.argmax(filtered_beams[:, col])
12        if row:
13            target_hits.append([row, col])
14            detection_point = col_row_image_to_sonar(col, row, polar)
15            r, t = col_row_to_polar(detection_point[0], detection_point[1], polar)
16            x, y = polar_to_cartesian(r, t)
17            points_rt.append([x, t])
18            points_xy.append([x, y])
19    points_rt = np.array(points_rt)
20    points_xy = np.array(points_xy)
21    return points_xy, points_rt

```

B-2 Line fitting

For the self written Python functions for line fitting, the inputs are the scan points (`points_xy`) from FLS range extraction and previous normal distance (`rho_prev`) and orientation measurements (`theta_prev`) of the quay wall line.

B-2-1 Total Least-Squares

```

1  import numpy as np
2  from math import *
3
4  def total_least_squares(points_xy):
5      """ Total Least Squares line fit inspired from Sommer (2018) """
6      # count number of points and mean positions
7      n = np.shape(points_xy)[0]
8      mean_x = np.mean(points_xy[:, 0])
9      mean_y = np.mean(points_xy[:, 1])
10
11     # compute covariances
12     c_xx = np.sum((points_xy[:, 0] - mean_x) ** 2) / n
13     c_yy = np.sum((points_xy[:, 1] - mean_y) ** 2) / n
14     c_xy = np.sum((points_xy[:, 0] - mean_x) * (points_xy[:, 1] - mean_y)) / n
15
16     # define line parameters
17     theta = 1 / 2 * atan2(-2 * c_xy, c_yy - c_xx)
18     rho = mean_x * cos(theta) + mean_y * sin(theta)
19     return rho, theta

```

B-2-2 RANdom SAMple Consensus

```

1  import numpy as np
2  from math import *
3  from random import sample
4
5  # Parameter Space
6  DELTA_RHO = 0.5 # [m]
7  DELTA_THETA = 2.5 # [deg]
8
9  def ransac(points_xy, rho_prev, theta_prev):
10     """ Line fitting with the RANdom Sampel Consensus algorithm. Based from Kovesi (2005) with
11         modifications. """
12     # define size of search space
13     theta_min = theta_prev - DELTA_THETA * pi / 180
14     theta_max = theta_prev + DELTA_THETA * pi / 180
15     rho_min = rho_prev - DELTA_RHO
16     rho_max = rho_prev + DELTA_RHO
17
18     # parameters
19     omega = 0.20
20     p = 0.99
21     error_tolerance = 0.25 # [m]
22     iterations = int((log(1 - p) / log(1 - omega ** 2)))
23
24     # other RANSAC specific parameters
25     inliers = np.array([])

```

```

25     number_of_points = np.shape(points_xy)[0]
26     min_number_of_inliers = 20
27     max_number_of_inliers = 0
28     ids = [i for i in range(number_of_points)]
29
30     # iterate to find largest set
31     for k in range(iterations):
32         # draw random sample of 2 points
33         sample_ids = sample(ids, 2)
34         sample_points = points_xy[sample_ids, :]
35
36         # unit vector from sample points
37         d = sample_points[0] - sample_points[1]
38         d = d / np.sqrt(np.sum(d ** 2))
39
40         # find angle and check if inside parameter space
41         theta = -np.arctan(d[0] / d[1])
42         if not theta_min < theta < theta_max:
43             continue
44
45         # find normal distance rho for candidate angle theta
46         rho = sample_points[0, 0] * cos(theta) + sample_points[0, 1] * sin(theta)
47         if not rho_min < rho < rho_max:
48             continue
49
50         # count the inliers
51         rho_i = points_xy[:, 0] * cos(theta) + points_xy[:, 1] * sin(theta)
52         d_i = abs(rho - rho_i)
53         inliers_ids = np.where(d_i < error_tolerance)[0]
54         number_of_inliers = len(inliers_ids)
55         if number_of_inliers > max_number_of_inliers and number_of_inliers > min_number_of_inliers:
56             max_number_of_inliers = number_of_inliers
57             inliers = inliers_ids
58
59     # determine line parameters on largest set
60     if max_number_of_inliers != 0:
61         # get covariance matrix
62         inliers_points = points_xy[inliers, :]
63         rho, theta = pca(inliers_points)
64     else:
65         rho = None
66         theta = None
67     return rho, theta

```

B-2-3 Hough Transform

```

1  import numpy as np
2  from math import *
3
4  # Parameter Space
5  DELTA_RHO = 0.5 # [m]
6  DELTA_THETA = 2.5 # [deg]
7
8  # Hough Transform Parameters
9  D_RHO = 0.10 # [m]
10 D_THETA = 0.150 # [deg]
11
12
13 def hough_transform(points_xy, rho_prev, theta_prev):
14     """ This function finds the line parameters by implementing the Hough transform.
15         Instead of using a binary image, it uses the list of detection points, described by
16         their coordinates (x, y) in the Cartesian space. """
17     # define bounds of Hough space
18     rho_lb = rho_prev - DELTA_RHO # lower bound
19     rho_ub = rho_prev + DELTA_RHO # upper bound
20     theta_lb = theta_prev * 180 / pi - DELTA_THETA # lower bound
21     theta_ub = theta_prev * 180 / pi + DELTA_THETA # upper bound
22
23     # define Hough space
24     rho_array = np.arange(rho_lb, rho_ub + D_RHO, D_RHO)
25     theta_array = np.arange(theta_lb, theta_ub + D_THETA, D_THETA)
26     number_of_rhos = len(rho_array)
27     number_of_thetas = len(theta_array)
28     accumulator = np.zeros((number_of_rhos + 2, number_of_thetas + 2))
29
30     # retain reusable values
31     cos_theta = np.cos(theta_array * pi / 180)
32     sin_theta = np.sin(theta_array * pi / 180)
33
34     # vote for line candidates
35     number_of_points = len(points_xy[:, 0])
36     for k in range(number_of_points):

```

```

37     for j in range(number_of_thetas):
38         # only vote if candidate line is inside parameter space
39         rho_candidate = points_xy[k, 0] * cos_theta[j] + points_xy[k, 1] * sin_theta[j]
40         if np.min(rho_array) < rho_candidate < np.max(rho_array):
41             i = int(round(((rho_candidate - rho_lb) / D_RHO)))
42             accumulator[i, j] += 1
43
44     # find index of line with maximum votes
45     idx_max = np.unravel_index(np.argmax(accumulator, axis=None), accumulator.shape)
46     rho, theta = rho_array[idx_max[0]], theta_array[idx_max[1]]
47     return rho, theta * pi / 180

```

B-3 Extended Kalman Filter

```

1  def f(x, u, dt):
2      """ Motion model of Fleet Cleaner robot """
3      return np.array([
4          x[0] + u[0] * cos(x[2]) * dt,
5          x[1] + u[0] * sin(x[2]) * dt,
6          x[2] + u[0] * sin(u[1]) / WHEEL_BASE * dt
7      ])
8
9  def F(x, u, dt):
10     """ Jacobian of f(x, u) => df/dx|(x, u) """
11     # first row
12     f13 = - u[0] * sin(x[2]) * dt
13     f23 = u[1] * cos(x[2]) * dt
14
15     return np.array([[1, 0, f13],
16                     [0, 1, f23],
17                     [0, 0, 1]])
18
19  def H(x, u, dt):
20     """ Jacobian of h(x, u) => dh/dx|(x, u) """
21     return np.array([[0, 1, 0],
22                     [0, 0, 1],
23                     [0, 0, 0],
24                     [0, 0, 0]])
25
26  def L(x, u, dt):
27     """ Jacobian of f(x, u) => df/dw|(x, u) """
28     return np.array([[cos(x[2]) * dt, 0],
29                     [sin(x[2]) * dt, 0],
30                     [sin(u[1]) / WHEEL_BASE * dt, u[0] * cos(u[1]) / WHEEL_BASE * dt]])
31
32  def extended_kalman_filter(x_prev, P_prev, R_rt, z, dt):
33     """ Extended Kalman Filter
34         R_rt is the covariance calculate from detection as described in Sommer (2018),
35         z are the measurements and dt the time step """
36     # Measurement noise covariance matrix R (variances)
37     R_v = 0.012 # Standard deviation of odometer (velocity)
38     R_a = 0.046 * pi / 180 # Standard deviation of steering encoder (steering angle)
39
40     # Control input noise
41     Q_v = 0.001 # Standard deviation of speed input (velocity)
42     Q_a = 11.3 * pi / 180 # Standard deviation of steering input (steering angle)
43
44     R = np.array([[R_rt[0, 0], R_rt[0, 1], 0, 0],
45                 [R_rt[0, 1], R_rt[1, 1], 0, 0],
46                 [0, 0, R_v ** 2, 0],
47                 [0, 0, 0, R_a ** 2]])
48
49     """ Predict """
50     # Predicted state estimate
51     u = np.array([z[2], z[3]])
52     w = np.array([0, 0])
53     x_pred = f(x_prev, u, dt)
54
55     # Predicted covariance estimate
56     Fc = F(x_prev, u, dt)
57     Lc = L(x_prev, u, dt)
58     Pu = np.diag([Q_v ** 2, Q_a ** 2])
59     P_pred = Fc @ P_prev @ Fc.T + Lc @ Pu @ Lc.T
60
61     """ Update """
62     # Innovation or measurement residual
63     v = np.array([0, 0, 0, 0])
64     y = z - h(x_pred, u, v, dt)
65
66     # Innovation (or residual) covariance and near-optimal Kalman gain
67     Hc = H(x_pred, u, dt)

```



```
68     S = Hc @ P_pred @ Hc.T + R
69     K = P_pred @ Hc.T @ np.linalg.inv(S)
70
71     # Updated state and covariance estimate
72     x_est = x_pred + y @ K.T
73     P_est = P_pred - K @ Hc @ P_pred
74     return x_est, P_est
```

Appendix C

General Arrangement

As mentioned in the introduction Chapter 1, General Arrangements (GAs) give an overview of the main decks of the ship but not the actual Flat Bottom (FB). In general, the tanktop is presented instead, i.e. the plating at the inner bottom of the ship (as shown in Figure C-1).

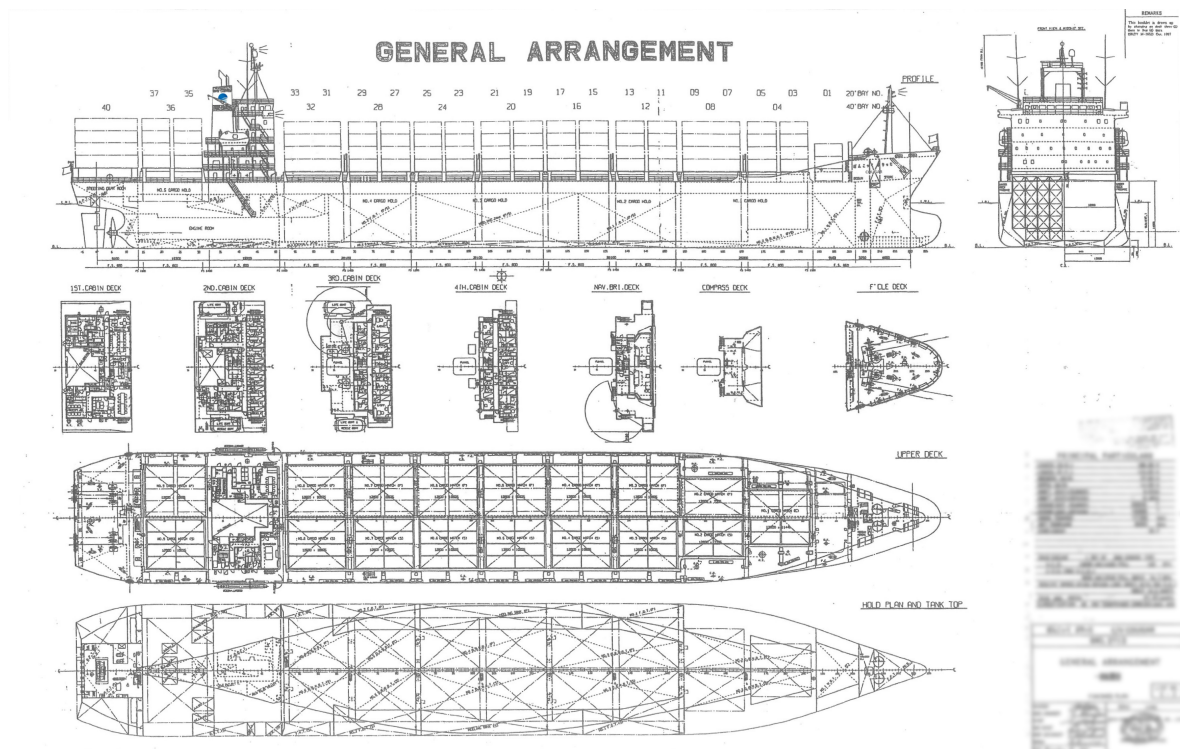


Figure C-1: Example of a standard General Arrangement, presenting a side view of the ship along with an upper view of its main decks. The best indication if the FB is found in the last row, generally depicting the tank top (i.e. the plating at the inner bottom of the ship) instead of the true FB. Moreover, these documents specify the main dimensions such as the ship length and breadth are given (these were blurred for confidentiality purposes).

Bibliography

- [1] M. A. F. Lewthwaite J. C. and K. W. Thomas, “An Investigation into the Variation of Ship Skin Frictional Resistance with Fouling,” in Transactions of Royal Institution of Naval Architects, vol. 127, pp. 269—284, 1985.
- [2] R. L. Townsin, “The Ship Hull Fouling Penalty,” Biofouling, vol. 19, no. sup1, pp. 9–15, 2003.
- [3] L. Montelli and J. Lewis, “Survey of Biofouling on Australian Navy Ships: Crustacea; Isopoda and Amphipoda; Caprellidea,” tech. rep., 2008.
- [4] E. C. Tupper, “Definition and Regulation,” in Introduction to Naval Architecture, pp. 9–32, Butterworth-Heinemann, 5 ed., jan 2013.
- [5] T. van Bergen, “Sonar-based Absolute Localization for Ship Hull Cleaning Robots: A Survey,” tech. rep., Delft University of Technology, Delft, 2019.
- [6] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics. MIT Press, 1999.
- [7] J. Borenstein, H. R. Everett, L. Feng, S. W. Lee, R. H. Byrne, J. Borenstein, and L. Feng, “Where am I? Sensors and Methods for Mobile Robot Positioning,” tech. rep., 1996.
- [8] J. Borenstein, H. R. Everett, L. Feng, and D. Wehe, “Mobile Robot Positioning: Sensors and Techniques,” Journal of Robotic Systems, vol. 14, no. 4, pp. 231–249, 1997.
- [9] T. Yaqub, Mobile Robot Motion, Perception and Environment Modelling. Phd, University of New South Wales, 2008.
- [10] R. Siegwart and I. R. Nourbakhsh, “Introduction to Autonomous Mobile Robots,” vol. 21, pp. 1–336, Cambridge: MIT Press, 2004.
- [11] J. Borenstein, “Experimental Results from Internal Odometry Error Correction with the OmniMate Mobile Robot 1,” Tech. Rep. 6, 1998.

- [12] 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, no. 2, pp. 1–153, 2017.
- [13] L. Stutters, H. Liu, C. Tiltman, and D. J. Brown, “Navigation Technologies for Autonomous Underwater Vehicles,” IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, vol. 38, no. 4, pp. 581–589, 2008.
- [14] H. Durrant-Whyte and T. C. Henderson, “Multisensor Data Fusion,” in Springer Handbook of Robotics Multisensor D 25. Multisensor Data Fusion (K. Siciliano, ed.), ch. 25, pp. 585–609, 2008.
- [15] D. Borota, “Design of a Position Determination System for a Ship’s Hull Maintenance Robot,” 2014.
- [16] W. J. L. Weitsma, “Design of an Above Water Position Determination System for a Ship’s Hull Maintenance Robot,” 2016.
- [17] K. Cassee, “Wheel Slip and Orientation Drift Correction for the Relative Localization System of a Ship Hull Cleaning Robot,” 2018.
- [18] B. Steensma, “Ship Hull Cleaning Robots: Obstacle Detection Using a Forward Looking Sonar System,” 2018.
- [19] D. Verstrate, “Framework Design and Image Registration for Sonar-Based Underwater SLAM,” 2018.
- [20] Y. Petillot, I. T. Ruiz, and D. Lane, “AUV Navigation using a Forward Looking Sonar,” in Proceedings from Oceanology International, 2000.
- [21] Y. Petillot, I. T. Ruiz, and D. M. Lane, “Underwater Vehicle Obstacle Avoidance and Path Planning Using a Multi-Beam Forward Looking Sonar,” IEEE Journal of Oceanic Engineering, vol. 26, no. 2, pp. 240–251, 2001.
- [22] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” Transactions of the ASME-Journal of Basic Engineering, vol. 82, pp. 35–45, 1960.
- [23] S. Y. Chen, “Kalman Filter for Robot Vision: A Survey,” IEEE Transactions on Industrial Electronics, vol. 59, no. 11, pp. 4409–4420, 2012.
- [24] H. Durrant-Whyte and T. Bailey, “Simultaneous Localization and Mapping (SLAM): Part I The Essential Algorithms,” IEEE Robotics and Automation Magazine, vol. 13, no. 2, pp. 99–110, 2006.
- [25] F. Guth, L. Silveira, S. Botelho, P. Drews, and P. Ballester, “Underwater SLAM: Challenges, State of the Art, Algorithms and a new Biologically-inspired Approach,” in 5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics, (São Paulo), pp. 981–986, IEEE, 2014.
- [26] T. Bailey and H. Durrant-Whyte, “Simultaneous localisation and mapping (slam): Part ii state of the art,” Robotics and Autonomous Systems, vol. 13, no. 3, pp. 108–117, 2006.

- [27] H. Johannsson, M. Kaess, B. Englot, F. Hover, and J. Leonard, "Imaging Sonar-aided Navigation for Autonomous Underwater Harbor Surveillance," in IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings, (Taipei), pp. 4396–4403, IEEE, 2010.
- [28] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. Leonard, "Advanced Perception, Navigation and Planning for Autonomous In-water Ship Hull Inspection," Tech. Rep. 12, 2012.
- [29] M. Walter, F. Hover, and J. Leonard, "SLAM for Ship Hull Inspection using Exactly Sparse Extended Information Filters," in IEEE International Conference Robotics and Automation, (Pasadena), pp. 1463–1470, Piscataway: IEEE Press, 2008.
- [30] J. Li, M. Kaess, R. M. Eustice, and M. Johnson-Roberson, "Pose-graph SLAM using Forward-looking Sonar," tech. rep., 2018.
- [31] P. Ozog, G. Troni, M. Kaess, R. M. Eustice, and M. Johnson-roberson, "Building 3D Mosaics from an Autonomous Underwater Vehicle , Doppler Velocity Log , and 2D Imaging Sonar," in 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 1137–1143, IEEE, 2015.
- [32] P. Ozog, N. Carlevaris-Bianco, A. Kim, and R. M. Eustice, "Long-Term Mapping Techniques for Ship Hull Inspection and Surveillance Using an Autonomous Underwater Vehicle," Field Robotics, vol. 33, pp. 265–289, 2016.
- [33] J. Li, "Place Recognition and Localization for Multi-Modal Underwater Navigation with Vision and Acoustic Sensors," ProQuest Dissertations and Theses, p. 141, 2017.
- [34] D. Ribas, P. Ridao, J. D. Tardós, and J. Neira, "Underwater SLAM in Man-Made Structured Environments," Journal of Field Robotics, vol. 25, no. 11/12, pp. 898–921, 2008.
- [35] A. Mallios, P. Ridao, D. Ribas, F. Maurelli, and Y. Petillot, "EKF-SLAM for AUV navigation under Probabilistic Sonar Scan-Matching," in The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, (Taipei), pp. 4404–4411, 2010.
- [36] Roy Edgar Hansen, "Introduction to Sonar," 2012.
- [37] N. Hurtós, D. Ribas, X. Cufí, Y. Petillot, and J. Salvi, "Fourier-Based Registration for Robust Forward-looking Sonar Mosaicing in Low-visibility Underwater Environments," Journal of Field Robotics, vol. 32, no. 1, pp. 123–151, 2015.
- [38] Teledyne, "BlueView M900-2250 Dual Frequency Series."
- [39] S. Karabchevsky, Real-Time Underwater Obstacle Detection using Forward Looking Sonar and FPGA. Msc, 2011.
- [40] K. Teo, K. W. Ong, and H. C. Lai, "Obstacle detection, avoidance and anti collision for MEREDITH AUV," MTS/IEEE Biloxi - Marine Technology for Our Future: Global and Local Challenges, OCEANS 2009, pp. 1–10, 2009.
- [41] S. T. Pfister, Algorithms for Mobile Robot Localization and Mapping, Incorporating Detailed Noise Modeling and Multi-Scale Feature Extraction. PhD thesis, California Institute of Technology, 2006.

- [42] X. R. Li and Z. Zhao, "Measures of Performance for Evaluation of Estimators and Filters," pp. 4473–61.
- [43] P. Goel, S. I. Roumeliotis, and G. S. Sukhatme, "Robot Localization Using Relative and Absolute Position Estimates," in Proceedings of International Conference of Intelligent Robots and Systems, pp. 1134–1140, 1999.
- [44] F. Lu and E. Milios, "Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 935–938, 1994.
- [45] Y.-H. Choi, T.-K. Lee, and S.-Y. Oh, "A line feature based SLAM with low grade range sensors using geometric constraints and active exploration for mobile robot," Autonomous Robots, vol. 24, pp. 13–27, jan 2008.
- [46] B. Schiele and J. L. Crowley, "A Comparison of Position Estimation Techniques using Occupancy Grids," tech. rep., 1994.
- [47] J. L. Crowley, "World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging," in IEEE Conference on Robotics and Automation (ICRA 89), vol. 89, (Scottsdale Az., USA), pp. 1574–1579, 1989.
- [48] D. Ribas, P. Ridao, J. Neira, and J. D. Tardós, "A Method for Extracting Lines and their Uncertainty from Acoustic Underwater Images for SLAM," IFAC Proceedings Volumes, vol. 40, no. 15, pp. 61–66, 2007.
- [49] W. Kazmi, P. Ridao, D. Ribas, and E. Hernández, "Dam wall detection and tracking using a mechanically scanned imaging sonar," Proceedings - IEEE International Conference on Robotics and Automation, no. May, pp. 3595–3600, 2009.
- [50] A. Burguera, G. Oliver, and Y. González, "Range Extraction from Underwater Imaging Sonar Data," Proceedings of the 15th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2010, no. June 2014, 2010.
- [51] P. Ridao, M. Carreras, D. Ribas, and R. Garcia, "Visual Inspection of Hydroelectric Dams using an Autonomous Underwater Vehicle," Journal of Field Robotics, vol. 27, no. 6, pp. 759–778, 2010.
- [52] G. C. Karras, C. P. Bechlioulis, H. K. Abdella, T. Larkworthy, K. Kyriakopoulos, and D. Lane, "A Robust Sonar Servo Control Scheme for Wall-following using an Autonomous Underwater Vehicle," in IEEE International Conference on Intelligent Robots and Systems, no. November 2013, pp. 3893–3898, 2013.
- [53] J. Jaybhay and R. Shastri, "A Study of Speckle Noise Reduction Filters," An International Journal (SIPIJ), vol. 6, no. 3, 2015.
- [54] B. Steensma, "Ship Hull Maintenance Robots: a Survey on Underwater Object Detection Using a Forward Looking Sonar System," tech. rep., Delft University of Technology, Delft, 2017.

- [55] L. Weng, M. Li, and Z. Gong, "On Sonar Image Processing Techniques for Detection and Localization of Underwater Objects," Applied Mechanics and Materials, vol. 236-237, pp. 509–514, 2012.
- [56] M. J. Chantier and J. P. Stoner, "Automatic interpretation of sonar image sequences using temporal feature measures," IEEE Journal of Oceanic Engineering, vol. 22, no. 1, pp. 47–56, 1997.
- [57] J.-S. Lee, "Digital image enhancement and noise filtering by use of local statistics," IEEE Trans. Pattern Anal. Mach. Intell., vol. 2, p. 165–168, Feb. 1980.
- [58] V. S. Frost, J. A. Stiles, K. S. Shanmugan, and J. C. Holtzman, "A model for radar images and its application to adaptive digital filtering of multiplicative noise," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-4, pp. 157–166, March 1982.
- [59] D. T. Kuan, A. A. Sawchuk, T. C. Strand, and P. Chavel, "Adaptive noise smoothing filter for images with signal-dependent noise," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7, pp. 165–177, March 1985.
- [60] H.-N. Ho, J.-J. Lee, C. Park, and B.-W. Jo, "An Efficient Image Enhancement Algorithm for Sonar Images," International Journal of Latest Research in Science and Technology, vol. 2, no. 6, pp. 38–43, 2013.
- [61] G. Maragatham and S. M. Mansoor Roomi, "A Review of Image Contrast Enhancement Methods and Techniques," Research Journal of Applied Sciences, Engineering and Technology, vol. 9, no. 5, pp. 309–326, 2015.
- [62] M. D. Aykin and S. Negahdaripour, "On feature extraction and region matching for forward scan sonar imaging," OCEANS 2012 MTS/IEEE: Harnessing the Power of the Ocean, pp. 1–9, 2012.
- [63] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-9, no. 1, pp. 62–66, 1979.
- [64] E. Galceran, V. Djapic, M. Carreras, and D. P. Williams, "A Real-time Underwater Object Detection Algorithm for Multi-beam Forward Looking Sonar," IFAC Proceedings Volumes, vol. 45, no. 5, pp. 306–311, 2012.
- [65] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart, "A Comparison of Line Extraction Algorithms using 2D Laser Rangefinder for Indoor Mobile Robotics," in Intelligent Robots and Systems, pp. 1–6, 2005.
- [66] P. O. V. Hough, "Methods and Means for Recognizing Complex Patterns," 1962.
- [67] J. Illingworth and J. Kittler, "A Survey of the Hough Transform," Computer Vision, Graphics, and Image Processing, vol. 44, pp. 87–116, 1988.
- [68] D. Ribas, J. Neira, P. Ridao, and J. D. Tardós, "AUV Localization in Structured Underwater Environments Using an A-Priori Map," in 7th IFAC Conference on Maneuvering and Control of Marine Crafts, (Lisbon, Portugal), 2006.

- [69] L. Banjanovic, I. Petrovic, and E. Ivanjko, "Sonar-Based Self-Orientation of Indoor Mobile Robots," in 13th International Conference on Electrical Drives and Power Electronics, no. December 2013, 2005.
- [70] W. Wojcikiewicz, "Hough Transform: Coursework for Image Processing," 2008.
- [71] H. Li, M. A. Lavin, and R. J. Le Master, "Fast Hough transform: A hierarchical approach," Computer Vision, Graphics, and Image Processing, vol. 36, pp. 139–161, nov 1986.
- [72] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," Communications of the ACM, vol. 24, no. 6, pp. 381–395, 1981.
- [73] H. A. Kather, A. S. Gad, E. A. Omran, and A. A. Abdel-Fattah, "Enhancement Matching Algorithms Using Fusion of Multiple Similarity Metrics for Sonar Images," World Applied Sciences, vol. 6, no. 6, pp. 759–763, 2009.
- [74] S. T. Pfister, S. I. Roumeliotis, and J. W. Burdick, "Weighted Line Fitting Algorithms for Mobile Robot Map Building and Efficient Data Representation,"
- [75] V. Sommer, "A Closed-Form Error Model of Straight Lines for Improved Data Association and Sensor Fusing,," Sensors (Basel, Switzerland), vol. 18, apr 2018.
- [76] M. Kam, X. Zhu, and P. Kalata, "Sensor Fusion for Mobile Robot Navigation," Proceedings of the IEEE, vol. 85, no. 1, pp. 108–119, 1997.
- [77] L. D'Alfonso, W. Lucia, P. Muraca, and P. Pugliese, "Mobile robot localization via EKF and UKF: A comparison based on real data," Robotics and Autonomous Systems, 2015.
- [78] D. Simon, Optimal State Estimation. Hoboken, NJ: John Wiley & Sons, 2006.
- [79] K. O. Arras and R. Y. Siegwart, "Feature Extraction and Scene Interpretation for Map-based Navigation and Map Building," Mobile Robots XII, vol. 3210, no. January 1998, pp. 42–53, 1998.
- [80] S. Konatowski, P. Kaniewski, and J. Matuszewski, "Comparison of Estimation Accuracy of EKF, UKF and PF Filters," Annual of Navigation, vol. 23, no. 1, pp. 69–87, 2017.
- [81] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," tech. rep., University of North Carolina, Chapel Hill.
- [82] E. A. Wan and R. Van Der Merwe, "The Unscented Kalman Filter for Nonlinear Estimation," tech. rep.
- [83] N. Gordon, D. Salmond, and A. Smith, "Novel Approach to Non-linear/non-Gaussian State Estimation," IEE Proceedings F (Radar and Signal Processing), vol. 140, no. 2, pp. 107–113, 1993.
- [84] G. G. Rigatos, "Extended Kalman and Particle Filtering for sensor fusion in motion control of mobile robots," Mathematics and Computers in Simulation, vol. 81, no. 3, pp. 590–607, 2010.

- [85] L. Jetto, S. Longhi, and D. Vitali, "Localization of a wheeled mobile robot by sensor data fusion based on a fuzzy logic adapted Kalman filter," Control Engineering Practice, vol. 7, no. 6, pp. 763–771, 1999.
- [86] J. Sasiadek and P. Hartana, "Odometry and Sonar Data Fusion for Mobile Robot Navigation," IFAC Proceedings Volumes, vol. 33, pp. 411–416, sep 2000.
- [87] K. Cassee, "Sensor fusion and localization for the positioning system of a ship hull cleaning robot," tech. rep., Delft University of Technology.
- [88] S. Godsill, "Particle Filtering: The First 25 Years and Beyond Simon Godsill," Signal Processing and Communications (SigProC), pp. 7760–7764, 2019.
- [89] T. Li, S. Sun, T. P. Sattar, and J. M. Corchado, "Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches," 2014.
- [90] A. Bemporad, M. D. Marco, and A. Tesi, "Wall-Following Controllers for Sonar-Based Mobile Robots," in Proceedings of the 36th IEEE Conference on Decision and Control, pp. 1–6, 1997.
- [91] J. Borenstein and L. Feng, "Gyrodometry: A New Method for Combining Data from Gyros and Odometry in Mobile Robots," tech. rep., 1996.
- [92] L. Carlone, R. Aragues, J. A. Castellanos, and B. Bona, "A Fast and Accurate Approximation for Planar Pose Graph Optimization," tech. rep.
- [93] M. Kaess, "iSAM: Incremental Smoothing and Mapping," IEEE Transactions on Robotics, vol. 7, pp. 1–14, 2008.
- [94] M. Kaess, H. Johannsson, B. Englot, F. Hover, and J. Leonard, "Towards Autonomous Ship Hull Inspection using the Bluefin HAUV," Symposium on Technology and the Mine Problem, 2010.

Glossary

List of Acronyms

AWGN	Additive White Gaussian Noise
CLAHE	Contrast Limited Adaptive Histogram Equalization
EKF	Extended Kalman Filter
FB	Flat Bottom
FHT	Fast Hough Transform
FLS	Forward Looking Sonar
FOV	Field of View
GA	General Arrangement
HAUV	Hovering Autonomous Underwater Vehicle
HE	Histogram Equalization
HT	Hough Transform
IMU	Inertial Measurement Unit
ML	Maximum Likelihood
PCA	Principal Component Analysis
PSNR	Peak Signal to Noise Ratio
PF	Particle Filter
RANSAC	RANdom SAMple Consensus
ROI	Region of Interest
RMSE	Root-Mean-Square Error
ROV	Remotely Operated Vehicle
RT	Radon Transform
SLAM	Simultaneous Localization And Mapping
SNR	Signal to Noise Ratio

ToF	Time of Flight
TLS	Total Least-Squares
TVG	Time Varying Gain
UHP	Ultra High Pressure
UKF	Unscented Kalman Filter
VS	Vertical Side
WLS	Weighted Least-Squares