CONFIDENTIAL

Framework Design and Image Registration for Sonar-Based Underwater SLAM

Dave Verstrate

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

CONFIDENTIAL

Framework Design and Image Registration for Sonar-Based Underwater SLAM

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Dave Verstrate

September 24, 2018

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

Fleet Cleaner

The work in this thesis was supported by Fleet Cleaner.

TU Delft Delft University of Technology

DCSC

Abstract

The startup company Fleet Cleaner has developed a mobile robot, specialized in the hull cleaning of large cargo vessels. Navigation and localization of this robot is currently performed manually. This is a difficult process that is greatly complicated during operation. This is mainly due to the availability of relative positioning sensors only, which are prone to error build-up and noise, and to the difficulty of interpreting optical underwater images in turbid water conditions. Instead, operators must rely on acoustic images from a forward-looking sonar. In the field of mobile robotics, Simultaneous Localization and Mapping (SLAM) is an often used technique to improve navigation and localization by utilizing visual information. The objective of this thesis is to develop a sonar-based SLAM framework, tailored to working environment of the Fleet Cleaner robot. The thesis scope has been restricted to the conceptual design of such a framework and the implementation of one of the subsystems, visual odometry.

A conceptual design of a SLAM system is proposed using a systematic approach. Different working principles are evaluated according to operating conditions and requirements that specify desired behavior. Analysis of operating conditions reveal the limitations of sonar imagery, such as a high signal-to-noise ratio and inhomogeneous intensity patterns. In addition, the environment is sparse, with few distinct recognizable landmarks, limiting feature-based approaches. Because of these limitations, visual odometry is essential to reduce error build-up between loop closure corrections.

A Fourier-based approach to visual odometry is implemented, taking the whole image view into account instead of extracted features. By analyzing the dominant peak in the phase correlation matrix, the in-plane sonar motion between consecutive image frames can be estimated. Several image processing steps are necessary to improve peak sharpness, increasing the quality of registration.

To validate the proposed method, an experiment was conducted during cleaning of the Pioneering Spirit, the worlds largest construction vessel. Under normal circumstances, visual odometry showed less error build-up in the position estimate than wheel odometry. However, outliers appear when driving near the waterline, caused by reflections and wave reverberations. Ultimately, the proposed visual odometry method improves the current positioning system and serves as a basis for an integral SLAM implementation.

Table of Contents

List of Figures

List of Tables

Acknowledgements

The work presented in this document is part of my MSc. thesis project that is in collaboration with the company Fleet Cleaner and the Department of Systems and Control at the University of Technology in Delft.

I would like to thank my supervisors dr. A. Tejada Ruiz from the University of Technology in Delft, together with ir. D. Borota, ir. drs. C. de Vet, and ir. drs. A. Noordstrand from Fleet Cleaner, for their advice and guidance. Furthermore, I would like to thank my colleagues K. Cassee and B. Steensma for their support and companionship during the writing of this thesis.

Delft, University of Technology
September 24, 2018

Dave Verstrate

Chapter 1

Introduction

1-1 Motivation

The start-up company Fleet Cleaner has developed a cleaning robot that is able to remove fouling that has built up on the hull of ships. Fouling may consist, among other things, of algae, slime and barnacles. This fouling increases the frictional drag of a ship, resulting in increased fuel consumption. Compared to a hydraulically smooth hull, frictional drag may increase by up to 20.4% depending on the amount of fouling [1]. Fleet Cleaner has reported a reduction of fuel consumption by up to 5% from past cleanings. These savings provide economic benefits to shipping companies, while also reducing the environmental impact of the shipping industry.

The robot attaches itself to a ship's hull by the use of three large permanent magnets and moves on the surface using three hydraulically actuated wheels. The ship's hull is cleaned using high-pressure water jets. In contrast, other underwater vehicles often use thrusters to maneuver. The main advantage of using magnets and wheels is that the Fleet Cleaner robot is able to clean the ship's hull under and above the waterline. This is important as fouled areas may be present above the waterline due to variance in the ship's draft, especially during loading and unloading.

The Fleet Cleaner robot in its current operation, is controlled by operators on a support vessel. This is in contrast to Autonomous Underwater Vehicles (AUVs), a more recent type of underwater vehicle that is able to carry out missions without human intervention and without the need for a support vessel. AUV's have been applied in a variety of differend missions including under-ice exploration [2], neutralization of mines [3] and coral inspection [4].

When the robot is underwater, the operators have no direct line of sight to the robot and rely on a multitude of sensors to keep track of the pose (position and orientation) of the robot. Accurate and reliable localization is necessary for the operators to avoid obstacles, plan a correct path and keep track of cleaned areas. Furthermore, Fleet Cleaner has the desire to transition towards a semi-autonomous mode of operation, reducing operation costs and

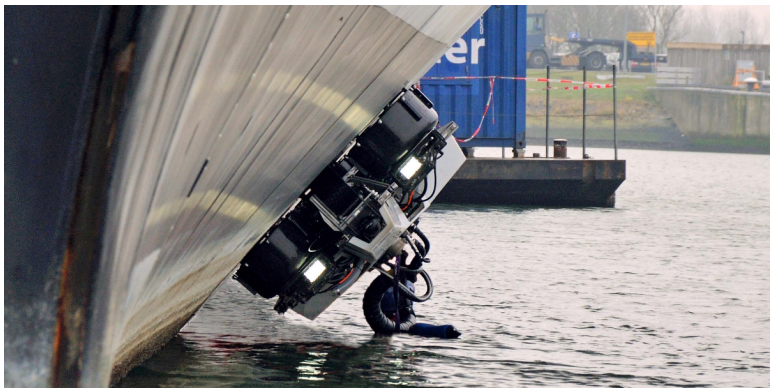


Figure 1-1: The Fleet Cleaner robot attached to a ship's hull.

improving speed and accuracy. A necessary contribution to this transition is the improvement of navigation and localization which is the motivation for this thesis.

This chapter serves as a background for the main problem that has motivated this thesis. The problem will be further detailed in Section 1-2. The objectives and methodology used in this thesis are described in Section 1-3 and finally the structure of this document is presented in Section 1-4.

1-2 Problem statement

Navigation and localization of underwater vehicles is a challenging subject due to the absence of an absolute positioning system, such as GPS. Therefore, modern underwater robots rely on, among others, the deployment of acoustic beacons, dead reckoning and terrain-based navigation [5]. The Fleet Cleaner robot currently uses a relative positioning system, using measurements from a depth sensor, wheel encoders and an IMU, which has been developed in a preliminary study [6]. This approach, however, suffers from unbounded error build-up over time resulting in a drifting position estimate. Currently the operators need to periodically reset the position estimate based on visual feedback from the optical and sonar imaging systems. They correspond recognizable landmarks, such as weld lines, from the visual images to the map of the ship's hull.

The use of pre-deployed infrastructure, such as acoustic beacons, is considered to be impractical and expensive. Therefore, Fleet Cleaner is looking for other ways to achieve drift-free localization. In the field of mobile robotics, drift-free localization using visual information has been a key topic of research. Simultaneous Localization and Mapping (SLAM) approaches make use of visual information to construct a map of the environment while concurrently localizing the robot in that map. By recognizing previously visited locations, the error build-up can be bounded. This is essentially what the operators now do manually by resetting the position estimate. SLAM is a widespread technique in above water robots and has more recently been implemented in underwater systems as well [7].

Acquiring visual images underwater is not without its challenges however. Optical imaging systems suffer from poor visibility in the presence of floating particles and turbulence. Cleaning operations are usually conducted in the harbor, where the visibility range is severely

Figure 1-2: Examples of optical images suffering from poor visibility.

limited, as shown in Figure 1-2. Due to this limitation, underwater applications often rely on sonar technology instead of optical technology. Sonar systems are able to operate at longer range underwater and are less affected by turbulent conditions. However, sonar images in general suffer from more noise, a lower resolution and are more difficult to interpret. In recent years 2D Forward Looking Sonars (FLSs) have emerged as an alternative to optical cameras in underwater environments. They are able to provide high quality acoustic images at a high frame rate. These properties make FLSs the optimal choice as input for an underwater SLAM system.

Most visual SLAM systems feature the same framework, which is described in Table 1-1. As the robot is expected to drive long segments without returning to a previously visited location, it is vital that the position estimate does not drift significantly during these segments. Visual odometry provides benefits to reduce the error build-up, as it is not affected by wheel slip.

Visual odometry	The robot pose is estimated by computing the motion between subsequent image frames.
Loop closure detection	A previously visited location is recognized by cross-referencing the current image frame to temporally distant frames.
Pose and map estimation	Odometry estimations are combined with the correspondences found by loop closure detection to estimate the pose trajectory and landmark positions.

Table 1-1: Subsystems of a typical visual SLAM system.

1-3 Thesis approach

Following the problem description, the goal of this thesis is to *develop a sonar-based underwater SLAM framework tailored to the working environment of ship hull cleaning robots.*

This objective has been split in multiple sub-objectives to fit the scope of the thesis. Due to time constraints a complete SLAM system has not been realized, but the scope will be restricted to the conceptual design of this system and a proof of concept of one of its components, a sonar-based visual odometry algorithm:

1. Propose a conceptual design for a SLAM system
2. Validate the sonar-based visual odometry algorithm with a proof of concept

Methodology

The systematic approach to engineering design by Pahl et al. [8] is used as a guide to fulfill the objectives. The approach used consists of four main phases, described in Table 1-2.

1.	**Task clarification**	The objective of task clarification is to gather information about the requirements that the design needs to fulfill and the existing operating conditions.
2.	**Conceptual design**	The principle solution is determined by abstracting the essential problems, defining function structures, searching for suitable working principles and combining these in a working structure.
3.	**Embodiment design**	The conceptual design is implemented in line with the technical criteria. Subsequently, system parameters are weighted and analyzed to be able to solve the problem
4.	**Proof of concept**	In the final design phase the working structure is demonstrated in an expiremental setup. The results serve as a validation of the conceptual design.

Table 1-2: Main phases of systematic approach to engineering design [8].

The task clarification and conceptual design phase will be conducted for a complete SLAM system, resulting in a principle solution for the Fleet Cleaner robot. The embodiment of this design and proof of concept will be restricted to one of the sub components, visual odometry, due to the scope of this thesis.

1-4 Document road map

An overview of the document structure is given in Figure 1-3 while a more detailed outline is given below:

Chapter 2 describes the sonar operation principles and a geometry model outlining the image acquisition process. This chapter serves as a background for sonar imaging and as context for the subsequent chapters.

Chapter 3 defines the operating conditions that are imposed on a potential SLAM system. The environment as well as current limitations of the navigation system are considered.

Chapter 4 sets up a list of desired requirements for the various subsystems. The requirements are split into functional and non-functional requirements. The former specifies the abstract behavior while the latter defines criteria that are used to evaluate performance.

Chapter 5 addresses the conceptual design phase. A small literature survey is conducted to analyze the main paradigms for each subfunction. A final solution is proposed in a qualitative manner.

Chapter 6 covers the implementation of the visual odometry component. The underlying principles are discussed as well as the specific implementation for the Fleet Cleaner robot.

Chapter 7 presents the final results of the visual odometry algorithm. An experiment was conducted during a cleaning operation. The algorithm is rated against the requirements.

Finally, **Chapter 8** concludes the document with the main contributions and recommendations for future work.

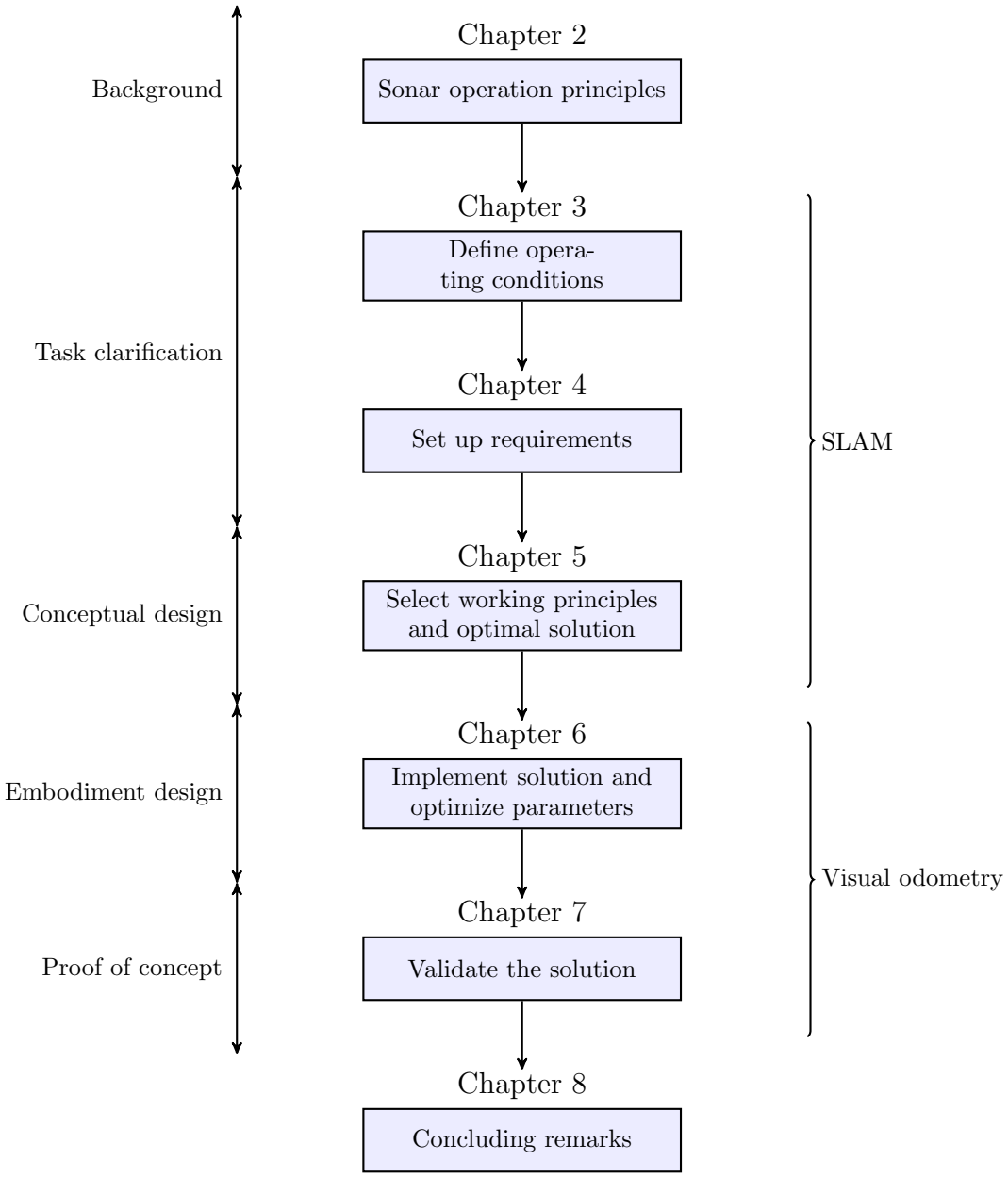


Figure 1-3: Overview of the activities in each chapter. On the left the main phases of the design methodology are listed. On the right it is outlined whether the chapter is concerned with the complete SLAM system or the visual odometry sub-component.

Chapter 2

Sonar operation principles

This chapter describes the basic principles of Forward Looking Sonar (FLS) operation and image formation, which will serve as the basics that contextualize the later methods. In Section 2-1 the fundamentals of acoustic propagation are described, which is the fundamental working principle for all sonar systems. The specifics of FLS operation are discussed in Section 2-2. Finally in Section 2-3 an imaging model is proposed, which is used for the remainder of the document.

2-1 Acoustic propagation

The following introduction to the basic principles of sonar and underwater acoustic propagation is given based on [9]. Acoustic waves are characterized by the wave frequency and speed of sound. Loss and attenuation can happen while propagating simply through the water or by traveling from one medium to a different medium. Both situations will be discussed next.

2-1-1 Transmission loss

Acoustic waves lose energy simply by propagating through a medium. The main contributing factors are spherical spread and absorption.

Spherical spread

The sound wave spreads as a spherical wave from the source. As the wave travels, the intensity I decreases with range R in inverse proportion to the sphere's surface:

$$I \sim \frac{1}{R^2} \tag{2-1}$$

In two way propagation the reflector returns the sound wave back to the receiver. The returned signal spreads again as a spherical wave. Thus the intensity of the returned signal becomes

$$I \sim \frac{1}{R^4} \tag{2-2}$$

Absorption

The sound wave is attenuated by the seawater due to viscosity and chemical processes. This absorption is frequency dependent such that lower frequencies travel farther than higher frequencies. In addition the absorption is dependent on temperature, salinity, depth and pH.

2-1-2 Refraction, reflection and scattering

When the sound wave travels from one medium to a different medium with a different sound speed, the wave will partially reflect and partially refract into the other medium, see Figure 2-1.

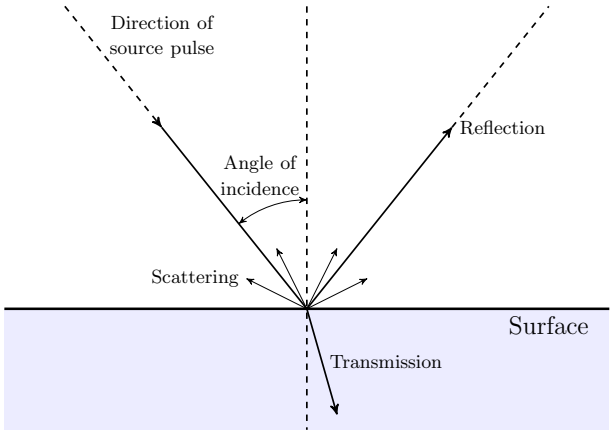


Figure 2-1: Refraction, reflection and scattering when an acoustic wave travels to a different medium.

When the sound wave is refracted, the angle of refraction is given by Snell's law:

$$\frac{\sin\theta_1}{c_1} = \frac{\sin\theta_2}{c_2} \tag{2-3}$$

The incoming acoustic wave will partly reflect with the reflection angle equal to the incident angle. The amount of reflection and refraction is dependent on the angle of incidence and material properties. In addition to this specular reflection, diffuse reflection is also considered which results in scattering.

When reflecting from a rough surface the acoustic waves are scattered diffusely in random directions. The amount of scattering depends on the roughness of the surface. Scattering is vital when the incident wave is not normal to the surface, i.e. the specular reflection cannot reach the receiver. For these constraints, visibility depends heavily on the roughness of the observed surface.

2-2 FLS operation

Fleet Cleaner employs a multibeam FLS which provides high definition acoustic images at a high frame rate. An FLS is aimed forward and often tilted, granting the same functionality as optical cameras. These type of sonars are more recently used by hovering Autonomous Underwater Vehicles (AUVs), inspecting man-made structures [5].

The FLS, like other active sonar systems, operates by emitting a sound wave, spanning the azimuth (θ) and elevation (ϕ) angles. Figure 2-2 shows the geometric definitions that will be used. The intensity of the acoustic return is sampled by an array of transducers as a function of range and bearing. The range, r is estimated from the travel time, t, of the acoustic wave and the sound speed, c, of the medium:

$$r = \frac{ct}{2} \tag{2-4}$$

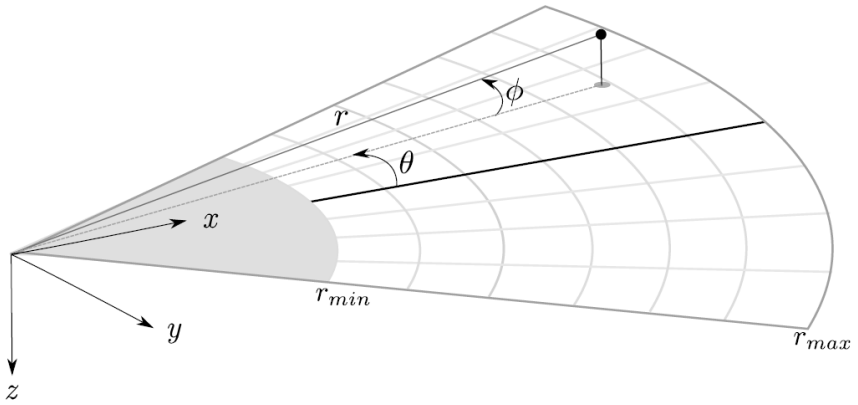
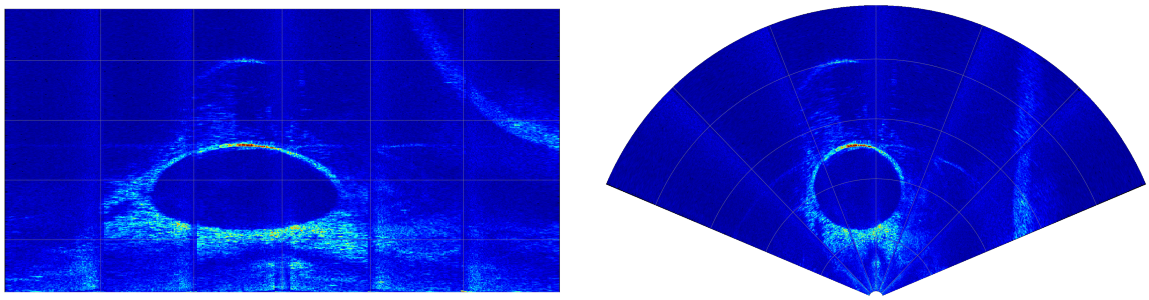


Figure 2-2: Projection of a point in 3D to the 2D sonar image plane [10].

Note that only the azimuth angle θ can be estimated by the sonar, i.e. for a point on the returned image it can have originated from anywhere on the corresponding elevation arc. This 3D information is lost in the mapping to a 2D image. Figure 2-2 shows a 3D point that is mapped to the 2D plane, removing information of the elevation angle ϕ. The 3D to 2D projection is further detailed in Section 2-3.

Following the particular nature of the transducers, the sonar images are captured in polar coordinates, representing range r and azimuth angle θ. The images can be transformed to Cartesian coordinates, which allow for a more easier interpretation of the images. It should be noted that the Cartesian images have a non-uniform resolution due to this transformation. The difference is shown in Figure 2-3.



(a) Polar coordinates **(b)** Cartesian coordinates

Figure 2-3: Sonar image captured in the polar domain (a) and transformed to Cartesian coordinates (b).

2-3 Sonar geometry model

In this section the sonar imaging geometry is analyzed. Two models are considered: a linear approximation and a more exact model. Shortcomings of both models are discussed and a final model is proposed that will be used for the remainder of this report.

Consider point $\mathbf{P}$ in spherical coordinates (r, θ, ϕ), again refer to Figure 2-2 for the geometric definitions. In the sonar (x_s, y_s, z_s) frame it is described by the following Cartesian coordinates

$$\mathbf{P} = \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} = \begin{bmatrix} r\cos\phi\cos\theta \\ r\cos\phi\sin\theta \\ r\sin\phi \end{bmatrix} \tag{2-5}$$

The projection of this point on the image plane (u, v) is defined as

$$\hat{I}(\mathbf{P}) = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix} = \frac{1}{\cos\phi}\begin{bmatrix} x_s \\ y_s \end{bmatrix} \tag{2-6}$$

This projection is non-linear and depends on the elevation angle ϕ.

2-3-1 Approximated model

Considering the narrow elevation angle of FLSs, the non-linear part can be approximated by

$$\cos\phi \approx 1 \tag{2-7}$$

This approximation is equivalent to assuming that all points are located on the zero-elevation plane ($\phi = 0$). A typical elevation angle of $[-10°, 10°]$ corresponds to a limit on the non-linear component of $1 \leq \frac{1}{\cos\phi} \leq 1.0154$ or a maximum error of 1.54%. This approximation holds when the scene's relief in the elevation direction is small compared to the range [11]. With a small beam width in the elevation angle and tilted to a small grazing angle, the sonar

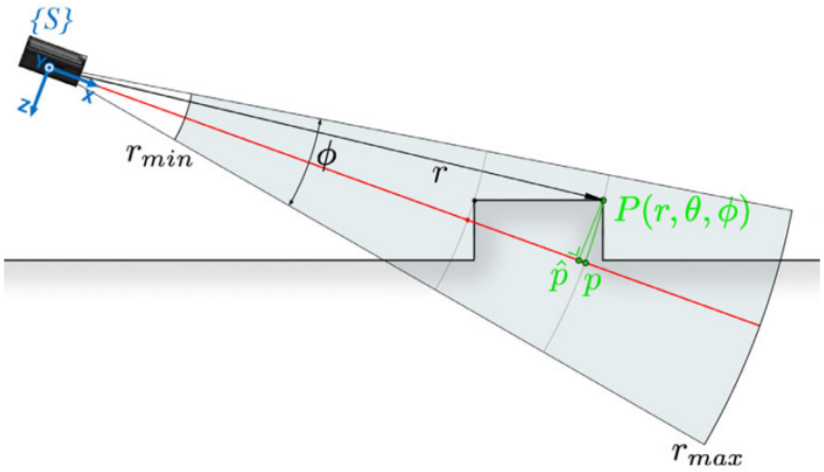


Figure 2-4: The sonar projects point $\mathbf{P}$ on the image plane along the elevation angle, mapping to point $\mathbf{p}$, which is approximated by an orthographic projection, mapping to point $\hat{\mathbf{p}}$. [12]

imaging geometry falls within this condition. The difference between the approximation and the actual projection is illustrated in Figure 2-4.

A change in azimuth angle θ is preserved by the projection, i.e. the image rotates by the same angle as the sonar with respect to its vertical axis. This can be shown by calculating the angle between two image points [10]. Let P and P' be the same point, rotated by α around the z-axis. The estimate of rotation of the sonar is $\hat{\alpha}$. Then, the angle between the image points can be found by calculating the inner product:

$$\|\mathbf{p}\|\|\mathbf{p}'\|\cos(\hat{\alpha}) = \hat{I}(\mathbf{p}) \cdot \hat{I}(\mathbf{p}') \tag{2-8}$$

$$= \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix} \cdot \begin{bmatrix} r\cos(\theta+\alpha) \\ r\sin(\theta+\alpha) \end{bmatrix} \tag{2-9}$$

$$= \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix} \cdot \begin{bmatrix} r(\cos\theta\cos\alpha - \sin\theta\sin\alpha) \\ r(\sin\theta\cos\alpha + \cos\theta\sin\alpha) \end{bmatrix} \tag{2-10}$$

$$= r^2\cos^2\theta\cos\alpha + r^2\sin^2\theta\cos\alpha \tag{2-11}$$

$$= r^2\cos\alpha \tag{2-12}$$

$$\hat{\alpha} = \alpha \tag{2-13}$$

Rotations in the pitch direction will affect intensity values and limits of the sonar view, but do not change the projection of the points. Changes in roll will cause a compression of points along the y-axis. The roll angle is assumed to be small on the flat side of the ship, but will increase at the bow and stern of the ship where the surfaces are curved. Like roll, changes in z-direction do not change the projection of points, but only the insonified area.

Therefore, two images can be related by a translation (t_x, t_y) and a rotation θ. Considering two points from different images $\mathbf{p}$ and $\mathbf{p'}$ that represent the same point $\mathbf{P}$ in 3D, the points can then be related by a homography matrix $\mathbf{H}$, which describes the translation t_x, t_y and

rotation θ of the two images:

$$\mathbf{p'} = \mathbf{Hp} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p} \tag{2-14}$$

Obvious shortcomings of the simplified model are twofold:

- Only simple motion in 3 DOF can be estimated.
- Projection error due to orthographic approximation.

2-3-2 Non-approximated model

By lifting the small elevation angle assumption, a more exact model can be proposed. The rigid body motion model for features lying on the plane $\mathbf{n}$ is defined as [13]

$$\mathbf{P'} = (\mathbf{R} + \mathbf{t}\mathbf{n}^T)\mathbf{P} = \mathbf{QP} \tag{2-15}$$

where R is the rotation matrix and t is the translation vector, describing the rigid body motion. From this transformation, the homography relating two image points can be found [14]:

$$\mathbf{p'} = \mathbf{Hp} = \begin{bmatrix} \alpha q_{11} & \alpha q_{12} & \beta q_{13} \\ \alpha q_{21} & \alpha q_{22} & \beta q_{23} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p} \tag{2-16}$$

$$\text{with} \quad \alpha = \frac{\cos\phi}{\cos\phi'},\ \beta = \frac{r\sin\phi}{\cos\phi'} \tag{2-17}$$

q_{ij} denote the components of the Q-matrix. This more complex transformation is dependent on the elevation angles of every image point. This information is not directly available from the sonar sensor. However, the elevation arc of points on a flat surface can be estimated by its surface normal [15]. Estimation of the surface normal is done by assuming that the leading and trailing edges of the image correspond to the extreme elevation angles $(-\phi_{max}, \phi_{max})$. Then, the surface normal can be estimated using the sonar pitch and height. Furthermore, the elevation angle of objects can be estimated by considering the shadow cast by that object.

While the 6 DOF motion of the sonar can be estimated with the exact model, the estimation of elevation angles is not always trivial. When the image limits do not correspond to the extreme range values, error are introduced. This can be avoided by detecting the actual leading and trailing edged, but this can be a complex and inaccurate process [15].

2-3-3 Selection of model

Both models are summarized in Table 2-1. The exact model offers obvious benefits over the approximated model. First, the model allows the estimation of sonar motion in 6 DOF. However, when driving on the flat surfaces of the ship (sides and bottom) the movements in the sonars z-direction, pitch and roll are minimal. It is sufficient to estimate the planar transformation (x, y and yaw).

Furthermore, the exact model uses elevation angles in the estimation, which reduces the approximation errors in the simplified model. Computation of the elevation angles, however, is not completely trivial and is susceptible to its own inaccuracies.

Considering these benefits and downsides, the approximated model is found to be suitable to describe the image formation process. Although the projection is approximated, the error is expected to be insignificant as long as the elevation is small compared to the range of the sonar. The later sections of this report will therefore assume the use of the 2D approximated sonar model.

Model	Remarks
Approximated model	-3 DOF
	-Projection errors due to approximation
Exact model	-6 DOF
	-Requires estimation of elevation angles

Table 2-1: Overview of sonar image formation models.

Chapter 3

Operating conditions

Clearly establishing the operating conditions of the Fleet Cleaner robot is fundamental, because they influence the performance and functionality of the proposed Simultaneous Localization and Mapping (SLAM) design. Constraints and limitations of robots sensors as well as of the environment are analyzed. The final contribution of this chapter is a list of operating conditions for the SLAM system.

Conditions imposed by the employed sonar system are discussed in Section 3-1. In Section 3-2, the auxiliary sensors of the positioning system are discussed. The environment of the robot and its impact on a potential SLAM system is discussed in Section 3-3. Finally, an overview of the discussed operating conditions is presented in Section 3-4.

3-1 Sonar operating conditions

The sonar used by Fleet Cleaner is a Blueview M900-2250-130 [16]. This sonar has two frequency modes: 900 kHz and 2250 kHz, the full specifications are listed in Appendix A-1. The sonar unit mainly operates in the high frequency (2250 kHz) mode, which allows for higher quality images at the expense of a shorter range.

3-1-1 Sound speed

The acoustic propagation of the transmitted and received sound waves depend heavily on the properties of the medium. A simple empirical formula to estimate the speed of sound in seawater has been developed by Medwin [17]:

$$c = 1449.2 + 4.6T - 0.055T^2 + 0.00029T^3 + (1.34 - 0.010T)(S - 35) + 0.016D \qquad (3\text{-}1)$$

The speed of sound in water is found to be dependent on the temperature (T), depth (D) and salinity (S) of the water. The depth is dependent on the draft of the ship and does not feature

fluctuations that have a noticeable impact on the speed of sound. Salinity and temperature may vary throughout the year and across different harbors. As range estimation in the sonar images is dependent on the sound speed, a small approximation error is present.

Furthermore, some random fluctuations may affect the acoustic propagation. These effects include turbulence, currents and surface waves and are impacted by the harbor conditions. These can show up as artifacts or noise in the sonar images.

3-1-2 Geometry

The sonar imagery is further impacted by the placement of the sonar in relation to the observed surface. From the robot frame the sonar unit is displaced vertically and horizontally. Furthermore the unit is tilted to a small grazing angle of 10°. The current location and dimensions of the sonar unit are detailed in Appendix B.

The tilt angle is of particular importance as it impacts the field of view of the observed surface. A small tilt angle provides a wider field of view, while a large tilt angle can provide better observability of objects at the expense of a narrower perspective.

3-1-3 Image formation

The nature of sonar image formation introduces some further limitations that impact the ability to handle and process the images in subsequent processing steps [18]. The most important and relevant aspects are summarized here:

- **Non-uniform resolution:** As mentioned before, the transformation from the polar images to a Cartesian representation results in a non-uniform representation. The measurement sparseness increases with the range of the sonar, meaning that objects at a farther range are represented by a degraded resolution.

- **Speckle noise:** Sonar imagery in general suffer from a low Signal-to-Noise-Ratio (SNR), due to the presence of speckle noise. This type of noise is introduced by interference patterns of the sampled acoustic returns.

- **Inhomogeneous insonification:** A time varying gain mechanism is included that is used to compensate for transmission loss by spreading and absorption. As a result, similar objects at different ranges are represented by the same intensity values. However, inhomogeneous illumination pattern may still be present due to an increasing angle of incidence on some segments. Furthermore, vertical stripes are visible on the sonar images due to the overlapping of sonar beams.

- **Reflections:** In some circumstances reflections and reverbation artifacts may be visible. These result from acoustic returns reflecting from the water surface. These artifacts disturb the image content and may cause ambiguities between images.

3-2 Positioning system operating conditions

In this section the conditions imposed by the current positioning system are discussed. First the movement of the robot is detailed. Subsequently, the positioning system is discussed focusing on the employed sensors and their limitations.

3-2-1 Movement

The robot is attached to the ship's hull by three large magnets. As such the movement of the robot is restricted to the surface of the ship's hull, meaning that the orientation of the robot with respect to the earth-fixed frame depends on the curvature of the surface.

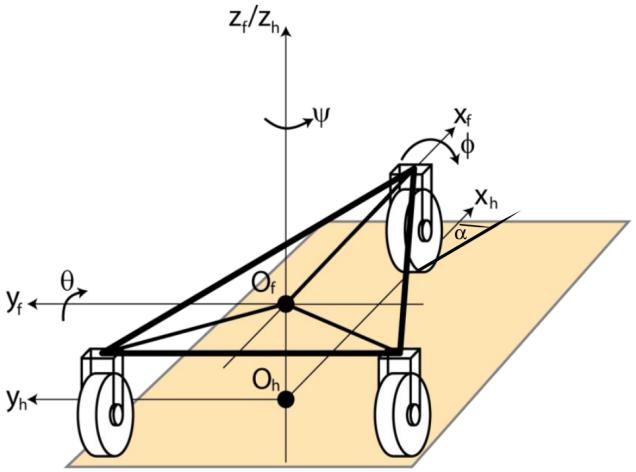


Figure 3-1: Robot parameters and coordinate frames [6].

In Figure 3-1 displays the different coordinates and frames for the robot. The robot is steered with the angle of the front wheel, α. The rate of change in the global coordinates can then be given as

$$\dot{x} = \cos(\psi + \alpha)v \tag{3-2}$$

$$\dot{y} = \sin(\psi + \alpha)v \tag{3-3}$$

with v the velocity of the front wheel, aligned with the steering angle. The rate of turn of the robot can be found by decomposing the velocity in the x- and y-component with respect to the robot frame. The y-component is always perpendicular to the direction of the robot and contributes to the rate of turn:

$$\dot{\psi} = \frac{v_y}{l} \tag{3-4}$$

$$\dot{\psi} = \frac{v \sin \alpha}{l} \tag{3-5}$$

with l the distance between the front wheel and the center of the robot.

3-2-2 Auxiliary sensors

A set of relative and absolute sensors are currently used to localize the robot. These include:

- Inertial Measurement Unit (IMU)
- Depth sensor
- Wheel encoders
- Steering angle encoder

Full specifications are listed in Appendix A. The IMU is used to measure the orientation of the robot. This document uses an intrinsic Euler angle convention. The names of the rotation angles are based on the aerospace convention: yaw (ψ, z-axis), pitch (θ, y-axis), and roll (ϕ, x-axis), as depicted in Figure 3-2a. Roll and pitch measurements are referenced to the direction of gravity. Due to the use of magnets to attach to the ship's hull, the IMU cannot employ a magnet, causing the orientation of the heading to be un-referenced. This results in a heading drift over time. When the robot is attached to the side of the ship with the coordinate system as shown in Figure 3-2b, the un-referenced orientation becomes the rotation around the ship's hull fixed y-axis, or pitch.

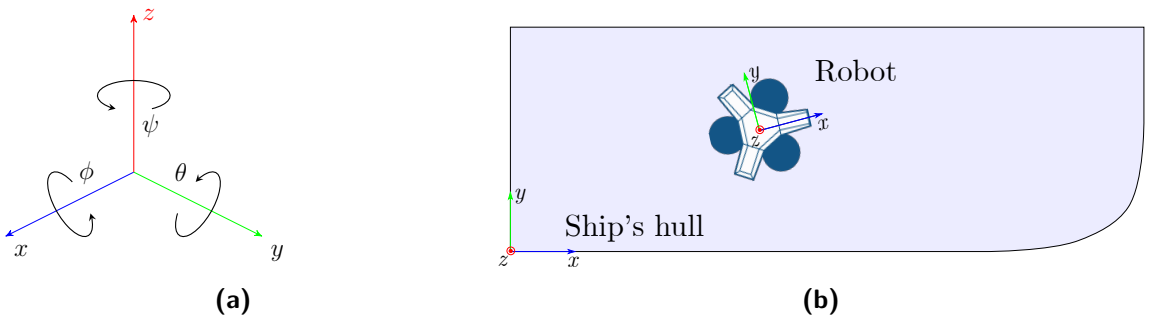


Figure 3-2: Coordinate systems used throughout the document. The associated orientations are found using the right-hand rule (a). On the flat side of the ship's hull, the coordinate system as shown in (b) is used.

For the localization, only the depth or y-coordinate has an absolute measurement. For the x- and z-coordinates, the position system relies solely on relative positioning. The current location is estimated through wheel odometry, introducing error build-up in the estimate. Sensor noise contributes to the error build-up as small errors are added up and integrated to the position estimate. However, the main cause of the error build-up is wheel slip, which happens when the robot is driven over uneven or slippery surfaces. Furthermore, the wheel encoders suffer from a high rate of hardware failure due to the build-up of dirt.

3-3 SLAM operating conditions

The typical trajectory of the robot during cleaning looks like a lawn mower pattern to ensure maximal coverage, as shown in Figure 3-3. This means that the robot is not in the exact same position twice. When the robot re-observes a landmark, this will be from a slightly different viewpoint, complicating the ability to recognize previously visited locations.

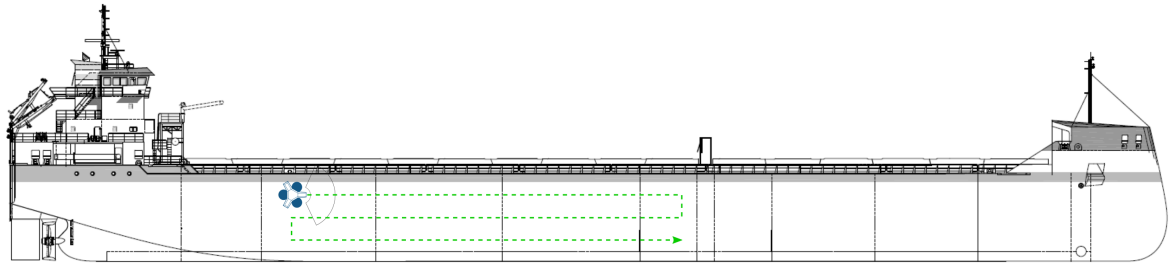


Figure 3-3: Trajectory of robot during cleaning in land mower pattern

From cleaning operations and inspection reports, Steensma has identified the following elements that may be encountered during navigation on the ship's hull [19]:

- bilge keels
- bow thrusters
- depth markings
- holes
- sea chests
- stabilizing fins
- stern thrusters
- weld lines
- zinc anodes

Some of these elements, such as holes or thrusters may be used as recognizable landmarks for a SLAM system. However, the ships are typically sparse with these elements and segments without distinct visible landmarks are expected.

3-4 Overview

The conditions and limitations discussed in this chapter are summarized in Table 3-1.

Operating conditions	Topic	Specifications	
FLS sonar	FLS specifications	Max range	$r_{max} = 10\,\text{m}$
		Range resolution	$dr = 6\,\text{mm}$
		Angular resolution	$d\theta = 0.18°$
		Field of view	FOV $= 130°$
		Beam width	$\theta \times \phi = 1° \times 20°$
		Number of beams	$n = 768$
		Acoustic frequency	$f_{sonar} = 2250\,\text{kHz}$
		Update frequency	$f_{update} = 10\,\text{Hz}$
	Acoustic propagation	Temperature	
		Salinity	
		Depth	
		Water fluctuations	
	Geometry	Displacement from robot frame	$[1141, 0, 306]$ mm
		Tilt	$\phi = 10°$
	Image Formation	Non-uniform resolution	
		Speckle noise	
		Inhomogeneous insonification	
		Reflections from water surface	
Robot	Movement	Attached to ship's hull	
	Positioning system	Unreferenced IMU heading orientation	
		Relative localization of x- and z-coordinates	
		Drifting due to wheel slip	
		Hardware failure of wheel encoders	
SLAM	Landmarks	Landmarks are re-observed from different viewing angles, due to lawn mower pattern	
		Segments without any distinct visible landmarks	

Table 3-1: Overview of operating conditions

Chapter 4

Requirements

The task clarification phase is finalized by setting a list of requirements for the Simultaneous Localization and Mapping (SLAM) system. A distinction is made between functional and non-functional requirements. The former describe the desired behavior of the SLAM system, while the latter specify criteria that are used as evaluation metrics. The final contribution of this chapter is a list of requirements and metrics that the proposed design needs to fulfill.

In Section 4-1 the functional requirements are proposed. The system is split into multiple subsystems and for each subsystem the requirements are set. The non-functional requirements are detailed in Section 4-2. A final overview is presented in Section 4-3.

4-1 Functional requirements

The set of functional requirements describe the behavior and functionality of the system. These requirements are described in a qualitative way and specify the input that may be used as well as the desired output of the system.

The basic SLAM system architecture is discussed first and serves to give the reader a basic understanding for the later sections in this chapter. The system is further divided into subsystems, for which requirements are further specified.

4-1-1 SLAM architecture

A visual SLAM system uses visual information to track the robot position and landmark positions. Position estimates from auxiliary sensors, such as IMU and wheel odometry, may optionally be used as well to improve the estimate. In the most basic form this amounts to visual odometry. As with wheel odometry, this results in error build-up. The error build-up can be reset in a SLAM system by observing previously visited locations. In literature this is often referred to as loop closure or data association. The associations to previous observations, along with the position estimates from visual odometry and auxiliary sensors,

are used to produce an updated position of the robot on the ship's hull as well as a map representation of tracked landmarks.

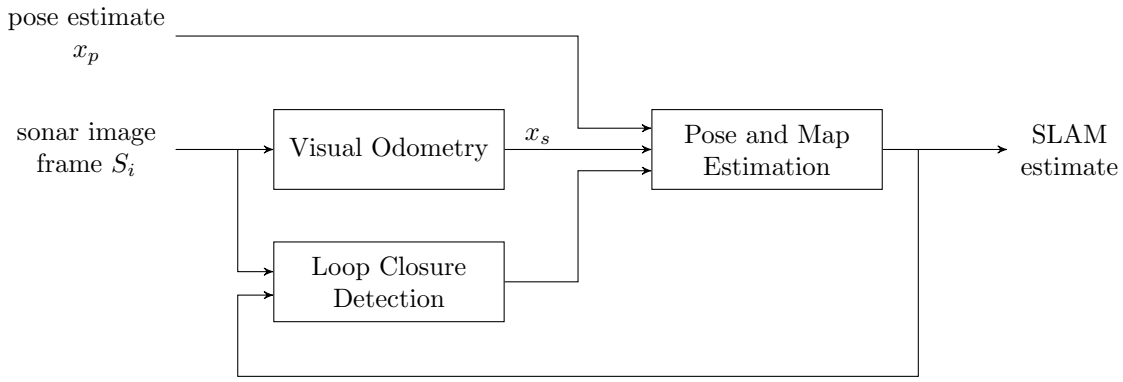


Figure 4-1: Basic system architecture of SLAM.

This basic interaction between the different subsystems is shown in 4-1. Visual odometry and loop closure detection are often referred to as the front-end, as these subsystems are concerned with abstracting sensor data. The SLAM back-end is the pose and map estimation which uses the abstracted data to produce an estimate.

Given the operating conditions, the SLAM system uses as input:

- Sonar images (polar and/or Cartesian coordinates)
- Position estimate from IMU and wheel odometry

Where the usage of auxiliary sensors is optional. After abstraction of the sensor data, the system must produce as output:

- Position estimate of robot on ship's hull
- Map representation of tracked landmarks

In the next sections the requirements of each subsystem will be further detailed.

4-1-2 Visual odometry

Visual odometry is used to estimate the robot motion from subsequent sonar images. The motion estimates can then be integrated to a position estimate of the robot on the ship's hull. Along with the position estimate from IMU and wheel odometry measurements, this provides an initial position estimate for SLAM.

Given the operating conditions, this subsystem uses as input:

- Pair of subsequent sonar image frames (polar and/or Cartesian coordinates)
- Auxiliary sensor measurements (optional)

Auxiliary sensor measurements are optional and may be used for absolute measurements of depth and orientation. Visual odometry must produce as output:

- Position estimate of the robot on the ship's hull

4-1-3 Loop closure detection

With loop closure detection the system is able to correct drift that has accumulated from the odometry measurements. This process consists of two steps. First, the current location has to be matched to a set of previously visited locations. When an association between locations is found, the current robot pose has to be related to the tracked landmark.

Given the operating conditions, this subsystem uses as input:

- Current sonar image frame (polar and/or Cartesian coordinates)
- List of tracked landmarks
- Current position estimate

Loop closure detection relies on the current SLAM estimate and thus the accuracy of this estimate is vital. As an output this subsystem produces:

- Match between current observation and a previous observation
- Relative transformation to previous observation

4-1-4 Pose and map estimation

This subsystem is involved in simultaneously estimating the robot position and map of the ship's hull with information from the previous subsystems. The map can be represented in different ways, but the most important aspect is tracking landmarks that are used for navigation and obstacle avoidance. Instead of building a map from scratch, some shipping companies offer general arrangements of the ship, which can be used to locate landmarks and the robot position relative to those landmarks.

Given the operating conditions, this subsystem uses as input.

- Position estimate from visual odometry
- Position estimate from IMU and wheel odometry
- Loop closure match and relative transformation, if detected
- A priori map information (optional)

The estimation is dependent on the accuracy of the previous subsystems. The output of this system is the final output of the SLAM system:

- Position estimate of robot on ship's hull
- Map representation of tracked landmarks

4-2 Non-functional requirements

This set of requirements specify the desired performance of the system. The requirements are proposed in the form of metrics that can be used to compare and evaluate different working principles.

4-2-1 Accuracy

Improvement of the accuracy of the positioning system of the robot is one of the main objectives of the SLAM system and thus the accuracy of the position estimate is an important criteria to consider. An important term to express this term is the error build-up, which is used to describe the accumulated error between the true position and estimated position per true meter traveled:

$$\text{EBU} = 100\% \cdot \frac{||\mathbf{p}_{est}(T) - \mathbf{p}_{true}(T)||}{\sum_{i=T-N}^{T} ||\mathbf{p}_{true}(t_i) - \mathbf{p}_{true}(t_{i-1})||} \tag{4-1}$$

where $\mathbf{p}_{true}(T)$ is the true position and $\mathbf{p}_{est}(T)$ is the estimated position at time instance T and N is the size of the data sequence over which the error build-up is calculated.

With correct loop closure detection, the error build-up can be reset. The maximum error build-up is thus expected before a loop closure event, on segments without distinct visible landmarks. In a loop closure event the acceptable error tolerance is assumed to be 10 m, as the landmark should then be somewhere in the sonar field of view. Assuming a typical ship length of 200 m and distinct landmarks at the start and end of the lane, where the error build-up can be reset. Traveling along the length of the ship, the error build-up should then not be more than 10 m or 5 % to ensure successful loop closure detection.

- EBU $< 5\,\%$

4-2-2 Robustness

A SLAM system may be susceptible to many failure modes, either algorithmic or hardware-related. Robustness to these failure modes is thus of particular interest. The most important cause of algorithmic failure in a SLAM system is erroneous loop closure detection [20]. Incorrect landmark associations (false positives) cause the pose and map estimation to produce wrong estimates. On the contrary, when the system rejects or misses a correct loop closure event (false negatives), fewer measurements are used for estimation, reducing accuracy.

In pattern recognition precision and recall are often used metrics to describe the accuracy of an algorithm:

$$\text{PRC} = \frac{TP}{TP + FP} \tag{4-2}$$

$$\text{RCL} = \frac{TP}{TP + FN} \tag{4-3}$$

Where TP are True Positives, FP false positives and FN false negatives. Since false positives can lead to completely wrong estimates, from which recovery is not trivial, precision needs to be as high as possible. While false negatives contribute to a degraded accuracy of the estimate, their presence is not enough to trigger failure in the system. As such, the requirements are chosen as:

- PRC > 0.98
- RCL > 0.90

4-2-3 Computation time

For autonomous behavior of the Fleet Cleaner robot it is necessary for the algorithm to run online. The computation time for visual odometry, during landmark sparse segments, and the computation time during a loop closure event are considered.

Visual odometry relies on subsequent image pairs and as such the computation time should be faster than the image update rate, which is 10 Hz. However, the image pairs can also be chosen every n-th frame, so that a longer computation time is allowed. Fleet Cleaner has stated a required accuracy within 10 cm, in order to make the robot autonomous. With the travel speed of $0.3\,\mathrm{m\,s^{-1}}$ of the robot, the computation time of the postion estimate needs to be at most 0.33 s, otherwise the true position exceeds the 10 cm requirement. This means the computation can be done every 3 sonar image frames.

Loop closure detection and re-localization based on these observations is computationally heavy, increasing with time and scale of the mission. However, a low computation time is desired to improve accuracy. During a successful loop closure event the error build-up may be removed, whereas erogenous detection may be the cause of failure as discussed before. It is for this reason that the requirement for computation time during loop closure is chosen to be more lenient, to make the trade-off in favor of accuracy.

- $T_{VO} < 0.3\,\mathrm{s}$
- $T_{LC} < 2\,\mathrm{s}$

4-3 Overview

An overview of the requirements is presented in Table 4-1 (functional) and 4-2 (non-functional).

Subsystem	Requirements	
Visual odometry	Input:	- Pair of sonar frames (polar or Cartesian)
		- Auxiliary sensor data (optional)
	Output:	- Robot position estimate
Loop closure detection	Input:	- Sonar image frame polar or Cartesian
		- List of tracked landmarks
		- Robot position estimate
	Output:	- Association with previous observation
		- Relative transformation to previous observation
Pose and map estimation	Input:	- Visual odometry position estimate
		- IMU and wheel odometry position estimate
		- Loop closure detection and relative transformation
		- A priori map information (optional)
	Output:	- Robot position estimate
		- Map representation of tracked landmarks

Table 4-1: Overview of functional requirements

Metric	Requirements
Accuracy	EBU $< 5\,\%$
Robustness	PRC > 0.98
	RCL > 0.90
Computation time	$T_{VO} < 0.3\,\mathrm{s}$
	$T_{LC} < 2\,\mathrm{s}$

Table 4-2: Overview of non-functional requirements

Chapter 5

Working principles

In this chapter technical solutions that match the requirements are presented. Different working principles are discussed for each of the subsystems. The working principles are to be combined in an optimal working structure or principle solution to the problem, taking into account previously discussed goals, operating conditions and requirements.

In Sections 5-1, 5-2 and 5-3 working principles for respectively visual odometry, loop closure detection and, pose and map estimation are discussed. The optimal solution is proposed in Section 5-4 and a final overview is given in Section 5-5.

5-1 Visual odometry

The basic principle of visual odometry is to estimate camera (or in this case sonar) motion by analyzing image pairs. Most techniques are based on extracting features from the images and tracking their movement. View-based approaches, in contrast, make use of the entire image content.

All feature-tracking approaches basically consist of the following steps:

- Feature extraction
- Feature matching
- Motion estimation

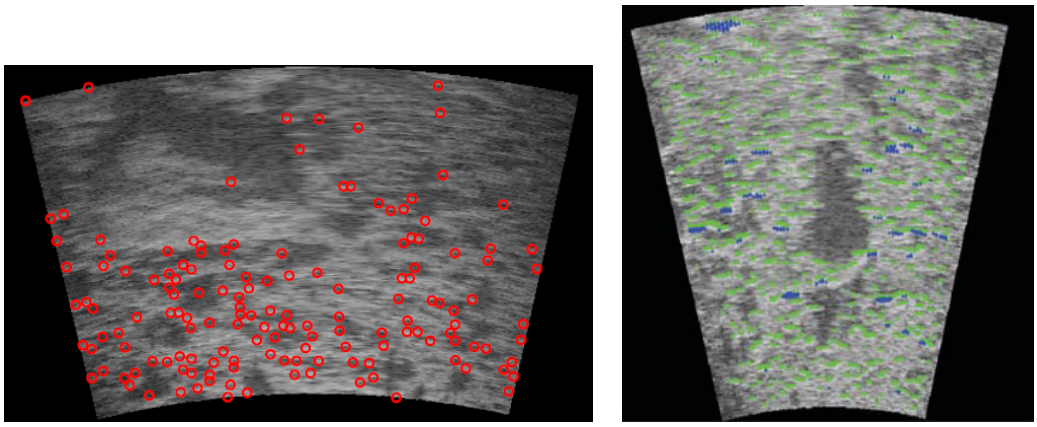
Where the approaches differ is the type of features that are extracted. The features can be extracted at pixel-level or at region-level, as shown in Figure 5-1. We will go into detail into each of these approaches.

5-1-1 Pixel-level feature registration

Pixel-level feature points are described by the local area around the feature point. These are based on a change of an image property, such as intensity, color and texture [21]. These points need to be matched in different images and thus are often invariant to scale and/or rotation. The types of features extracted by these methods include corners, edges and blobs, the more popular feature descriptors being Harris corners [22], Scale-Invariant Feature Transform (SIFT) [23] and Speeded Up Robust Features (SURF) [24].

The points are matched based on the similarity of the desciptors. Usually this procedure is followed by an outlier-rejection algorithm such as RANdom SAmple Consensus (RANSAC) [25], to remove wrong correspondences. The sonar motion is calculated by estimating the homography [26].

Most feature descriptors were developed for optical imagery and their performance on sonar images can be unreliable. In [27], the authors used SIFT descriptors to match features. They report a very low amount of inliers (12 correct matches from 150 features). This results is further underlined by Hurtós [28], noting low repeatability rates.



(a) **(b)**

Figure 5-1: Example of extracted features at (a) pixel level [27] and (b) region level [29].

5-1-2 Region-level feature registration

As noted before, sonar images suffer from weaker textures and more speckle noise when compared to optical images, which impacts the ability to reliably extract pixel-level feature descriptors. More stable features can be extracted on a region-level. These features can be extracted based on sharp intensity changes [10]. In a different approach the features are focused on bright blobs, representing objects or structures, and darker regions that correspond to cast shadows [29]. A clustering algorithm to detect objects such as gratings was proposed by Steensma in the preceding work [19].

Alignment and matching of the region-level features is often solved as an optimization problem. A popular method is Normal Distribution Transform (NDT) [30]. This region-level feature registration approach has been implemented on sonar images in various applications [10, 31, 32]. These results show more stability than the pixel-level features and have

been successfully applied in sonar-based Simultaneous Localization and Mapping (SLAM) applications.

5-1-3 View-based registration

View-based registration approaches are characterized by the lack of using detected features. Correspondences are estimated by analzing large areas or even the complete image content. Since these approaches are able to process more information, it is presumed they are more robust to changes in the viewpoint and outliers. The downside to these approaches is that only similarity transformations can be identified. However, the simplified sonar geometry model falls within this limitation. These methods include approaches based on spatial-correlation and Fourier analysis [33].

Spatial correlation

Spatial correlation is found by analyzing the normalized cross-correlation [34]:

$$\gamma(u,v) = \frac{\sum_{x,y}(f(x,y)-\bar{f})(g(x-u,y-v)-\bar{g})}{\sqrt{\sum_{x,y}(f(x,y)-\bar{f})^2 \sum_{x,y}(g(x-u,y-v)-\bar{g})^2}} \tag{5-1}$$

Where $f(x,y)$ and $g(x,y)$ are the image frames and $\bar{f}$, $\bar{g}$ indicate the means of the images. The displacement of the images can then be found by estimating the location of the peak of the cross correlation. Downsides of this method are high computation effort and difficulty of reliably estimating maxima [33].

Fourier analysis

This method follows a similar approach to spatial correlation, but instead correlation is found by analyzing the phase correlation matrix in the Fourier domain:

$$C(u,v) = \frac{F(u,v)G^*(u,v)}{|F(u,v)G^*(u,v)|} = e^{-j(ut_x+vt_y)} \tag{5-2}$$

Where $F(u,v)$ and $G(u,v)$ are the 2D Fourier transformed image frames, and $G*(u,v)$ denotes the complex conjugate. The inverse Fourier transform is applied to the phase correlation matrix and the displacement is then found by estimating the location of the peak of this function. In a comparison with feature-based methods, the Fourier registration method showed superior performance and seemed better suited to the limitations of sonar images [28], especially in feature sparse environments.

5-1-4 Comparison

The discussed methods are compared against the non-functional requirements from Section 4-2. The comparison is made qualitatively using information and results from the discussed literature.

Working principle	Accuracy	Robustness	Computation time
Pixel-level features	+/−	− −	++
Region-level features	+	+/−	+
Spatial correlation	++	−	− −
Phase correlation	++	+	+/−

Table 5-1: Comparison of visual odometry methods

While both feature extraction methods have a simple and fast implementation, performance on sonar images is questionable. Low stability and repeatability will impact the robustness of these methods. The correlation methods are better tailored to the limitations of sonar images and are expected to perform better. From these methods, phase correlation offers higher computational efficiency and more reliable peak detection.

5-2 Loop closure detection

Loop closure detection can be carried out by not only matching the current location to recent locations but to all previous locations. Matching can be done based on features or complete images, as was described for visual odometry. The discussed loop closure detection approaches are categorized based on where the loop closure detection takes place: in the map space or in the image space [35].

Image processing

Loop closure detection methods are heavily dependent on the extracted information from images. The same methods that were discussed in Section 5-1 can be used. However, now with the added requirement of long-term matching. As the robot itself interacts with the environment by cleaning fouling, feature descriptors are expected to vary too much over time. View-based approaches suffer from the same problem and in addition provide no description of the image content. However, objects on the ship, such as gratings, remain static and are therefore viable landmarks to be used in loop closure detection. Object detection from sonar imagery has been explored by Sawas [36] and in the preceding work of Steensma [19].

Next, loop closure methods are discussed that can match the tracked objects.

5-2-1 Submap matching

In this type of loop closure detection, correspondences are found in the map space. The map is divided into smaller submaps and loop closures are found by looking at common features, taking visual appearance and spatial information into account. Figure 5-2 shows matched features between two different submaps. The detection can take place in the background and in this way can achieve real time operation. Often, features from different submaps are related using a branch and bound search based on joint compatibility [37]. This method has been applied on monocular cameras [38], as well as mechanically scanned imaging sonar [39].

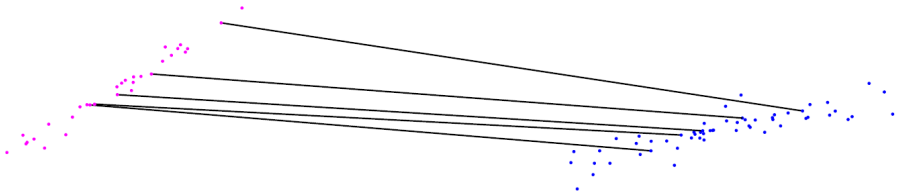


Figure 5-2: Loop closure by finding correspondences between common features in different submaps [35]. The red and blue dots represent features from two different submaps. Features that are matched to each other are indicated by the black lines.

5-2-2 Image to image matching

In the image space correspondences can be found by comparing the latest image to previously seen images, as shown in Figure 5-3. This matching is done by purely considering appearance information. As such scalability can become a issue as the number of stored images increases. Bag of words and vocabulary tree search methods are therefore often used as efficient means to look through the large image space [40]. These methods can retrieve images in a similar way as Google retrieves text or webpages. Such a model is used to detect loop closing in a SLAM system in [41]. A visual vocabulary is build based on SURF descriptors in images and is learned offline using training data.

While the accuracy and precision of this method is good and it can be run real time with efficient search methods, there are some downsides [35]. Only visual appearance information is considered and the spatial transformation is not given directly. Furthermore it requires offline learning of a good vocabulary.

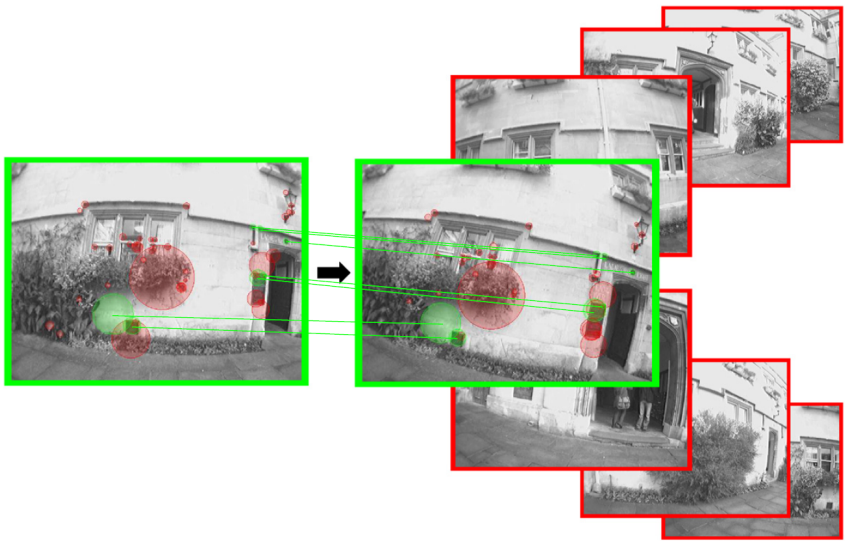


Figure 5-3: Loop closures are detected by finding correspondences between the current image and previously viewed images [35].

5-2-3 Image to map matching

The latest category combines the image and map space. Correspondences are sought between the latest image and features in the map, as shown in Figure 5-4. In [42] the authors use a relocalization module to determine the pose of the current view in relation to a map of point features. The pose is computed from feature point correspondences using RANSAC [25].

In a quantitative comparison, this method shows a high precision and fast real-time performance , but it requires a high amount of memory usage [35].

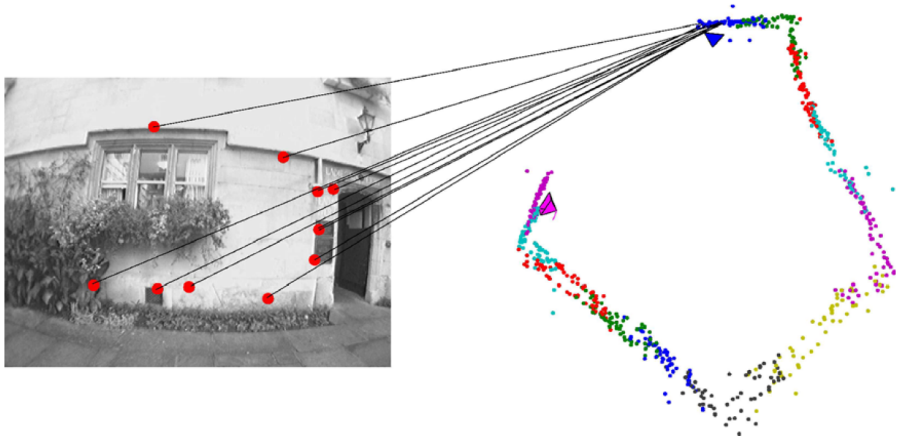


Figure 5-4: Loop closure detection by finding correspondences between the current image and map features [35].

5-2-4 Comparison

The discussed loop closure methods are compared against the non-functional requirements from Section 4-2. The analysis is made qualitatively using information and results from the previously discussed literature. The final comparison is shown in Table 5-2.

Working principle	Accuracy	Robustness	Computation time
Submap matching	+/−	− −	+
Image to image matching	+	+	−
Image to map matching	+	++	+

Table 5-2: Comparison of loop closure detection methods.

In the research of Wiliams et al. results show that submap matching has a poor performance, lacking the ability to reliably distinguish between true and false positives [35]. The other two methods that perform matching in the image space have a better performance in general. From these, image to image matching has the highest true positive rate and lowest computation time. However, this is at the cost of a higher memory usage.

5-3 Position and map estimation

Position and map estimation is considered as the back-end of a SLAM system. Many approaches exist for this problem and the optimal solution depends on the robot sensors, desired map resolution, nature of the environment and so on. In this section, the three main paradigms within the field of SLAM are discussed.

Recall that in SLAM, a robot uses all given information to simultaneously build a map of the environment and estimate its own location inside that map. There are two categories of the SLAM problem, full SLAM and online SLAM [43]. Full SLAM is involved in estimating the entire robot trajectory, while online SLAM instead only tries to recover the current robot pose. As full SLAM is solved in an offline manner, with all available data, it will not be further considered here, since it does not meet the stated requirements for the system.. The online SLAM problem is defined as

$$P(x_t, m|Z_t, U_t) \tag{5-3}$$

where x_t is the robot pose at time instant t, m is the true map of the environment, containing locations of objects and landmarks. Z_t is the sequence of observations, while U_t is the sequence of odometry measurements or control inputs.

The three main paradigms that are discussed in the subsequent sections are based on the Extended Kalman Filter (EKF), particle filter and an optimization based on a graphical representation. The paradigms are briefly discussed, listing the main benefits and downsides, for a more in-depth discussion the reader is referred to the work of Thrun, which explains the different algorithms and their application in SLAM [44].

5-3-1 Extended Kalman filter

The EKF is an extension of the popular Kalman filter, applied to non-linear state models [45]. Its application to the SLAM problem is one of the first reported solutions. Benefits of the EKF include an optimal mean-square error estimate and a strong convergence [46]. However due the nature of the covariance matrix, the computational effort increases quadratically with the scale of the map, making it unviable for large-scale environments (>100 landmarks).

Several variants of the EKF are developed to reduce the computational effort needed. Approaches that are based on the information form of the Kalman filter are shown to be more efficient. Sparse Extended Information Filter (SEIF) [47] and Exactly Sparse Extended Information Filter (ESEIF) [48] are sparse variants of the information form and have been applied on ship hull inspection [49]. Another strategy is divide the map into smaller submaps, which limits the cost associated with the covariance matrix [39]

5-3-2 Particle filter

The second category of solutions is based on the particle filter. Particle filters are non-parametric and able to handle non-linear state transitions [50] by representing the distribution

with a finite set of sample states or particles. They have become more popular in recent years and have been applied to the SLAM problem. Particle filters scale exponentially with the state space and as such are unsuitable for most SLAM applications. FastSLAM, however, alleviates this issue and makes particle filters suitable for SLAM [51].

The main advantage for FastSLAM is efficiency. It scales logarithmic to the size of the map and linear to the number of used particles. An improved version of FastSLAM was proposed in [52]. The so called FastSLAM 2.0 differs by taking into account the motion model and current observations to generate particles. While FastSLAM 2.0 is superior in most aspects, it has a more difficult implementation compared to FastSLAM 1.0, as it is more mathematically involved [53].

5-3-3 Graph-based optimization

This category of solutions is based on a graph formulation of the SLAM problem. This graph representation is shown in Figure 5-5. The nodes in the graph correspond to robot locations at different time instants. Edges, connecting the nodes, represent spatial constraints that are derived from measurements. Consecutive nodes are related by odometry measurements whereas other edges correspond to constraints that arise from multiple observations of the same landmark.

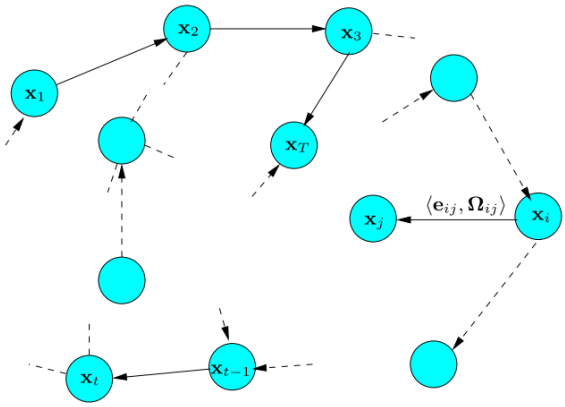


Figure 5-5: Graph representation of SLAM, where nodes correspond to robot locations at different time instants and edges are spatial constraints derived from odometry and observational measurements [54].

Let z_{ij} and Ω_{ij} be the mean and information matrix of a spatial transformation relating node i to node j. This can be obtained from odometry measurements or multiple observations of the same landmark. Furthermore, let $\hat{z}_{ij}$ be the predicted transformation, based on the relative transformation between the two nodes. Then, the error is defined as

$$e_{ij} = z_{ij} - \hat{z}_{ij} \tag{5-4}$$

The goal of the optimization is to find the configuration of nodes that minimize the negative log likelihood of all observations [54]:

$$x^* = \arg\min_x \sum_{<i,j>\in\mathcal{C}} e_{ij}^T \Omega_{ij} e_{ij} \tag{5-5}$$

where $\mathcal{C}$ is the set of all constraints. While most graph-based SLAM approaches are offline approaches, more recently iterative methods have emerged that are able to run the optimization in an online manner. Current iterative approaches include iSAM [55] and SLAM++ [56].

5-3-4 Comparison

The discussed pose and map estimation methods are compared using the non-functional requirements from Section 4-2. The comparison, shown in Table 5-3, is derived from the discussed literature in a qualitative manner.

Working principle	Accuracy	Robustness	Computation time
EKF	+/–	+/–	– –
Particle filter	+	+	+
Graph-based	+	++	–

Table 5-3: Comparison of pose and map estimation methods

It should be noted that all methods are viable and have been widely used in different applications, including sonar-based systems [7]. However, EKF methods have several disadvantages due to the linearization and high computational effort. Particle filter approaches are widely used due to their good performance and fast implementation. The graph-based have the potential to handle higher-dimension maps even better, but at the cost of computational costs, requiring an incremental method for online use.

5-4 Optimal working structure

The selection of an optimal working structure is done by choosing the optimal solution for each sub-function. This will be done in a qualitative manner, based on the goals, constraints and requirements. Recall from Chapter 1 the thesis objectives: development of a sonar-based underwater SLAM framework tailored to the working environment of ship hull cleaning robots. The final proposed design consists of the following working principles:

Visual odometry: Phase correlation

A view-based approach is chosen as it is suited for the feature sparse underwater environment. The Fourier-based phase correlation shows promising results on FLS images. The major downside of view-based approaches, identification of only similarity transformations, is not an issue with the simplified sonar geometry model.

Loop closure detection: Object-based image to map matching

For loop closure detection, an approach based on visible objects is chosen. While not many distinct landmarks are present on the ship, gratings and holes provide an orientation point. To detect previously visited locations, the image to map method is chosen as the most optimal. This method showed the best accuracy and precision in literature, while also being computationally efficient.

Pose and map estimation: Particle filter

Graph-based SLAM and particle filter methods both show a good performance. In the end, the particle filter method is chosen as the optimal solution for its fast and efficient implementation and provides a good starting point for further development. However, note that all methods are generally suitable for a SLAM system.

5-5 Overview

A final overview of all discussed methods in this chapter is shown in Table 5-4.

Subsystem	Working principle	Note
Visual odometry	Pixel-level feature registration	Unreliable on sonar imagery.
	Region-level feature registration	Low stability and repeatability.
	Spatial correlation	Unreliable peak detection and high computation time.
	Phase correlation	Best accuracy and robustness.
Loop closure detection	Submap matching	Unreliable in distinguishing true and false positives.
	Image to image matching	Good performance, high computation time.
	Image to map matching	Best performance, high memory usage.
Pose and map estimation	Kalman filter	Linearization errors and high computation time.
	Particle filter	Good performance, fast and efficient.
	Graph optimization	Growth in computation time in larger maps.

Table 5-4: Overview of the discussed working principles in this chapter for each sub-function.

Chapter 6

Sonar-based visual odometry

In the previous chapters, the conceptual design of a sonar-based Simultaneous Localization and Mapping (SLAM) system for the Fleet Cleaner robot was described. As detailed in Section 3-3, long segments without distinct landmarks are expected when driving on the hull. To reduce the error build-up on these segments, visual odometry is of importance. The implementation of a visual odometry approach is the first step towards the second thesis objective: *validate the sonar-based visual odometry algorithm with a proof of concept.*

As explained in Section 5-1, the phase correlation method was chosen as the most optimal method for sonar-based visual odometry. While this method has been applied on other domains such as optical imagery, its application on sonar imagery is not widespread. Therefore, the implementation phase will consist of tailoring the method to the specific limitations of sonar imagery. The chapter will detail several image processing steps necessary to improve the method, as well as the tuning of parameters in these operations.

In Section 6-1 the basic principles of the phase correlation method are explained. The implementation of different sub-functions, such as image processing operations, are discussed in Section 6-2, with the tuning of parameter values in Section 6-3. Finally, a discussion of the findings and limitations is presented in Section 6-4.

6-1 Phase correlation

Fourier-based registration is based on the Fourier shift property, which states that a shift between two functions is transformed into a linear phase shift in the Fourier domain. Consider two images $i_1(x, y)$ and $i_2(x, y)$, which are related by a translational shift (t_x, t_y):

$$i_1(x, y) = i_2(x - t_x, y - t_y) \tag{6-1}$$

Applying 2D Fourier transformation then yields the following relation, according to the Fourier shift property:

$$I_1(u,v) = I_2(u,v)e^{-j(ut_x+vt_y)} \tag{6-2}$$

Where I_1 and I_2 are the 2D Fourier transforms of the pair of images. The phase term can then be factored out, which results in the normalized cross power spectrum:

$$C(u,v) = \frac{I_1(u,v)I_2^*(u,v)}{|I_1(u,v)I_2^*(u,v)|} = e^{-j(ut_x+vt_y)} \tag{6-3}$$

Where I^* is the complex conjugate of I. This equation can be solved for (t_x, t_y) in the frequency domain, or more commonly in the time domain. Applying the Inverse Fast Fourier Transform (IFFT) to equation 6-3 yields the Phase Correlation Surface (PCS):

$$c(x,y) = \delta(x - t_x, y - t_y) \tag{6-4}$$

Since this an impulse function centered on (t_x, t_y), this leads directly to the identification of displacement. Although the presence of image noise and other disturbances may degrade this function, the displacement may be retrieved as long as it contains a dominant peak. The complete process is summarized in Figure 6-1. From the sonar displacements, the robot velocity can be derived. This is further detailed in Appendix B.

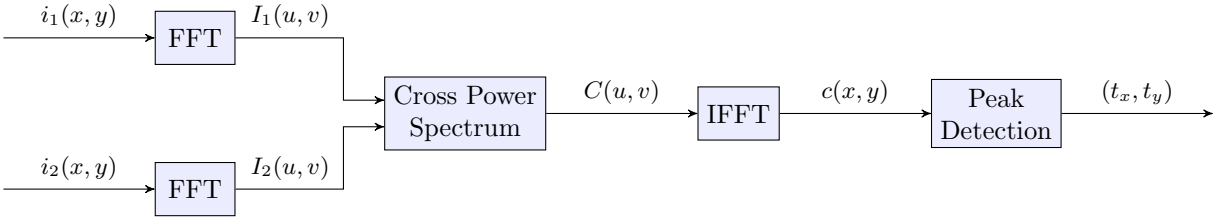
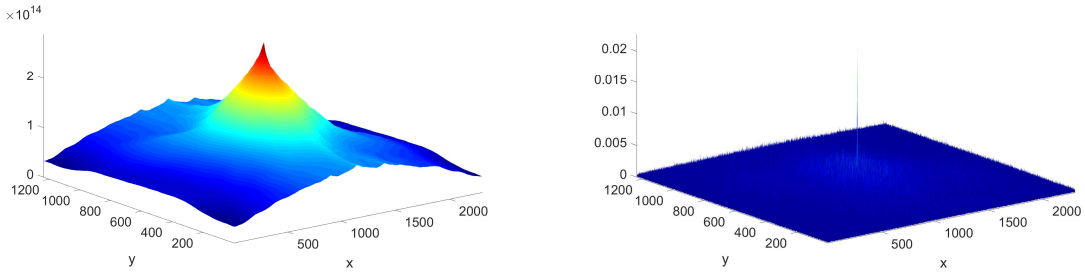


Figure 6-1: Diagram describing the phase correlation registration process of two input images to find the displacement (t_x, t_y).

Compared to classical correlation methods, the phase correlation method shows an advantage in the accuracy of peak detection. As shown in Figure 6-2, the phase correlation shows a very sharp and distinct peak, when compared to standard cross correlation. Furthermore, the cross power spectrum is normalized by the denominator, meaning that all frequency components have unity amplitude. This operation is equivalent to pre-whitening of signals, which makes the phase correlation robust to noise types that are correlated to the image function, such as uniform variations of illumination or offsets in average intensity [57].



(a) Standard cross correlation **(b)** Phase correlation

Figure 6-2: Phase correlation surface shows a more clear and distinct peak when compared to standard cross correlation.

6-2 Implementation

In the previous section, the basic principles of a phase correlation registration method were discussed. In practice, however, there can be some problems in reliably detecting a peak in the PCS. In this section, several operations are discussed that aim to improve the quality of peak detection.

Methods to detect sub-pixel displacements are discussed in Section 6-2-1. In Sections 6-2-2 and 6-2-3 masking and filtering operations on the sonar images are detailed, while filtering operations on the cross power spectrum are discussed in Section 6-2-4. Finally, in Section 6-2-5 methods are discussed that can estimate the rotational displacement instead of the translational displacement.

6-2-1 Peak detection

The dominant peak in the PCS can be found simply by examining the maximum value:

$$(t_x, t_y) = \arg\max_{x,y} |c(x,y)| \tag{6-5}$$

However, this yields only integer offsets and does not take into account the shape of the peak when the surface is not an ideal Dirac function due to noise and disturbances. The typical forward velocity of the robot is roughly $0.3\,\mathrm{m\,s^{-1}}$, meaning that with 10 Hz the robot displacement is 3 cm between every update. In the sonar image view this displacement translates to roughly 3.75 pixels. The small pixel displacements lead to large inaccuracies when only integer offsets are considered. Several methods exist to find sub-pixel displacement in the PCS. According to [57], $c(x,y)$ can be approximated as:

$$c(x,y) = \frac{\sin(\pi(x-t_x))}{\pi(x-t_x)} \frac{\sin(\pi(y-t_y))}{\pi(y-t_y)} + n(x,y) \tag{6-6}$$

Where $n(x, y)$ refers to interference terms, including noise. If this term is negligible due to high SNR, they propose sub-pixel displacement can be found by:

$$t_x = \frac{c(1,0)}{c(1,0) \pm c(0,0)} \qquad t_y = \frac{c(0,1)}{c(0,1) \pm c(0,0)} \tag{6-7}$$

In this equation the highest peak is located at (x_0, y_0) and the notation $c(k, l) = c(x_0+k, y_0+l)$ is used as a simplification. Sonar images, however, feature low SNR and as shown in [58] this method is inaccurate in the presence of interference terms. They instead propose a method that takes into account the difference between the two-sided neighbors:

$$D_x = c(1,0) - c(-1,0) \qquad D_y = c(0,1) - c(0,-1) \tag{6-8}$$

$$t_x = \frac{D_x}{c(0,0) + |D_x|} \qquad t_y = \frac{D_y}{c(0,0) + |D_y|} \tag{6-9}$$

The authors show that this approach is more robust in the presence of noise and leakage to side peaks, which is very common, especially in sonar imagery. An example of a phase correlation surface with side peaks is shown in Figure 6-3. As such, this method for sub-pixel peak detection will be used in the subsequent sections of the report.

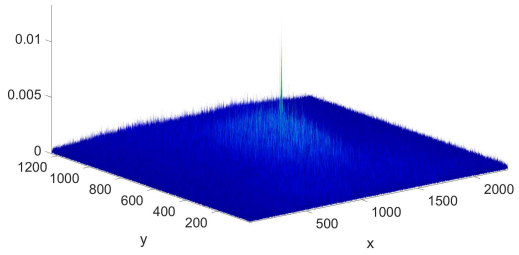


Figure 6-3: Phase correlation surface with a main peak and multiple side peaks.

6-2-2 Windowing operations

Some degrading effects occur due to the computation process of the Fourier transform. The Fast Fourier Transform (FFT) algorithm approaches the continuous Fourier transform by assuming that the finite-length signals are periodic in nature. However, the considered signals are not periodic in nature and this results in rough transitions when a cyclic repetition is imposed at the end and beginning of such a cycle. These transitions result in undesired frequency components appearing in the spectrum, which is known as spectral leakage. On 1D signals it is typical to perform a windowing operation on the signal before the FFT computation. These windows operations reduce the amplitude of points at the start and end of the signal, smoothing out the effects of spectral leakage. Often used windows include Gaussian, Hamming and Hann functions.

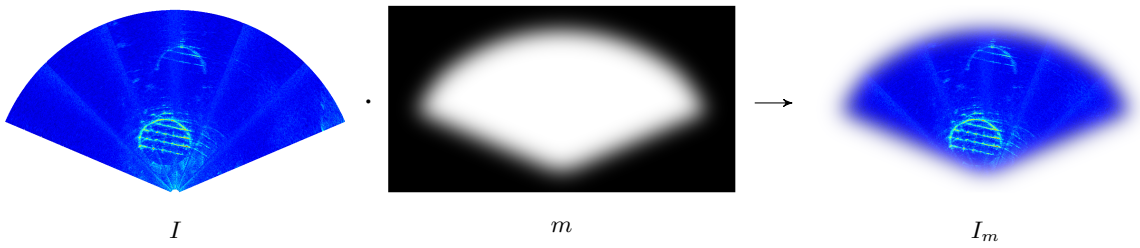
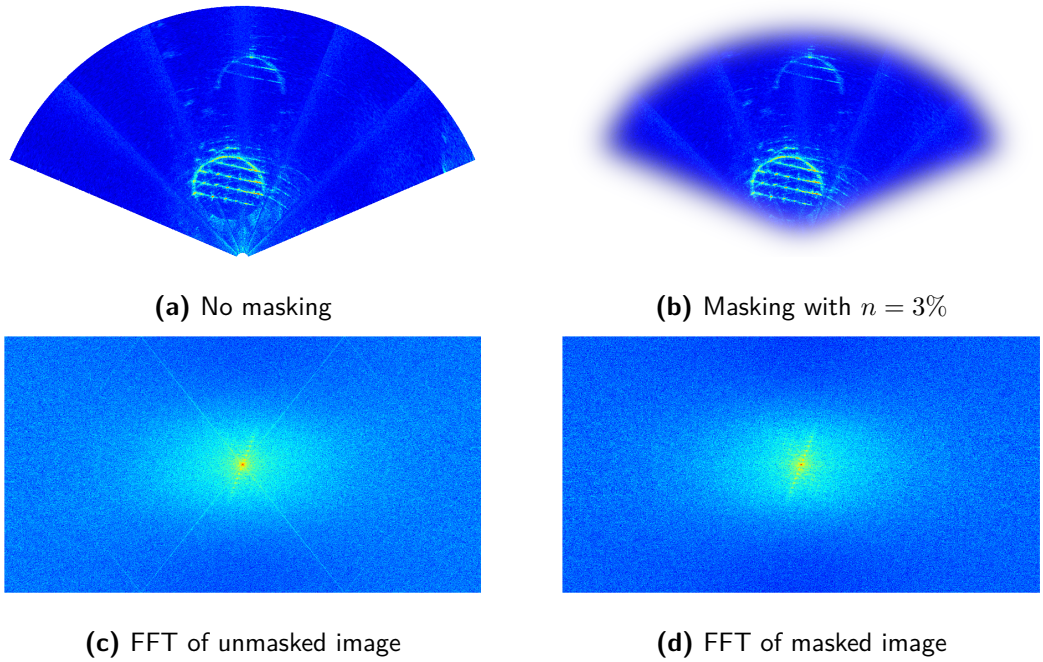


Figure 6-4: Fan-shaped edge mask applied on a sonar image in Cartesian coordinates.

These effects are similarly present in the 2D FFT applied on images, as can be seen in Figure 6-5. However, the Cartesian sonar images feature a fan shaped image frame instead of a rectangular frame. Hurtós describes the following procedure to create a mask for the Cartesian sonar images to reduce the effects of spectral leakage [18]:

1. Compute the footprint f of the input sonar image.
2. Apply a shrink operation to f with n pixels, resulting in f_s.
3. Apply a Gaussian filter k to f_s with standard deviation $\sigma = n$ and size $6n \times 6n$.
4. Convolve $f_s * k$, obtaining mask m.
5. Apply the mask to the input image: $I_m = I \cdot m$.



(a) No masking **(b)** Masking with $n = 3\%$

(c) FFT of unmasked image **(d)** FFT of masked image

Figure 6-5: The FFT of the unmasked image shows patterns due to the edge effects, most notable in the diagonal lines. By applying a mask on the image, some of these effects are reduced and patterns related to the image content gain more relevance.

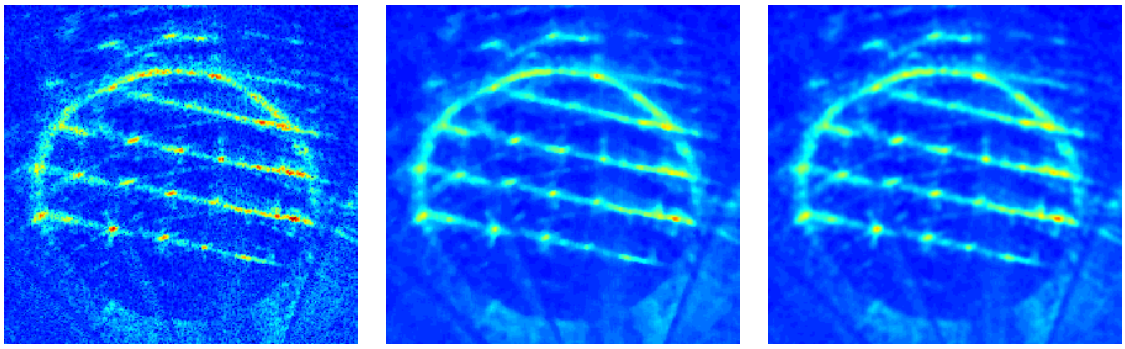
6-2-3 Image filtering

To reduce the effect of unwanted frequencies that lead to a noisy phase correlation matrix, it is common to apply a filtering step to the sonar images before further processing. The predominant type of noise present in sonar imagery is speckle noise. The main difference to white noise is that speckle noise is assumed to have a multiplicative error model:

$$I_n(x, y) = I(x, y) \cdot n(x, y) \tag{6-10}$$

Several filtering operations exist that can reduce the amount of speckle noise. Two different categories of filters are distinguished, i.e. adaptive and non-adaptive. Adaptive filters use a kernel with parameters that are changed with respect to local image properties. Therefore, these filters in general feature a better reduction of speckle noise than non-adaptive filters. The improved performance is at the cost of increased computation time as most of the adaptive filters are not optimized for real time use.

Different speckle noise reduction filters were compared in [59] and [60] by measuring metrics such as MSE, SNR and ENL. In both comparisons, adaptive filters performed consistently better than non-adaptive filters. From the adaptive filters, the Frost filter [61] was found to have the best performance in all metrics except for edge preservation. Implementation of the Frost filter is provided in Appendix C-2-2. In addition, the median filter is considered too, as it shows good results for a non-adaptive filter and its computation time is considerably faster than the Frost filter.



(a) Original **(b)** Median filter **(c)** Frost filter

Figure 6-6: Median filter and Frost filter applied to reduce speckle noise effects.

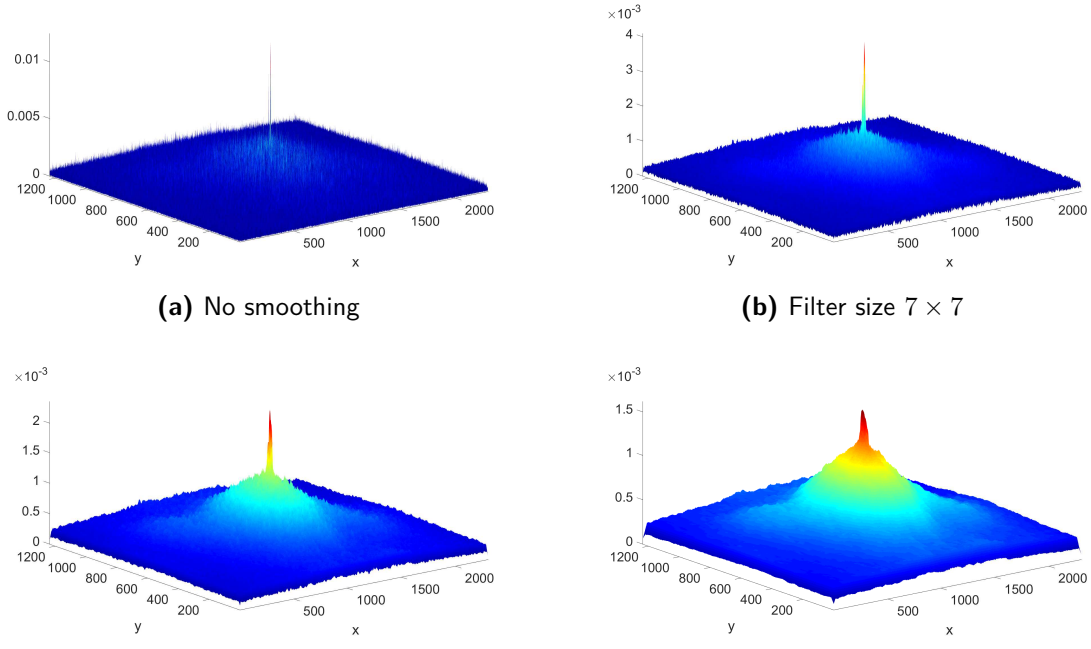
Both filters are applied to the sonar image in Figure 6-6. The Frost filter produces the best resulting, smoothing out the noise while the image content is a little sharper compared to the median filter. However the computation time of the Frost filter on one image is more than 60 s, obstructing the requirement to run the algorithm in real time. Therefore, the median filter will be selected which is considerably faster at the expense of lost sharpness in the images.

6-2-4 Phase correlation filtering

In the foregoing sections, filtering and masking operations were discussed that are applied directly on the input images. With these operations it is still possible to have unwanted frequencies in the cross power spectrum, originating from e.g. noise. Therefore filtering operations applied on the cross power spectrum become necessary as well. Two main approaches to filter out the noise are identified, directly in the frequency domain or in the spatial domain, after the IFFT is applied.

Filtering in spatial domain

The most straightforward approach is to smooth out the noise in the correlation surface. This can be done in a simple way by, for example, applying an averaging filter. However, for these filters selecting the right kernel size is critical. A wrong peak may be selected when the kernel size is too small, while data may be smeared when the filter size is too large, losing significant accuracy. Figure 6-7 shows the effect of various filter sizes on the peak of the surface. Furthermore, the appropriate size is heavily dependent on the specific surface and may vary greatly between different registrations.



(a) No smoothing

(b) Filter size 7×7

(c) Filter size 15×15

(d) Filter size 33×33

Figure 6-7: The effect of averaging filter size on the peak of the phase correlation surface.

Filtering in frequency domain

Hurtós [18] has looked instead at filters in the frequency domain that may be adapted to the specific properties of each registration. The frequencies that are most likely to introduce

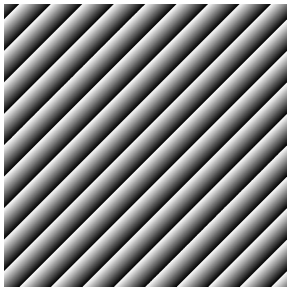
problems are the higher frequencies. The intuition is then to use a simple low-pass filter, such as a Butterworth filter. As in the spatial case, the problem here is to tune the filter parameters to the specific situation. The Butterworth filter is described as:

$$H(u,v) = \frac{1}{1+(\frac{r}{f_c})^{2k}}, u = 1,2,..,M, v = 1,2,...N \tag{6-11}$$

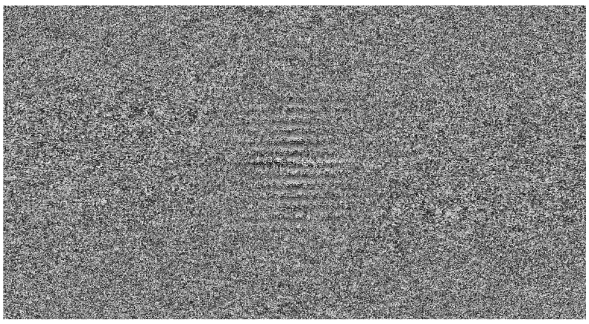
Where $r = \sqrt{m^2+n^2}$ is the frequency radius in normalized coordinates:

$$m = \frac{u-\frac{M}{2}}{M} \qquad n = \frac{v-\frac{N}{2}}{N} \tag{6-12}$$

Furthermore, k is the order of the filter and f_c is the normalized cutoff frequency in the range $(0, 0.5]$. These filter parameters are again dependent on the image content and finding a static value for different image pairs can be a hard task. Stones et al. propose to filter out frequencies outside of a range from the frequency origin [62]. They find a suitable range to be $0.6N/2$, where N is the minimum of samples in the x and y direction of the image.



(a) Phase difference matrix of a 2D delta function.



(b) Phase difference matrix of a pair of sonar images.

Figure 6-8: The sawtooth pattern is visible in the phase difference matrix of a 2D delta function (a). In the phase difference matrix of a pair of sonar images, the pattern is visible in an area around the origin (b).

However, this static approach may not be robust to different image pairs with different image content and frequency responses. A more adaptive approach is established by Hurtós. The basic principle behind this approach will be summarized here. As shown in [63], the phase difference matrix of a pair of images correspond to a sawtooth pattern. Figure 6-8a shows the phase difference matrix of a pure delta, being the ideal scenario without noise. The period of this sawtooth pattern along each axis correspond to the shift along that same axis. However, the sonar images produce a phase difference matrix in which this sawtooth pattern is not clearly visible due to the presence of noise in the images. As shown in Figure 6-8b there is an area around the origin where the sawtooth pattern is visible. The intuition is that the frequencies inside that area contribute to the correlation peak, while frequencies outside the area correspond to noise and can be filtered out. A watershed segmentation algorithm [64] is used by Hurtós to find the range of this area and the corresponding cutoff frequency, see Figure 6-9. The MATLAB method `grayconnected` is used here instead for its simplicity and efficiency. The implementation of this function is provided in Appendix C-2-3. This method uses a tolerance value, which is tuned in Section 6-3.

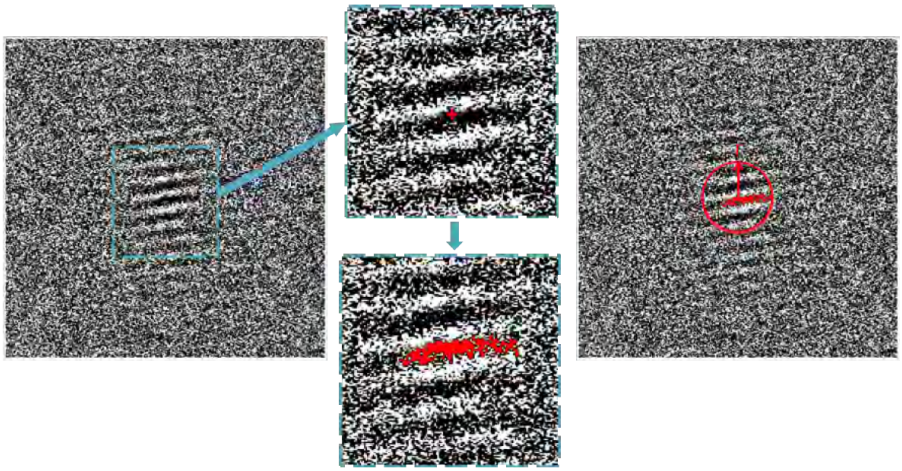


Figure 6-9: A segmentation algorithm is used from the origin, indicated by the red area. The radius of this area is used as an indication for the cutoff frequency [18].

6-2-5 Rotation estimation

The phase correlation method discussed up until this point is used to estimate only translational shifts between a pair of images. Since the robot is able to rotate using a steering wheel, rotational shifts between images are expected as well. Except for driving in a straight line, all movement of the robot is a combination of translations and rotations. Two images related by a rotation θ_0 and translation (t_x, t_y) can be expressed by:

$$i_1(x, y) = i_2(x \cos\theta_0 + y \sin\theta_0 - t_x, -x \sin\theta_0 + y \cos\theta_0 - t_y) \tag{6-13}$$

The aim is to find θ_0 and correct the rotation between images so that the previous discussed phase correlation method can be used to find the translational shifts, as shown in Figure 6-10. Several approaches to find the rotational shifts are discussed here.

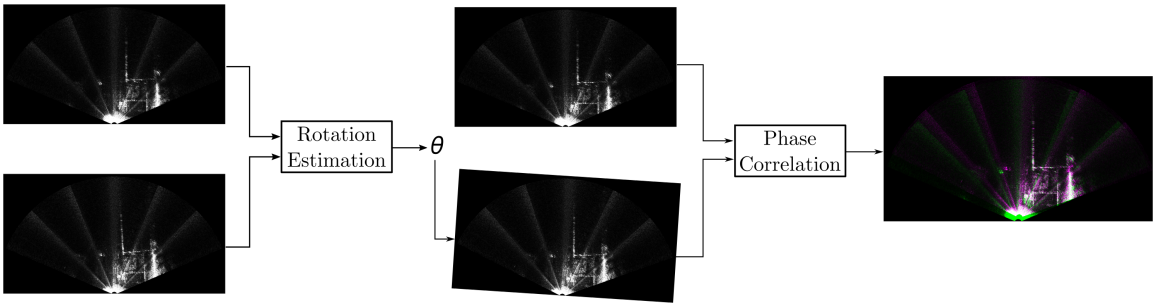


Figure 6-10: One of the images is compensated for rotation, so that only a translational displacement remains, which can be estimated with the phase correlation method.

A brute force approach is possible by finding the angle θ that yields the maximum correlation between i_1 and i_2. However, this requires computation of the phase correlation for each angle

hypothesis, making it unviable to perform in real time. Therefore, this brute force method will not be considered here.

Using IMU measurement to estimate the rotations is a different approach using already available data without further computations. While an absolute measurement of yaw is possible on the side of the ship, this measurement is unreferenced on the bottom of the ship. Therefore, methods that estimate the rotations using the FLS images are considered first as they can be used to extend the current yaw measurements.

Fourier-Mellin transform

This method is an extension of the phase correlation method and is used to estimate rotation and scale changes using the log-polar domain [65]. A simplified explanation of the method is given here. Using the Fourier translation and rotation property, Equation 6-13 is expressed in the Fourier domain as

$$I_1(u,v) = I_2(u\cos\theta_0 + v\sin\theta_0, -u\sin\theta_0 + v\cos\theta_0)e^{-j(ut_x+vt_y)} \tag{6-14}$$

As stated before the translational displacement only affects the phase spectrum, so the magnitudes of these Fourier transforms are given as:

$$|I_1(u,v)| = |I_2(u\cos\theta_0 + v\sin\theta_0, -u\sin\theta_0 + v\cos\theta_0)| \tag{6-15}$$

This equation shows that both magnitudes are the same, but with a rotation applied to the second one. This rotation can be expressed as a linear translation in polar coordinates:

$$|I_1(r,\theta)| = |I_2(r,\theta-\theta_0)| \tag{6-16}$$

From this θ_0 can be found using the phase correlation method. In this case the phase correlation is not applied on the original images, but on the magnitude of the polar Fourier transform which have a low structural nature. Furthermore, these images degrade from the conversion to polar coordinates as the Cartesian images are sampled to obtain the polar grid. This is especially apparent in the lower frequencies, which may require the use of a high-pass filter.

Direct polar estimation

Problems with the Fourier-Mellin transformation are caused by using magnitude of polar Fourier transformed images as input for the phase correlation. The transformation is numerically intensive and prone to inaccurate estimations.

Remember that the FLS images are actually recorded in the polar domain, capturing range and azimuth angle, and then transformed to Cartesian coordinates. Up until this point phase

correlation has been applied to the Cartesian images to find the translational displacement. However, when phase correlation is applied directly to the polar images, the rotational displacement can be recovered. While this method is viable in the presence of pure rotations, some inaccuracies may arise in the case of combined rotations and translations. There is a difference between the sonar origin and the actual center of rotation, which can be accounted for by recomputing the polar images. Furthermore, in the polar domain rotations are not decoupled from the translational shifts, i.e. a pure translational movement will result in a change in both range and azimuth angle in the polar domain. Hurtós notes that when the translations are small compared to the image size, the distortions in the polar images are small enough to allow for the recovery of the rotational displacement. The high frame rate of the FLS ensures large overlaps and thus small translations between consecutive frames, reducing significant errors by this effect.

In [66], Hurtós et al. compare different methods for estimating rotation on FLS images, including the Fourier-Mellin transform and direct polar estimation. They show that direct polar estimation has a better accuracy in the case of pure rotations. When introducing translations in combination with rotations the quality degrades and the observed accuracy is similar or slightly worse than the other methods. However, the direct polar estimation is considerably faster than other methods, facilitating the desired real time use of the method. Because of the faster computation time, similar accuracy and simplicity of this method, direct polar estimation will be used in the remainder of this report as the method of choice to estimate rotational displacements.

6-3 Parameters

In the previous section, the used filter and processing steps were detailed. The process is summarized in Figure 6-11. As the quality of the final peak detection step is dependent heavily on these steps, proper tuning of the filters is necessary. The parameters that are considered are summarized in Table 6-1.

Subfunction	Parameter	Remarks
Input	n_f	FLS imagery is used as input. Every n_f-th frame is analyzed.
Edge mask	n_m	Edge mask size as a percentage of the maximum image size.
Image filter	S_{med}	Median filter size.
Phase correlation filter	T_{flood}	Butterworth filter with cut-off frequency f_c and order k_{bw}. f_c is estimated with a flood fill algorithm with tolerance T_{flood}.
	k_{bw}	
Rotation estimation		Previous filter and masking operations are applied on polar images before phase correlation. Different values are used on the polar images.

Table 6-1: Overview of parameters

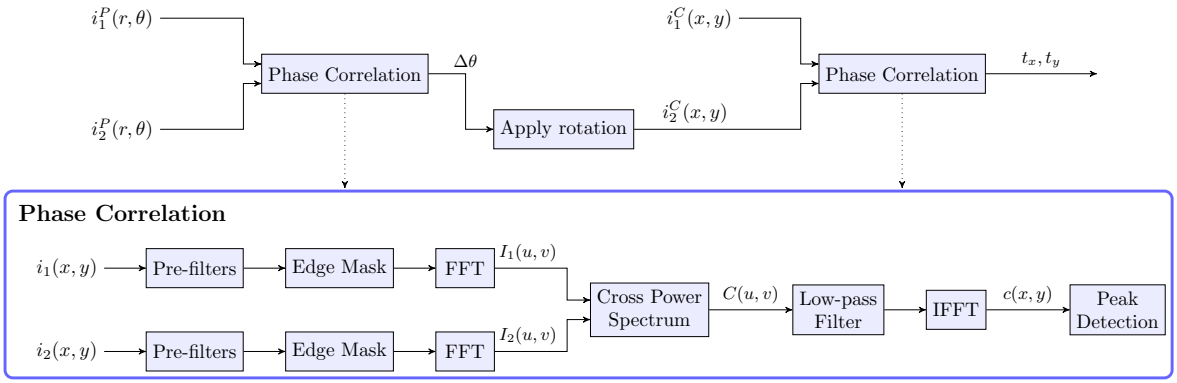


Figure 6-11: Overall pipeline for the phase correlation method. Phase correlation is used twice, first to estimate rotation and then to estimate translation on the rotation compensated images.

Validation datasets

From a multitude of datasets different pairs of consecutive images were selected as validation datasets to tune the parameters. The datasets include varying environmental circumstances summarized in Table 6-2. The datasets do not include an accurate ground truth measurement and as such a performance metric for peak detection is used to qualify the registration process.

Dataset	Ship	Remarks
1	Mineral China	Visible weld line, fouling
2	OOCL Europe	Reflections
3	Tosca	Reflections, multiple weld lines
4	Fagelgracht	Fouling, weld lines
5	Cosco Pride	Multiple gratings and weld lines

Table 6-2: Datasets used for tuning, from each set a pair of images is chosen.

Performance metrics

The performance of peak detection is quantified using the Peak to Sidelobe Ratio (PSR) metric:

$$\text{PSR} = \frac{peak - \mu}{\sigma} \tag{6-17}$$

This metric essentially describes how many standard deviations the peak is above the average. A high distinct peak indicates a high correlation, whereas leakage to side peaks can indicate erroneous correlation caused by noise or spectral leakage. The intuition of the PSR metric is that lower values indicate leakage to side peaks, degrading the registration process.

6-3-1 Input

As input the Forward Looking Sonar (FLS) images are used. The Cartesian representation is used to estimate the translational displacement and the polar representation to estimate the rotations, as explained in Section 6-2-5. The other important factor to consider is the frequency of the used input images. Fleet Cleaner employs a FLS with an update rate of roughly 10 Hz, while the required update rate of the algorithm is 3 Hz (as stated in Section 4-2). Instead of analyzing every frame, one can choose to analyze every 2nd (5 Hz) or 3th (3 Hz) frame.

Dataset	PSR		
	$n_f = 1$	$n_f = 2$	$n_f = 3$
1	400.6	210.6	153.3
2	73.8	50.8	37.7
3	99.1	73.1	48.0
4	136.2	92.3	83.2
5	149.2	75.5	53.5

Table 6-3: PSR rating of image pairs are compared, varying the update rates from $n_f = 1$ to $n_f = 3$.

In Table 6-3 it is shown that higher update rates improve the performance of registration. This can be explained by the higher overlap between images and therefore more correlation. However, this does not inherently mean that higher update rates lead to higher accuracy of the registration. A high update rate leads to smaller pixel displacements between images. Relative to the small pixel displacement, the sub-pixel peak estimation becomes more important. However as explained before in Section 6-2-1, the sub-pixel estimation is not without inaccuracies. To minimize the relative error in the sub-pixel estimation, the update rate can be lowered, such that a higher pixel displacement is observed. Furthermore a lower update rate allows more time per calculation, which is not a trivial advantage as the filtering and masking operations all increase the required computational effort. Therefore, n_f is set to 3, despite the improved performance at lower values.

6-3-2 Edge masks

Edge masks are applied to the datasets with different mask sizes in the range $n_m = 1 - 4\%$, expressed in a percentage of the max image size. The resulting PSR metric for registration is shown in Table 6-4. For most datasets (dataset 2 being the exception), a larger mask size results in a higher PSR metric. However, larger mask sizes may remove artifacts on the edges of the window, so a too large mask size is not desirable as well. Mask size $n_m = 3\%$ was chosen as the best balance between these priorities.

Dataset	PSR				
	$n_m = 0\%$	$n_m = 1\%$	$n_m = 2\%$	$n_m = 3\%$	$n_m = 4\%$
1	153.3	172.1	192.7	218.8	235.2
2	37.7	40.8	39.7	38.8	36.7
3	48.0	55.5	62.6	76.4	93.3
4	83.2	95.3	99.2	104.5	109.1
5	53.5	57.9	63.0	72.8	81.4

Table 6-4: PSR metric for the image pairs with different edge mask sizes in the range $n_=1-4\%$

6-3-3 Image filter

Prior to the edge mask operation, the images are adjusted with a median filter to reduce the amount of speckle noise in the images. The filter is tuned by varying the filter size S_{med}. A too large filter size reduces sharpness and detail in the image. For this reason the filter size is tested for the odd-number values in the range $1 - 15$. The max PSR value in this range is shown in Table 6-5. In the tested datasets there is no clear value that is suitable for all datasets. The preference is for smaller filter sizes as more detail is preserved in the images. Therefore, a filter size of $S_{med} = 5$ was chosen.

Dataset	PSR(no filter)	Max PSR	S_{med}
1	218.8	-	1
2	38.8	44.0	3
3	76.4	89.0	5
4	104.5	149.5	5
5	72.8	101.2	15

Table 6-5: Max PSR metric for different median filter sizes in the range $S_{med} = 1 - 15$

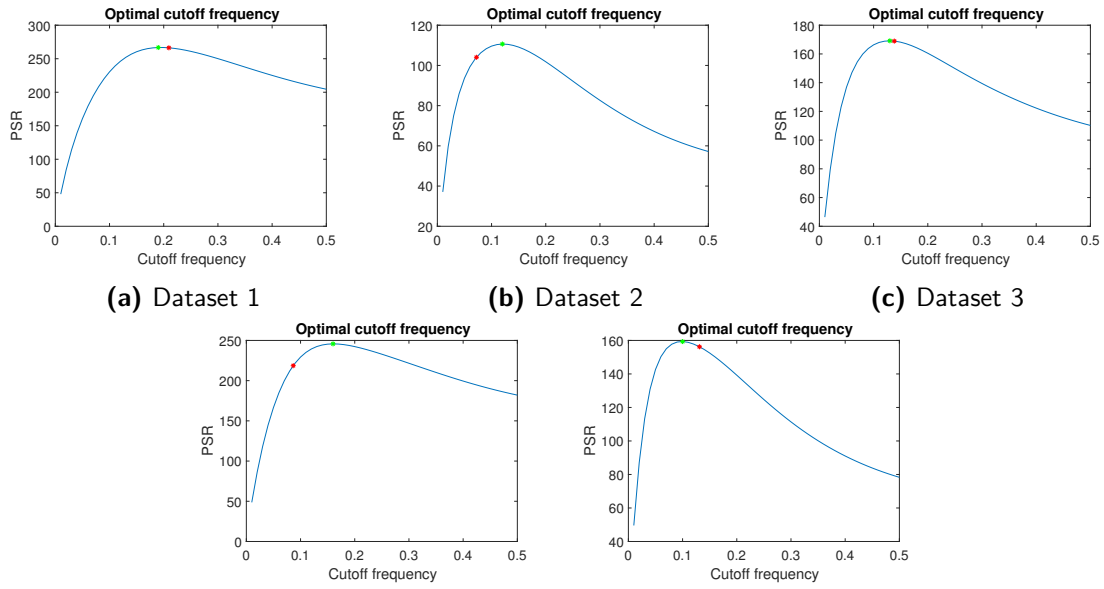
6-3-4 Correlation filter

The cutoff frequency f_c and filter order k_{bw} are the parameters that impact the performance of the Butterworth filter. The cutoff frequency is estimated by using a flood fill algorithm with a tolerance T_{flood}. The filter order is set to $k_{bw} = 2$, which showed the highest PSR for all datasets. The flood tolerance is tested in the range 0.1 to 0.3 in increments of 0.01. The tolerance values for which the PSR is maximal, are shown in Table 6-6. While the variance in cutoff frequency is significant, the optimal tolerance values are all close to each other, varying between 0.23 and 0.26. The average value of 0.24 is chosen as the flood tolerance in the remainder of the report.

Dataset	PSR(no filter)	Max PSR	T_{flood}	f_c	$f_{c,opt}$
1	167.0	266.6	0.23	0.19	0.19
2	42.4	109.8	0.25	0.14	0.12
3	89.0	169.1	0.23	0.14	0.13
4	149.5	242.9	0.26	0.13	0.16
5	59.3	159.3	0.23	0.09	0.1

Table 6-6: Max PSR metric for different tolerance values in the range $T_{flood} = 0.1$-0.3

A comparison is made between the cutoff frequency estimation resulting from the flood fill algorithm and the optimal cutoff frequency. The optimal cutoff frequency is calculated by testing values in the range 0.01 to 0.5 in increments of 0.01. The results are shown in Figure 6-12, where the green dots represent the optimal cutoff frequency and the red dots the estimated cutoff frequency. It can be seen that the PSR metric of the estimated cutoff frequency is close to the optimal value in most cases.



(a) Dataset 1 **(b)** Dataset 2 **(c)** Dataset 3

(d) Dataset 4 **(e)** Dataset 5

Figure 6-12: PSR metric for cutoff frequencies in the range $f_c = 0.01 - 0.5$. The optimal cutoff frequency is indicated by a green dot, whereas the cutoff frequency estimated with the flood fill algorithm is indicated by a red dot.

6-3-5 Rotation estimation

The same procedure is repeated for the images in polar coordinates. The methods to estimate the parameter values are same as above and only the results are shown here, summarized in Table 6-1. Differences with the Cartesian images are found in the median filter and Butterworth filter. The median filter seems to have a more degrading effect on performance, so the filter size will be set to $S_{med} = 3$ for rotation estimation. The Butterworth filter produces similar results, however with a slight change for the flood fill tolerance value.

6-3-6 Overview

Final parameter values are summarized in Table 6-7

Subfunction	Parameter	Value (Cartesian images)	Value (Polar images)
Edge mask	n_m	3%	3%
Image filter	S_{med}	5	3
Phase correlation filter	T_{flood}	0.24	0.20
	k_{bw}	2	2

Table 6-7: Final parameter values for the images in Cartesian coordinates, to estimate translations, and polar coordinates, to estimate rotations.

6-4 Discussion

In the preceding sections the underlying principles and implementation of the Fourier-based registration method were detailed. Here, the main findings are discussed, as well as limitations of the proposed method and implementation.

6-4-1 Findings

The Fourier-based phase correlation method was presented as a viable alternative to the more widely used feature-based registration approaches. There are various advantages when applied specifically on sonar imagery:

- Correlation can be found in areas with minimal details and distinct features.
- Because of the normalization in the method, the approach offers some robustness to noise, illumination changes and occlusions [57].

However, even with these advantages, some operations are needed to improve the quality of peak detection in the correlation surface. In order to reduce the edge effects, caused by the FFT, masks are applied to the images. In the case of sonar images in Cartesian coordinates,

a fan-shaped mask is used, instead of a rectangular shape. Furthermore, speckle noise is reduced by applying a filter directly on the sonar images. The Frost filter, though having better performance, is turned down due to the heavy computational costs. A simple median filter is considered instead. For both the median filter and edge masks, a trade-off needs to be made between reducing negative effects and maintaining enough details in the images.

A final operation is introduced on the phase correlation matrix in order to sharpen the main peak and reduce the side-peaks in the surface. This is done in the frequency domain with a low-pass filter. The cut-off frequency is estimated depending on the structure of the phase matrix, such that it is adaptive to the image content.

The method is further extended to estimate rotational displacements in addition to translational displacements by applying the same phase correlation method on the sonar images in polar coordinates. The same operations are applied on the polar images.

6-4-2 Limitations

However, the proposed method is subjected to the following limitations:

- The phase correlation method is unable to handle complex transformations. Only translational displacements can be recovered and with extension rotations. However, this falls within the simplified FLS geometry model, introduced in Section 2-3.
- The simple median filter, used to reduce speckle noise, is non-adaptive. The optimal filter size varies heavily between different images and a suitable value for different situations was not found. More complex filters however are not able to fulfill the real-time requirement of the algorithm.
- Rotations estimated by the phase correlation are inherently susceptible to inaccuracies. Rotation is not decoupled from translations in the polar representation, creating distortions in the presence of combined rotations and translations.

Chapter 7

Experiments and results

In Chapter 6, the final implementation of the proposed Fourier-based visual odometry method was detailed. This implementation should work under the operating conditions, described in Chapter 3 and achieve the required performance, proposed in Chapter 4. In this chapter, the algorithm is tested using data collected during cleaning operations.

The performance in several cases is considered and compared to the current positioning system. The most typical scenario is the robot driving in a straight line. Furthermore, rotation in combination with translation is considered. Finally, a comparison is made in the case of wheel slip, which is a major cause of error build-up in the current setup. The performance is measured according to the non-functional requirements, discussed in Section 4-2.

Section 7-1 describes an experiment that was conducted during cleaning of the Pioneering Spirit, the worlds largest construction vessel. During this experiment the robot is driving from a known start point to a known ending location. The performance is compared to the true known traveled distance and the estimate by the wheel encoders. In Section 7-2 the rotational displacement is considered and compared against the IMU measurements. In Section 7-3, a comparison is made against the wheel encoders in the case of wheel slip. Finally, the findings are summarized and discussed in Section 7-4.

7-1 Weld line to weld line

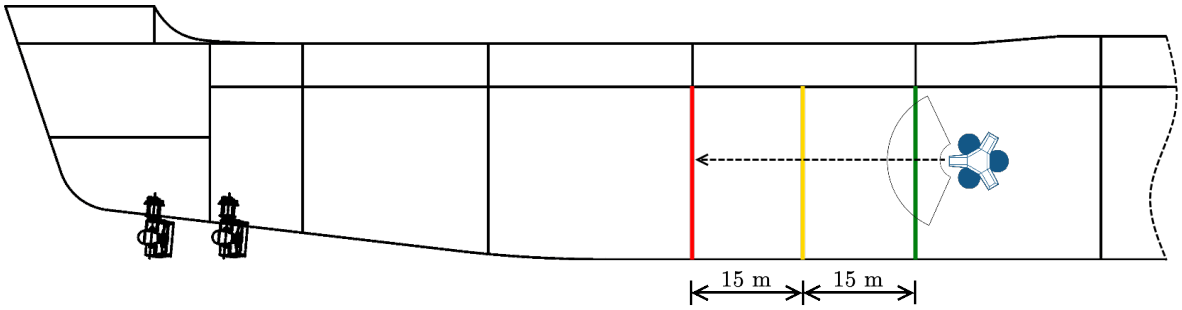
7-1-1 Experiment description

This experiment was carried out during cleaning operation of the Allseas Pioneering Spirit in the Port of Rotterdam. With a length of 382 m this is the largest construction vessel in the world. The robot was guided on a segment between weld lines, maintaining constant depth and velocity. The weld lines act as markers for a ground truth measurement, where the distances are estimated from the ship's general arrangements and validated by measuring on the actual ship. Weld line markers are visible at the start, midway and end of the segment, as shown in Figure 7-2. The total distance is 30.0 m with a marker midway at 15.0 m.

Figure 7-1: Allseas Pioneering Spirit in the Port of Rotterdam.

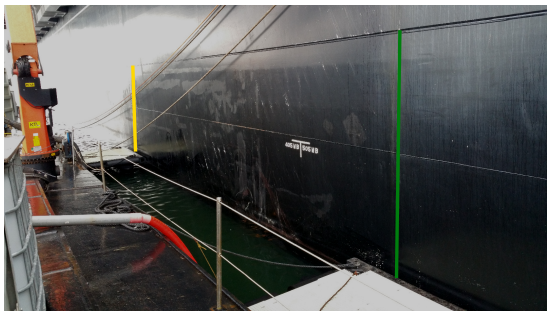
Weld line datasets	
Date	24-May, 2018
Ship	Pioneering Spirit
Location	Port, flat side
Total distance	$30.0 \pm 0.05\,\mathrm{m}$
Midway marker	$15.0 \pm 0.05\,\mathrm{m}$

Table 7-1: Specifications of weld-line experiment



(a) Segment on the general arrangements, weld lines are marked by red, yellow and green.

(b) First weld line on the ship.



(c) Second and third weld line on the ship.

Figure 7-2: Segment on the Pioneering Spirit. Marked lines on the GA correspond to weld lines on the ship.

The segment was repeated several times along different directions, depth levels maintaining a reference velocity of around $0.14\,\mathrm{m\,s^{-1}}$, as summarized in Table 7-2. Depth and velocity were controlled by an autopilot function, though small changes are to be expected.

Auxiliary sensor measurements are collected so that the visual odometry output can be compared to the current positioning system, which relies on wheel encoders and IMU measurements. In the current setup each of the wheels is equipped with an encoder. To improve the estimate when one of the wheels is slipping, the minimum value of the encoders is taken as the output:

$$v_{enc} = \min(v_{front}, v_{left}, v_{right}) \tag{7-1}$$

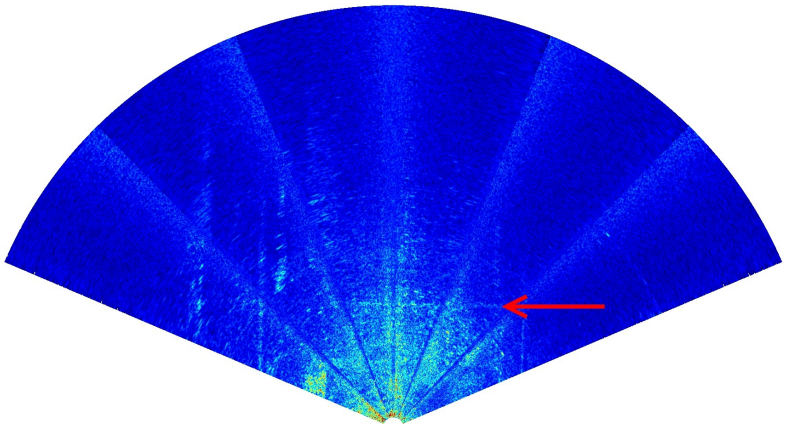


Figure 7-3: Weld line visible on sonar image, indicated by red arrow.

Dataset	Reference depth [m]	Average velocity [m/s]	Duration [s]	Frames
1	9.2	0.14	229.7	1965
2	12.2	0.13	240.4	1966
3	12.4	0.14	224.9	1909
4	6.5	0.13	234.1	2049

Table 7-2: Details of the recorded datasets.

It should be noted that the conditions for the wheel encoders during the Pioneering Spirit experiment are as ideal as currently possible. Multiple wheel encoders were performing without mechanical error. Furthermore, this ship featured a reliable surface without slime, such that minimal wheel slippage was encountered during recording.

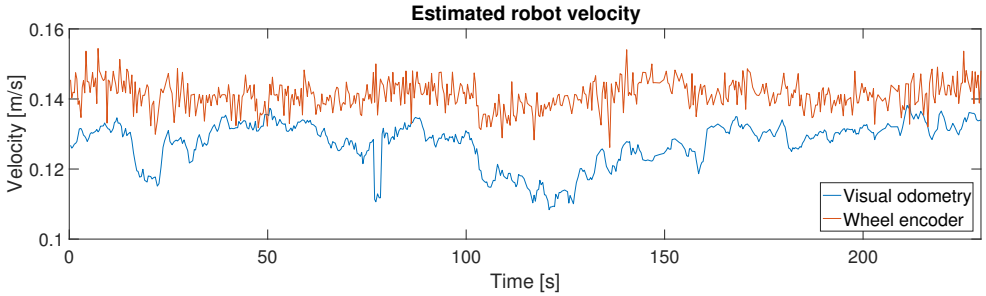
7-1-2 Results

Performance of the visual odometry algorithm is compared to ground truth measurements and the current positioning system. The criteria set up in Chapter 4 are used as metrics to evaluate the performance.

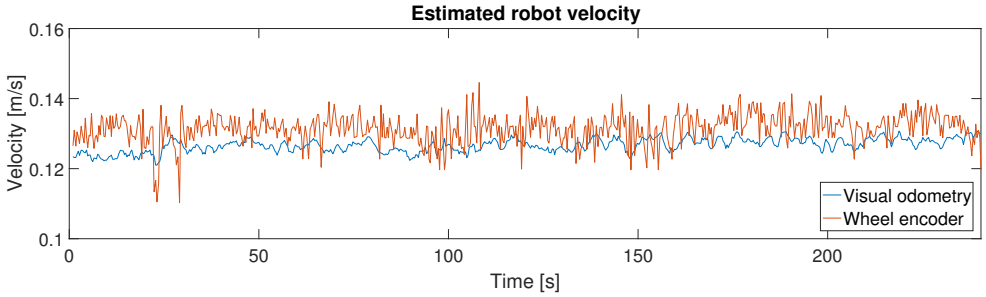
Accuracy

The estimated distances by visual odometry and wheel odometry are compared to the ground truth measurements from the general arrangements. The error build-up is calculated indicating the accumulated error during the segment. Table 7-3 shows these results.

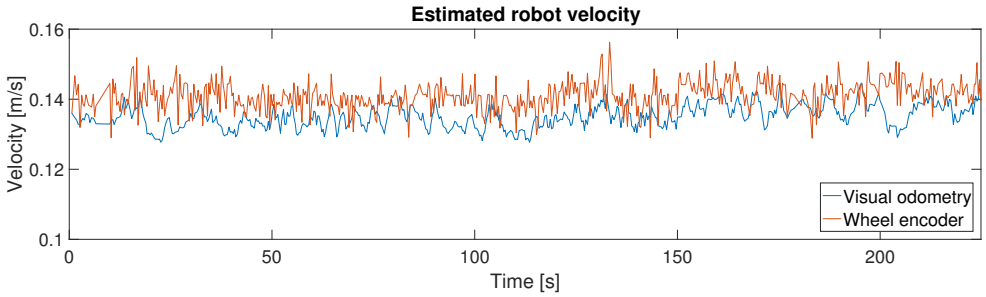
While this is an ideal scenario for the wheel encoders, the visual odometry method shows a better accuracy in almost all datasets. Dataset 4 is the only scenario where the accuracy exceeds the desired requirement of EBU $> 5\%$. As this dataset is recorded at a lower depth level, this is most likely caused by the visibility of the waterline and the occurrence of reflections and wave perturbations in the sonar images. This effect can be seen in Figure 7-5.



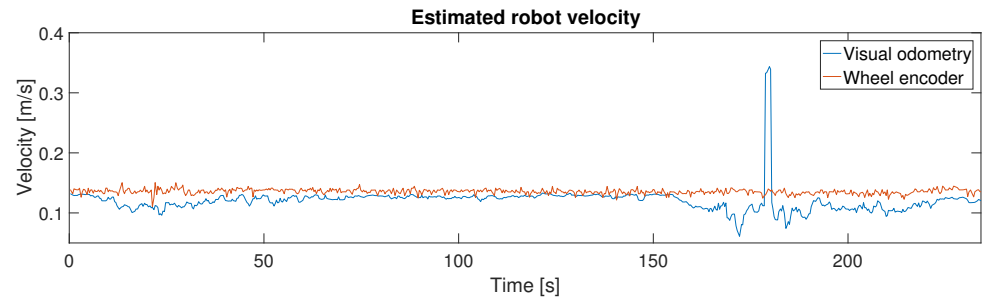
(a) Dataset 1



(b) Dataset 2



(c) Dataset 3



(d) Dataset 4

Figure 7-4: Estimated velocity by visual odometry compared to wheel encoders.

Dataset		Visual odometry		Wheel encoders	
		Est. distance	Error build-up	Est. distance	Error build-up
1	midway	14.42 m	**3.8%**	15.90 m	6.0%
	final	29.21 m	**2.6%**	32.39 m	8.0%
2	midway	-	-	-	-
	final	30.33	**1.1%**	31.45	4.8%
3	midway	14.80	**1.3%**	15.50	3.3%
	final	30.29	**1.0%**	31.63	5.4%
4	midway	14.48	**3.5%**	16.15	7.7%
	final	28.17	6.1%	31.72	**5.7%**

Table 7-3: Error build-up during the segment by visual odometry and wheel odometry. In dataset 2 the midway marker was not clearly visible on the sonar view.

These artifacts can show up at the same location in multiple sonar images and as such visual odometry will correlate these artifacts instead of the robot motion.

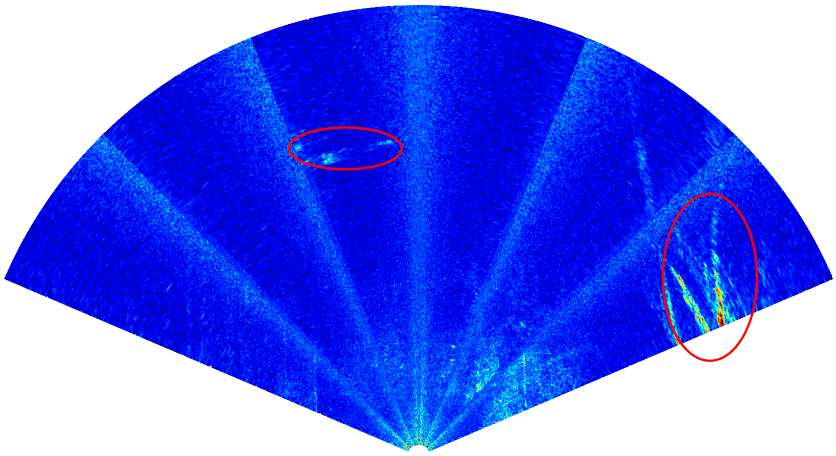


Figure 7-5: Artifacts that cause disturbances indicated by the red circles. Reflections and wave perturbations are caused by the presence of the waterline.

Robustness

Robustness of the algorithm is impacted by the quality of registration. Since each error accumulates over time, each erroneous registration impacts the final output. In Figure 7-4 the estimated velocity over time is shown from both visual and wheel odometry methods. In datasets 1-3, the results of visual odometry seem quite stable, but in dataset 4 some outliers are clearly visible. The lowered accuracy was already explained by the occurrence of waterline reflections and wave perturbations. As a consequence of that, erroneous registrations are clearly present in dataset 4.

When taking a look at the Peak to Sidelobe Ratio (PSR) metric for the registrations in Figure 7-6 it can be seen that the PSR dips to very low values between 150s and 200s, around the same time the outliers are present in the velocity graph. This implies that the PSR metric can be used as an indicator for erroneous registrations.

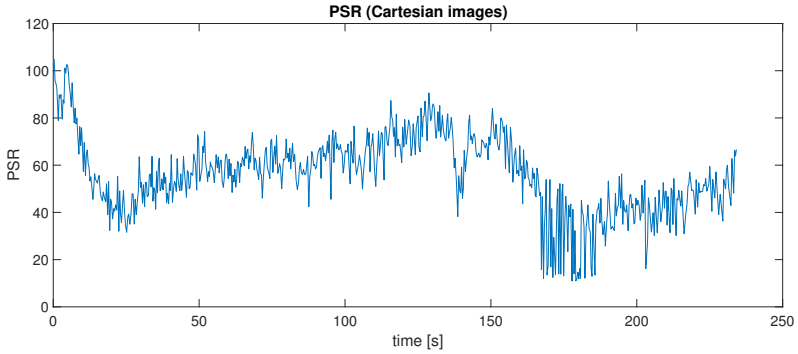


Figure 7-6: PSR metric for each registration in dataset 4.

Recall from Section 4-2 that precision and recall were proposed as metrics for robustness. False negatives are not relevant as every registration is currently accepted, thus for now only precision is considered. The PSR is used as an indicator for false registrations. When correlating images with no overlap, PSR values below 20 are found. With that information and experience of different scenarios, a PSR value of 40 is taken as a threshold for erroneous registrations.

Dataset	Erroneous registrations	Precision
1	0	100%
2	0	100%
3	1	99.8%
4	91	86.7%

Table 7-4: Precision for each dataset. An erroneous registration is defined as a registration where PSR < 40.

From Table 7-4 it can be seen that only dataset 4 contains registrations that fall below the threshold, which is consistent with earlier observations. For the other datasets the registrations (almost) never fall below the threshold, implying the consistency of the algorithm under expected circumstances.

Computation time

Computation time is obviously dependent on the specifications of the used hardware. However, examining computation time gives insight in the possibility of the algorithm running in real time and which subfunction employs the most resources. Table 7-5 shows the average computation time of each registration in dataset 1.

Subfunction	Computation time [ms]	
	Cartesian	Polar
Preprocessing (filter and mask)	74	29
Phase correlation	354	97
Correlation filter	1348	430
Peak detection	12	4
Apply rotation	49	
Total	**2397**	

Table 7-5: Average subfunction computation time of dataset 1

The current computation time needed is 2.4 s, which is more than the 0.3 s that is required for real-time operation when every third frame is analyzed. It can be seen that the Butterworth filter that is applied on the phase correlation matrix has the highest contribution to the overall computation time.

7-2 Rotation

7-2-1 Experiment description

In this experiment the proposed method for rotation estimation is further considered. When cleaning on the bottom of a vessel, the yaw measurement of the IMU is unreferenced and as a consequence drifts over time. As rotational differences are estimated by the proposed visual odometry method as well, the intuition is that this estimation may be used in the scenario where the IMU measurement is unreliable.

The used dataset was recorded again on the Pioneering Spirit. Instead of driving in a straight line, the robot makes a turn. An overview of the dataset is given in Table 7-6. As a ground truth the yaw output of the IMU is used. Since this experiment is conducted on the flat side of the ship, the yaw value is assumed to be drift-free.

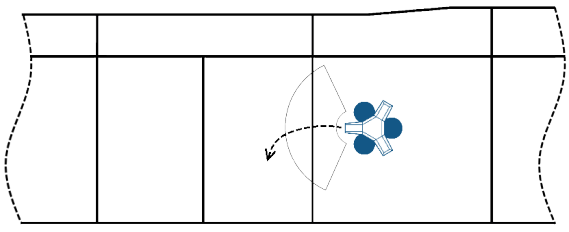
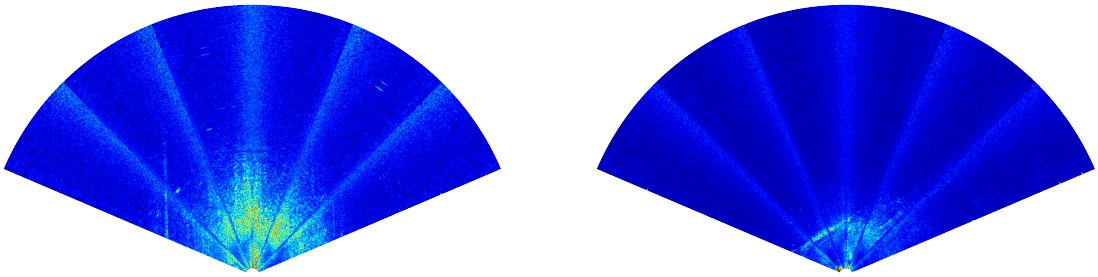


Figure 7-7: Schematic overview of rotation experiment.

Weld line datasets	
Date	24-May, 2018
Ship	Pioneering Spirit
Location	Port, flat side
Total angle difference	$56.4° \pm 0.5°$

Table 7-6: Specifications of rotation experiment



(a) Start frame **(b)** End frame

Figure 7-8: First and last frame of the rotation experiment dataset. From the sonar images the total rotational displacement is roughly $56°$.

7-2-2 Results

The estimated yaw orientation by visual odometry drifts far over time and acquires a 34.0% error build-up over the segment. However, the results are dependent on n_f (the number of frames between registrations), which can be seen in Figure 7-9. This is caused by the inaccuracies of peak detection when the overlap is high (low n_f) and degrading correlation when the overlap is low (high n_f). In any case, these results show too much error build-up for the method to be reliable as an alternative to the IMU measurements. Furthermore, the resulting estimates depend on the parameter value of n_f, where ideally the estimated value is independent of the overlap between images.

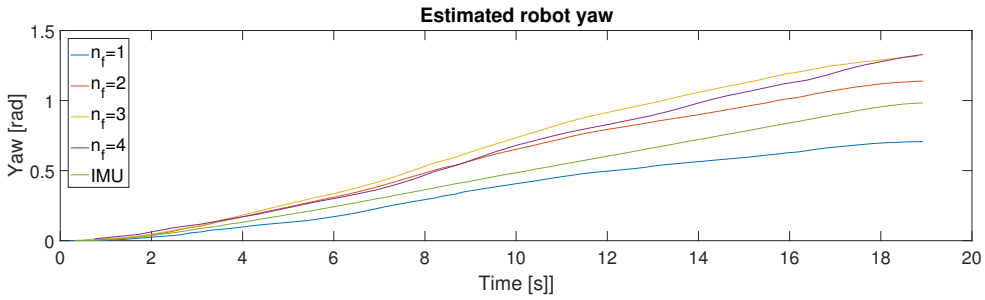


Figure 7-9: Estimated robot yaw orientation by visual odometry. Differen values for n_f (frames between registrations) are compared against the referenced IMU measurments.

One of the limitations described in Section 6-2-5, is the difference between the sonar origin and the actual center of rotation of the robot. We can compensate for this difference by recomputing the polar images while taking into account the transformation relating the sonar origin to the center of rotation. The implementation of this transformation is further detailed in Appendix C-1-3. With the recomputed sonar images the results are indeed more accurate as shown in Figure 7-10. It can be seen that the result is most accurate using $n_f = 4$ with an error build-up of 1.7%. With the desired value of $n_f = 3$, the error build-up is 9.0%, which is outside the required value, but still significantly more accurate than without the recomputed polar images. Finally, it should be noted that recomputing the polar images is an operation with a heavy computational cost, causing the computation time to increase to 16 s per registration.

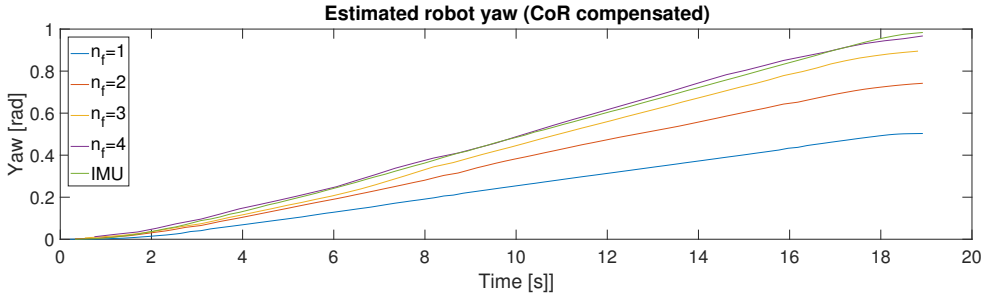


Figure 7-10: Estimated robot yaw orientation with recomputed polar images to compensate the center of rotation. Differen values for n_f (frames between registrations) are compared against the referenced IMU measurments.

7-3 Wheel slip

7-3-1 Experiment description

In the weld line experiment the proposed visual odometry method was compared to the most ideal scenario for the wheel encoders. However, in practice Fleet Cleaner has experienced problems with the wheel encoders. One of the major problems is the occurrence of wheel slippage on slippery surfaces due to slime and/or algae. In addition to this, the wheel encoders are prone to failure caused by the accumulation of dirt. This means that taking the minimum value of the encoders in the case of wheel slippage may not always prove successful.

Kes Cassee has researched methods to detect wheel slippage and correct the velocity estimation accordingly [67]. Nonetheless, visual odometry is obviously not affected by the occurrence of wheel slip and should prevent drifting of the position estimate in that case. In this section an occurrence of wheel slip during cleaning of the OOCL Europe is analyzed, see Table 7-7. A comparison between visual and wheel odometry is made in this specific case.

Wheel slip dataset	
Date	1-Aug, 2017
Ship	OOCL Europe
Location	Port, flat side
Total distance	$6.5\,\text{m} \pm 0.5\,\text{m}$

Table 7-7: Specifications of wheel slip experiment

7-3-2 Results

In Figure 7-11 it can be seen that from about 48 s the wheel encoder measurements show erratic behavior. This is caused by the wheels slipping, as the footage from the sonar clearly shows that the robot's motion is stagnated in the end of this segment. The visual odometry estimate is consistent with this observation as the estimated velocity slows down towards 0 at the expected time.

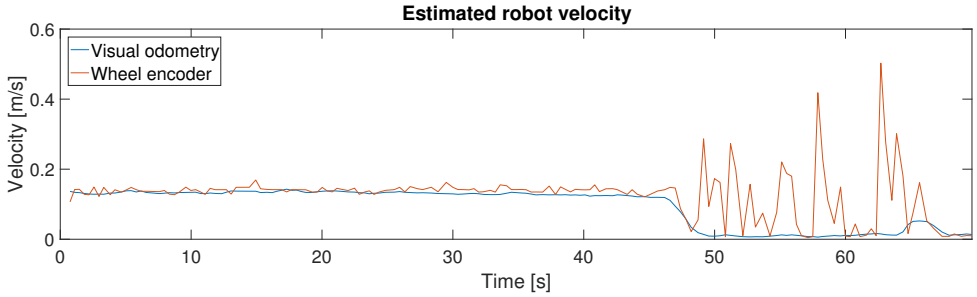


Figure 7-11: Estimated robot velocity by visual odometry and wheel encoders. From the 48 s point it can be seen that the encoders show unstable behavior due to wheel slip occuring.

The consequence of this is visible when observing the estimated location of the robot in Figure 7-12. The wheel encoder estimate drifts and the ending location is significantly farther away from the visual odometry estimate. For this particular segment there is no ground truth measurement available, but from visible weld lines, the traveled distance in x-direction is estimated to be around 6.5 m.

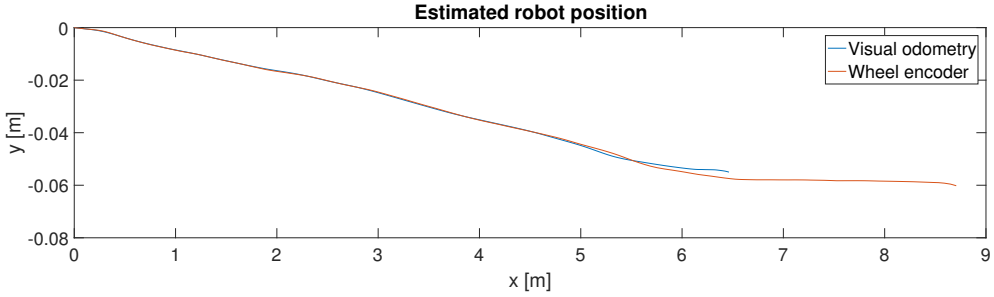


Figure 7-12: Estimated robot position by visual odometry and wheel encoders. It can be seen that the wheel encoders drift by a significant margin due to the occurrence of wheel slip.

7-4 Discussion

In this chapter the practical application of the proposed visual odometry method was analyzed. By recording datasets during live cleaning operations of the Fleet Cleaner, the behavior in actual operating conditions can be examined. The algorithm was tested in the scenario where the robot is driving in a straight line with a known start and end point. Furthermore, the ability to estimate the robot's yaw orientation was considered. The performance is compared to wheel encoder measurements, the current status quo. Finally, the results are compared in the case of wheel slip, a scenario where the wheel encoder estimate drifts heavily.

7-4-1 Findings

In the simple case of driving in a straight line, the proposed Fourier-based visual odometry method showed superior performance when compared to the wheel encoders. On 7 of the

8 distance markers, the estimated traveled distance was within the desired requirement for error build-up (EBU $< 5\%$).

As a metric for the quality of a registration, the PSR value was used. The intuition being that a bad correlation leads to a correlation matrix without a significant dominant peak and with many side-peaks. Under normal conditions the algorithm was able to perform consistent, reaching almost 100% precision scores. However, when driving at a lower depth, the waterline was visible which caused reflections and wave perturbations to be visible on the sonar images. This had a significant impact on the quality of registration as the precision lowered to 86.7%, below the requirement of 98%.

While this scenario features the ideal conditions for wheel encoders, with multiple functioning encoders and a rough surface on which the wheels did not slip. Even then, visual odometry was able to achieve a higher accuracy. However, the improvement becomes more significant when comparing to the wheel encoders in not so ideal conditions. In the case of wheel slip, which may happen often on surfaces fouled with slime and algae, the position estimate drifts a couple of meters over a short period of time. In this scenario it was shown that the visual odometry estimate showed the expected behavior of stagnated motion. In the case of wheel slip or hardware failure, visual odometry is able to provide an accurate alternative to the wheel encoders.

In addition to translational shifts, the algorithm also estimates rotational shifts. The estimate for yaw rotation was compared to IMU measurements to find if the estimate could be used as an alternative in situations where the IMU is unreferenced, i.e. when cleaning on the bottom of a vessel. However, the results showed a high error compared to the IMU reference. Furthermore, the results were dependent on the amount of overlap between images. While the rotation estimation is good enough for compensation in the phase correlation pipeline, it is not accurate enough to function as a yaw orientation estimate as an alternative to the IMU.

7-4-2 Limitations

In spite of the promising results of Fourier-based visual odometry, some limiting factors were found as well.

Reflections and other perturbations

While under normal conditions the performance of the proposed method seems promising, it is sensitive to reflections and waves that are visible at lower depth levels. The motion of these artifacts are independent from the robot motion and thus will interfere with the correlation process.

Rotation estimation

Estimation of the robots yaw orientation is not accurate enough with the proposed method. Rotation is estimated using phase correlation on the polar images, which has two limitations:

- Since the robot is a three-wheeled vehicle, it always features a translation in addition to a pure rotation. In the polar coordinates these motions are not decoupled and as such will deform the image, causing an inaccurate estimation of the rotational displacement.
- The algorithm assumes the origin of the sonar frame to be the center of rotation. However, this is not the case on the Fleet Cleaner robot, inducing an error in the estimation of the robot rotation.

The latter limitation can be accounted for by recomputing the polar images such that the center of rotation becomes the origin of the sonar image frame. It is shown that the resulting estimation is indeed more accurate. However, this operation adds a heavy computational cost to the process.

Computation time

Finally, a note on the computational cost of the proposed method. The current computation time is considerably higher than the required computation time of 0.3 s for real-time operation. The image processing steps, especially the Butterworth filer, are most demanding on the high resolution FLS images. MATLAB uses the CPU to make these calculations instead of the GPU, which should be faster with image processing steps. Furthermore, the expectation is that with better hardware and a more optimized code language, the algorithm should be able to achieve a computation time that is near the requirement. Another note is that the computation time is almost constant given the same image size and thus not dependent on the image content.

Chapter 8

Closing remarks

The thesis, presented in this document, is concluded in this final chapter. A summary of the completed work is provided in Section 8-1. The final conclusions regarding the thesis goal and objectives are presented in Section 8-2. Finally, in Section 8-3 the future work and recommendations are discussed.

8-1 Summary of completed work

The Fleet Cleaner robot currently suffers from unbounded error build-up resulting in a drifting position estimate. A visual Simultaneous Localization and Mapping (SLAM) system can be used to bound this error by utilizing visual information in addition to odometry and IMU measurements. Visual information is gathered with a FLS. Sonar images, however, suffer from noise, lower resolution and are difficult to interpret, compared to optical images. The thesis goal is, therefore, to *develop a sonar-based underwater SLAM framework tailored to the working environment of ship hull cleaning robots.* The thesis scope is limited by dividing the goal into two specific objectives: *propose a conceptual design for a SLAM system*, and *validate the sonar-based visual odometry algorithm with a proof of concept.* Visual odometry is of importance as the robot is expected to drive long segments without returning to a previously visited location and it is vital that the position estimate does not drift significantly during these segments. A systematic approach to engineering design is adopted as a guide to accomplish these objectives and consists of the following steps: task clarification, conceptual design, embodiment design and proof of concept.

Chapter 2 provides the reader with some necessary background information on the principles and working of FLS imagery. The fundamental properties of acoustic sound propagation are explained, which serves as the working principle for all sonar imaging systems. The operation of FLS devices is discussed, as well as different geometry models that derive the sonar motion from different images. A simplified model, based on orthographic projection, and an exact model that takes elevation angles into account are considered. Ultimately, the simplified

model is adopted as it has minimal approximation errors under the assumption that the elevation angles are small compared to the range.

Chapter 3 considers the operating conditions that are relevant to the development of a SLAM system. First, the limitations of the FLS imagery are discussed. These include non-uniform resolution, speckle noise, inhomogeneous insonification and reflection artifacts. Next, the auxiliary sensors used in the positioning system are detailed. These include the IMU, depth sensor and wheel encoders. Yaw orientation, and x- and y-position are unreferenced, resulting in error build-up in the position estimate. Finally, constraints in the environment that have an impact on SLAM are listed. The types of visible objects are noted, as well as the observation that ships are typically sparse with these elements and long segments without distinct landmarks are expected.

Chapter 4 proposes the requirements for the system as the final step in the task clarification phase. The SLAM system is split into three subsystems: visual odometry, loop closure detection and, pose and map detection. Functional requirements are defined for each subsystem, concerning the abstract behavior. Non-functional requirements specify the desired performance of the system and are used as metrics to compare and evaluate different working principles. These metrics consider accuracy, robustness and computation time.

Chapter 5 discusses different working principles for the subsystems. The principles are evaluated based on discussed literature and rated using the non-functional requirements. The proposed optimal solution is chosen, while considering performance, requirements and the operational conditions. For visual odometry, a phase correlation method is chosen as it is most suited for the sonar images. Furthermore, object-based image to map matching is scored best for loop closure detection and particle filter for pose and map estimation. The remainder of the report is concerned with the implementation and proof of concept of the visual odometry subsystem.

Chapter 6 details the implementation phase for the visual odometry subsystem. Motion between subsequent images is estimated by detecting a dominant peak in the phase correlation matrix. Several image processing steps are proposed to improve the peak sharpness, increasing accuracy and precision of the method. Rotation can be estimated by applying the same method to the sonar images in polar representation. Finally, tuning of parameters is done according to the PSR metric, which indicates how many standard deviations the dominant peak is above the average of the surface.

Chapter 7 validates the proposed method in an experiment that was conducted during cleaning of the Pioneering Spirit. While driving in a straight line, the visual odometry estimates are within the desired accuracy ($> 95\%$). However, outliers are present when driving near the waterline, caused by reflections and wave reverberations. Rotation estimation is compared to the IMU output, but failed to reach the required accuracy in a reliable manner. Finally, the algorithm is compared to wheel odometry in a scenario where wheel slip occurs.

8-2 Main conclusions

This thesis has contributed to the overall goal to *develop a sonar-based underwater SLAM framework tailored to the working environment of ship hull cleaning robots.* This main contribution can be split into more specific contributions:

The first objective of this thesis, *propose a conceptual design for a SLAM system*, has been completed by presenting an optimal working structure. The working structure combines a view-based approach for visual odometry and a object-based approach for loop closure detection. For pose and map estimation, the particle filter was chosen, but it should be noted that the other solutions are viable as well. This solution is tailored to the operating conditions of the Fleet Cleaner robot, thereby contributing to the main objective.

The second objective of this thesis, *validate the sonar-based visual odometry algorithm with a proof of concept*, has been accomplished by showing that the proposed Fourier-based method achieved the required accuracy (EBU $< 5\%$). Furthermore, the results showed an improvement over wheel encoders in almost all datasets. By reducing error build-up during segments between loop closures, this method contributes to the main objective.

Limitations

Beyond the visual odometry component, the conceptual SLAM design has not been implemented. Although it is believed that the proposed system would improve the current position and navigation system, this has not been experimentally validated under operational conditions.

As phase correlation is heavily dependent on the image content, the method is impacted by the particular nature of FLS images. Speckle noise and other artifacts such as reflections and wave reverberations were found to have the greatest impact. The effect of speckle noise can be reduced by filters, but these are either too costly (in the case of adaptive filters) or too general (in the case of median filter). Reflections and perturbations that occur when the waterline is visible are more problematic and have a degrading effect on the performance of visual odometry.

Using the phase correlation method to estimate yaw orientation did not accomplish the desired accuracy. Furthermore, the accuracy is dependent on the frames between registration pairs. The lowered accuracy is mainly explained by the combined rotations and translations, which deform the polar representation of the sonar images.

Finally, it should be noted that the proposed visual odometry method has not fulfilled the requirements regarding computation time. The image processing steps, applied on the high resolution images, currently have the greatest contribution to the total computation time. However, it is expected that the requirement could be fulfilled in the future with better hardware and more optimized coding.

8-3 Recommendations

The results could be further improved by addressing the previously listed limitations of the completed work. It is believed that the following areas are worth exploring in future work:

SLAM implementation

The next step is to further implement the SLAM system. The visual odometry subsystem showed promising results, but the error can be further bounded by incorporating detection of

loop closures. As proposed in Section 5-2, objects such as gratings can be used as landmarks for loop closure detection. This thesis has provided a framework for a SLAM implementation, but this should be further tested and validated in future work.

Opti-acoustic navigation

This work has focused completely on sonar imagery for visual information, disregarding optical cameras for their poor quality in the presence of turbulence and dirt. However, the optical images can provide visual information of textures and other properties not visible on the sonar, which may be used as constraints for loop closures. The main advantage of this is the wide library of existing algorithms that are tailored to the use of optical cameras. This combined approach of optical and acoustical visual navigation has been applied on ship hull inspection by Hover et al. [32].

Data fusion

Another are of interest is to fuse the current positioning methods. Visual odometry and wheel odometry can be used in conjunction to improve accuracy and precision of the overall estimate. Furthermore, fusion can be used to mitigate shortcomings of both methods. While previous approaches have taken fusion of the positioning sensors (wheel encoders, IMU) into consideration [6, 67], this can be further expanded by fusion of imaging data.

Computation time

While most requirements are fulfilled by the proposed visual odometry method, the desired computation time is not reached. By migrating from MATLAB to Python, the programming language used by Fleet Cleaner, and further hardware accelerations (i.e. using GPU), the necessary optimization should be achievable.

Improved image processing

Outliers and erroneous registrations in the visual odometry algorithm are mainly caused by the nature of sonar images. Noise, reverberation artifacts and acoustic returns from the water surface can cause false correlations between images. Efficient methods to further reduce the effect of speckle noise should be researched. Furthermore, the effect of waterline reflections could be partly reduced using waterline detection and clipping everything above the waterline out of the images.

Image mosaicing and blending

Besides visual odometry, other applications of image registration are possible. Mosaics or maps can be created by stitching consecutive sonar images to each other using the estimated translations and rotations [18, 27]. These mosaics can provide visual details of regions of interest. In addition, by blending or averaging consecutive images, speckle noise can be further reduced. This can be used to improve the results of object detection algorithms [19].

Appendix A

Sensor specifications

In this appendix, the auxiliary sensors are listed. The descriptions are based on technical documentation of the specific sensors.

A-1 Forward looking sonar

The sonar deployed by Fleet Cleaner is the Blueview M900-2250 [16]. This sonar has two frequency modes: a long range mode operating at 900 kHz and a higher quality short range mode operating at 2250 kHz. The full technical specifications are listed in Table A-1.

Sonar specifications	
Operating frequency	900 kHz/2250 kHz
Field of view	130°
Max range 900/2250	100 m/10 m
Range resolution 900/2250	1.3 cm/0.6 cm
Beam width	1°× 20°
Number of beams	768
Angular resolution	0.18°
Max update frequency	25 Hz

Table A-1: Blueview M900-2250-230 specifications [16].

Images are captured using the provided Proviewer software [68]. The raw data, a 16 bit unsigned integer image, can be streamed to third-party applications, such as MATLAB. The data contains the polar and Cartesian image, ping number, time stamp, resolutions and imaging range.

A-2 IMU

The IMU used by Fleet Cleaner is the Xsens MTi-20 [69]. Pitch and roll values can be measured absolutely, while yaw orientation is unreferenced. The IMU outputs a rotation matrix, from which the orientations can be derived. Furthermore, direct accelerations and velocities from the accelerometers and gyroscopes can be measured. Noise and bias specifications from the producer are listed in Table A-2.

	Accelerometers	Gyroscopes
Noise density	$60\,\mu g/\sqrt{Hz}$	$0.03\,^\circ/s/\sqrt{Hz}$
In-run bias stability	$15\,\mu g$	$18\,^\circ\,h^{-1}$

Table A-2: Noise and bias specifications of IMU [69].

A-3 Wheel encoders

The Fleet Cleaner robot is equipped with a RLI200 bearingless wheel encoder [70] on each of the three wheels. From the encoder, the forward velocity of the robot can be calculated. The specifications are listed in Table A-3.

Accuracy	0.3°
Pulse rate	2800 ppr
Input frequency	250 kHz

Table A-3: Accuracy, pulse rate and input frequency of wheel encoders [70].

A-4 Depth meter

Fleet Cleaner uses a depth meter to measure the absolute y-coordinate of the robot. The water pressure is measured by the ATM.1ST/N. The measured pressure by the depth meter is the absolute pressure. To account for the atmospheric pressure a barometer is installed that measures atmospheric pressure at initialization of the robot. This measurement is then deducted from the water pressure measurement. Specifications are listed in Table A-4 and A-5.

Accuracy	0.1 % FS
Total error	0.8 % FS

Table A-4: Accuracy and total error of depth sensor [71].

Accuracy	0.1 % FS
Total error	0.4 % FS

Table A-5: Accuracy and total error of barometer [72].

A-5 Steering angle encoder

The angle of the steering wheel is measured by an electromagnetic absolute encoder. The TBX36 has a resolution of 4096 steps per 360° [73]. The technical documentation reports a measured deviation of ±0.5 of its least significant bit or in degrees:

$$\frac{360^\circ}{4096} \cdot 0.5 = 0.044^\circ \tag{A-1}$$

Appendix B

Velocity estimation

This appendix describes the process to estimate robot velocity from the translation and rotation in the 2D-plane. The sonar is tilted around the y-axis by ϕ, see Figure B-1. The x-compoment of the sonar velocity therefore has to be multiplied by $\cos\phi$:

$$v_{sonar} = \begin{bmatrix} t_x \\ t_y \\ 0 \end{bmatrix} \cdot \begin{bmatrix} \cos\phi \\ 1 \\ 1 \end{bmatrix} \cdot \frac{1}{\Delta t} \tag{B-1}$$

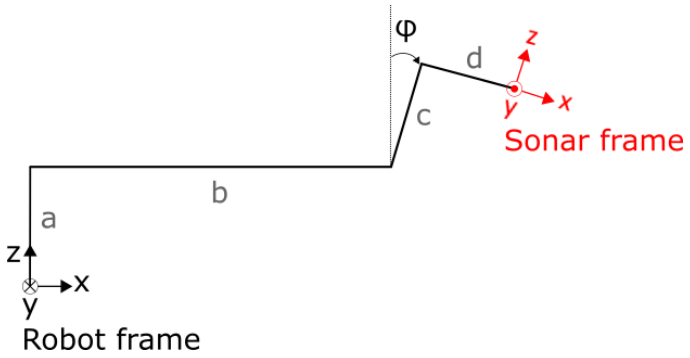


Figure B-1: Side view of the sonar geometry in relation to the robot frame [74].

Element	Unit
a	271 mm
b	880 mm
c	80 mm
d	251 mm
ϕ	10.0°

Table B-1: Dimensions of the elements in Figure B-1 [74].

The rotation of the image frames in Cartesian coordinates corresponds to the same rotation of the robot body as the projection preserves the change in yaw.

$$\omega_{robot} = \begin{bmatrix} 0 \\ 0 \\ \theta \end{bmatrix} \cdot \frac{1}{\Delta t} \tag{B-2}$$

The sonar velocity v_{sonar} consists of the translational velocity and the tangential velocity due to the rotational motion of the robot body. In the transformation to the velocity of the robot frame we take into account this component with the cross product $r_{sonar} \times \omega_{robot}$. r_{sonar} can be calculated from the sonar dimensions, see Figure B-1.

$$r_{sonar} = \begin{bmatrix} b + c\sin\phi + d\cos\phi \\ 0 \\ a + c\cos\phi - d\sin\phi \end{bmatrix} \tag{B-3}$$

$$v_{robot} = v_{sonar} + r_{sonar} \times \omega_{robot} \tag{B-4}$$

Appendix C

Algorithms

This appendix lists source code of self-written and existing algorithms, that were altered. If necessary, additional information is provided. The experiments were performed using MATLAB R2016b on a personal computer (PC) with an Intel Core i7-3630QM CPU that features a clock speed of 2.4 GHz and 8 GB of usable RAM.

C-1 Fourier-based odometry

C-1-1 Phase correlation

The following function is used to estimate the transformation between two images by analyzing the phase correlation in the Fourier domain.

```
1  function [shift, PSR, fc, C_uv, C_xy] = phaseCorrelation(I1, I2, regOptions, floodTol)
2  %PHASECORRELATION Estimates spatial shift of two images by analyzing phase
3  %correlation
4  %   I1, I2: Set of two images in polar or cartesian coordinates
5  %   regOptions: Options for filters
6  %   shift: Spatial transformation of I1 to I2
7  %   C_xy: Correlation matrix
8
9  F1 = fft2(I1);
10 F2 = fft2(I2);
11 [n,m]=size(I1);
12 C_uv = (F1.*conj(F2))./abs(F1.*conj(F2)); %phase correlation matrix
13 C_uv = fftshift(C_uv);
14 if (nargin > 2)
15     if ~regOptions.fcBool
16         fc = estimateFc(C_uv, floodTol);  %use adaptive cutoff freq for bw filter
17     else
18         fc = regOptions.fc;
19     end
20     C_uv = applyBwFilter(C_uv, fc, regOptions.k_filter);
21 end
22 C_xy = abs(fftshift(ifft2(C_uv,'symmetric')));
23
```

```
[t_x,t_y, peak] = detectPeak(C_xy);
t_x=floor(m/2)-t_x+1; %difference from center of image
t_y=floor(n/2)-t_y+1;
shift = [t_x, t_y];

u = mean(C_xy(:));
s = std(C_xy(:));
PSR = (peak - u)/s;
end
```

C-1-2 Peak detection

The function below is used to detect peaks in the phase correlation matrix. The method is able to find sub-pixel displacement, by taking into account the difference between the two-sided neighbor [58].

```
function [t_x, t_y, peak] = detectPeak(C_xy)
%DETECTPEAK This function finds the largest peak in the correlation matrix
%   C_xy: Correlation matrix
%   t_y, t_x: Coordinates of peak
%   peak: Value of peak
%   ------------------------------------------------------------------------
%   This function is based on the work of Ren et al. (2010)

% Find largest value and neighbors
[peak, index] = max(C_xy(:));
[t_y, t_x] = ind2sub(size(C_xy),index);
[yMax, xMax] = size(C_xy);
y1 = t_y + 1; y2 = t_y - 1;
x1 = t_x + 1; x2 = t_x -1;
% Situation where the peak is at the edge of the matrix
if (t_y == 1)
    y2 = yMax;
elseif (t_y == yMax)
    y1 = 1;
end
if (t_x ==1)
    x2 = xMax;
elseif (t_x == xMax)
    x1 = 1;
end
% Find subpixel value
D_y = C_xy(y1, t_x) - C_xy(y2, t_x);
D_x = C_xy(t_y, x1) - C_xy(t_y, x2);
delta_y = sign(D_y)/(1 + C_xy(t_y, t_x)/abs(D_y));
delta_x = sign(D_x)/(1 + C_xy(t_y, t_x)/abs(D_x));
t_y = t_y + delta_y;
t_x = t_x + delta_x;
```

C-1-3 Transform polar origin

The function below is used to transform the origin of the polar images to the centre of rotation of the robot. This is done by using the law of sines to calculate the angle from the robot centre, depicted in C-1:

$$\sin\theta_{r,max} = r_{s,max}\frac{\sin(180^\circ - \theta_{s,max})}{r_{r,max}} \tag{C-1}$$

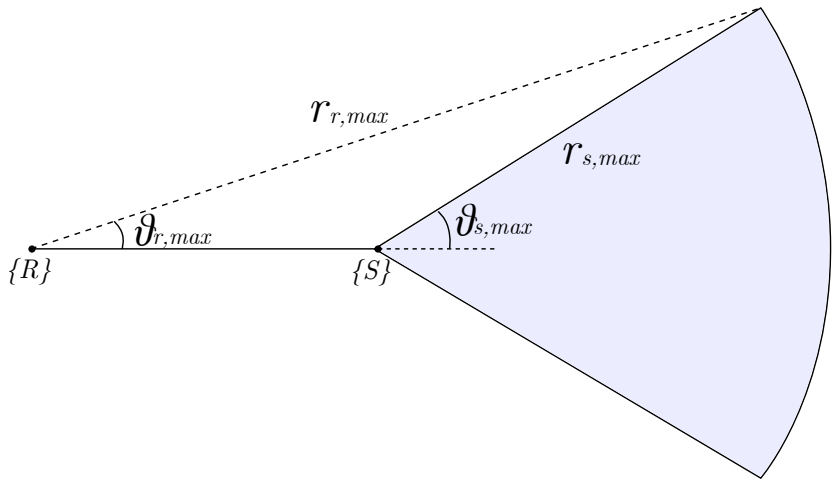


Figure C-1: Schematic overview of transformation from the sonar origin S to the robot centre of rotation R. r denotes the range and θ the bearing

```
function [I_transformed, r_r, theta_r] = transformPolarOrigin(I_old, sonar_arm,
    range_res, bearing_res)
%TRANSFORMPOLARORIGIN This function transforms the origin of the polar
%images to the centre of rotation of the robot
%   I_old: Original polar image
%   sonar_arm: Distance from robot centre to sonar
%   range_res: Resolution of range measurement
%   bearing_res: Resolution of bearing measurement
%   I_transformed: Transformed polar image
%   r_r: New max range
%   theta_r: New max bearing

I_old = double(I_old);
[n,m] = size(I_old);

thetaMax = m/2 * bearing_res;
rangeMax = n * range_res;

rangeMax_r = rangeMax + sonar_arm;
thetaMax_r = asin(rangeMax*sin((180 - thetaMax)*pi/180)/rangeMax_r);
range_res_r = rangeMax_r/n;
bearing_res_r = thetaMax_r * 2/m;

k = 1;
for i = 1:n
    for j = 1:m
        range_s = (n + 1 - i) * range_res;
```

```
27          theta_s = (j - m/2) * -bearing_res;
28          r_r(k) = range_s + sonar_arm;
29          theta_r(k) = asin(range_s/r_r(k) * sin((180-theta_s)*pi/180));
30          value(k) = I_old(i,j);
31          k = k + 1;
32      end
33  end
34
35  r_r = r_r/range_res;
36  theta_r = (theta_r*180/pi)/bearing_res + m/2;
37  [xq,yq] = meshgrid(1:m, 1:n);
38  I_transformed = griddata(theta_r,r_r,value,xq,yq);
39  I_transformed(isnan(I_transformed))=0;
40  I_transformed = flip(mat2gray(I_transformed),1);
41  I_transformed = flip(I_transformed,2);
42  end
```

C-2 Filters

C-2-1 Edge mask

The following function is used to create a mask that is used as a windowing operation to reduce edge effects. A fan-shaped mask is created for polar images and a rectangular mask for Cartesian images.

```
1  function [mask] = edgeMask(I, s, type)
2  %SONARMASK This function creates an edge mask based on the input image
3  %   Input type 'polar' or 'cartesian' define the type of mask that is
4  %   created. Standard deviation s defines the amount of smoothing of the
5  %   mask
6  %   I: Input image
7  %   s: Standard deviation
8  %   type: Cartesian or polar input image
9
10 % Image in Cartesian coordinates
11 if isequal(type, 'cartesian')
12     mask = I>0;
13     mask = imfill(mask,'holes');
14     mask = bwmorph(mask, 'shrink', s);
15     mask = imgaussfilt(double(mask), s, 'FilterSize', 6*s+1);
16
17 % Image in polar coordinates
18 elseif isequal(type, 'polar')
19     [n, m] = size(I);
20     mask = ones(n,m);
21     mask(1,:) = 0; mask(n-34:n,:) = 0;                %original image is not
           completely filled
22     mask(:,1:3) = 0; mask(:,m-1:m) = 0;
23     mask = bwmorph(mask, 'shrink', s);
24     mask = imgaussfilt(double(mask), s, 'FilterSize', 6*s+1);
25 end
26 end
```

C-2-2 Frost filter

The Frost filter [61] makes use of an adaptive kernel that changes its properties based on local image statistics. Source code from the Indian Institute of Technology Kharagpur [75] is used for this implementation.

```
function outputImage = fcnFrostFilter(inputImage,mask)

% fcnFrostFilter performs noise filtering on an image based
%   on using an adaptive filter proposed by Frost.
%
%   OUTPUTIMAGE = fcnFrostFilter(INPUTIMAGE) performs
%   filtering of an image using the Frost filter. It uses a square neighborhood of 5x5
%   pixels to estimate the gray-level statistics in default settings.
%   Supported data type for INPUTIMAGE are uint8, uint16, uint32, uint64,
%   int8, int16, int32, int64, single, double. OUTPUTIMAGE has the same
%   image type as INPUTIMAGE.
%
%   OUTPUTIMAGE = fcnFrostFilter(INPUTIMAGE,MASK) performs
%   the filtering with local statistics computed based on the neighbors as
%   specified in the locical valued matrix MASK.
%
%   Details of the method are avilable in Frost(1982)
%
%   2012 (c) Debdoot Sheet, Indian Institute of Technology Kharagpur, India
%       Ver 1.0     13 February 2012
%--------------------------------------------------------------------------

% Input argument check
iptcheckinput(inputImage,{'uint8','uint16','uint32','uint64','int8','int16','int32','
    int64','single','double'}, {'nonsparse','2d'}, mfilename,'I',1);
if nargin == 1
    mask = getnhood(strel('square',5));
elseif nargin == 2
    if ~islogical(mask)
        error('Mask of neighborhood specified must be a logical valued matrix');
    end
else
    error('Unsupported calling of fcnFirstOrderStatisticsFilter');
end

imageType = class(inputImage);
windowSize = size(mask);
inputImage = padarray(inputImage,[floor(windowSize(1)/2) floor(windowSize(2)/2)],'
    symmetric','both');
inputImage = double(inputImage);
[nRows,nCols] = size(inputImage);
outputImage = double(inputImage);
[xIndGrid yIndGrid] = meshgrid(-floor(windowSize(1)/2):floor(windowSize(1)/2),-floor(
    windowSize(2)/2):floor(windowSize(2)/2));
expWeight = exp(-(xIndGrid.^2 + yIndGrid.^2).^0.5);

for i=ceil(windowSize(1)/2):nRows-floor(windowSize(1)/2)
    for j=ceil(windowSize(2)/2):nCols-floor(windowSize(2)/2)
        localNeighborhood = inputImage(i-floor(windowSize(1)/2):i+floor(windowSize(1)
            /2),j-floor(windowSize(2)/2):j+floor(windowSize(2)/2));
        localNeighborhood = localNeighborhood(mask);
        localMean = mean(localNeighborhood(:));
        localVar = var(localNeighborhood(:));
        alpha = sqrt(localVar)/localMean;
        localWeight = alpha*(expWeight.^alpha);
        localWeightLin = localWeight(mask);
        localWeightLin = localWeightLin/sum(localWeightLin(:));
```

```
54          outputImage(i,j) = sum(localWeightLin.*localNeighborhood);
55      end
56  end
57
58  outputImage = outputImage(ceil(windowSize(1)/2):nRows-floor(windowSize(1)/2),ceil(
        windowSize(2)/2):nCols-floor(windowSize(2)/2));
59  outputImage = cast(outputImage,imageType);
```

C-2-3 Adaptive cut-off frequency

The following function is used to estimate the cut-off frequency. In [18] a flood segmentation algorithm is proposed. Instead, this function uses the simple `grayconnected` function to find regions with similar gray values.

```
1   function [fc] = estimateFc(C_uv, tol)
2   %ESTIMATEFC Estimate cutoff frequency based on flooding algorithm
3   %   C_uv: Phase correlation matrix
4   %   tol: Flood fill tolerance
5   %   fc: Adaptive cutoff frequency
6   %-------------------------------------------------------------------------
7   L = (angle(C_uv));
8   [Gmag, ~] = imgradient(L);)
9   [imageHeight, imageWidth] = size(C_uv);
10
11  %fill flooding from center, 8 connected with tolerace tol
12  BW = grayconnected(mat2gray(Gmag),floor(imageHeight/2), floor(imageWidth/2), tol);
13  [rows, cols] = find(BW==1);
14  y = rows - imageHeight/2;
15  x = cols - imageWidth/2;
16  norms = sqrt(sum([x,y].^2,2));
17  [~, ind] = max(norms);
18
19  fc = norm([y(ind)/imageHeight, x(ind)/imageWidth]);
20
21  end
```

Bibliography

[1] M. P. Schultz, J. A. Bendick, E. R. Holm, and W. M. Hertel, "Economic impact of biofouling on a naval surface ship," *Biofouling*, vol. 27, pp. 87–98, jan 2011.

[2] C. Kunz, C. Murphy, H. Singh, C. Pontbriand, R. A. Sohn, S. Singh, T. Sato, C. Roman, K. Nakamura, M. Jakuba, R. Eustice, R. Camilli, and J. Bailey, "Toward Extraplanetary Under-Ice Exploration: Robotic Steps in the Arctic," *J. Field Robotics*, vol. 26, no. 4, pp. 411–429, 2009.

[3] M. F. Fallon, J. Folkesson, H. McClelland, and J. J. Leonard, "Relocating underwater features autonomously using sonar-based SLAM," *IEEE Journal of Oceanic Engineering*, vol. 38, no. 3, pp. 500–513, 2013.

[4] H. Singh, R. Armstrong, F. Gilbes, R. Eustice, C. Roman, O. Pizarro, and J. Torres, "Imaging Coral I: Imaging Coral Habitats with the SeaBED AUV," *Subsurface Sensing Technologies and Applications*, vol. 5, no. 1, pp. 25–42, 2004.

[5] L. Paull, S. Saeedi, M. Seto, and H. Li, "AUV Navigation and Localization - A Review," vol. 39, no. 1, pp. 1–25, 2013.

[6] D. Borota, *Design of a position determination system for a ship's hull maintenance robot.* Msc thesis, Delft Center for Systems and Control, Technical University Delft, 2014.

[7] F. Hidalgo and T. Braunl, "Review of underwater SLAM techniques," *ICARA 2015 - Proceedings of the 2015 6th International Conference on Automation, Robotics and Applications*, pp. 306–311, 2015.

[8] G. Pahl, W. Beitz, J. Feldhusen, and K. Grote, *Engineering Design, a Systematic Approach.* Springer, 3rd ed., 2007.

[9] R. E. Hansen, *Introduction to sonar.* Course material inf-geo4310, University of Oslo, 2009.

[10] H. Johannsson, M. Kaess, B. Englot, F. Hover, and J. Leonard, "Imaging sonar-aided navigation for autonomous underwater harbor surveillance," *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems*, pp. 4396–4403, 2010.

[11] M. R. Walter, *Sparse Bayesian Information Filters for Localization and Mapping*. PhD thesis, MIT/WHOI, 2008.

[12] 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.

[13] H. Sekkati and S. Negahdaripour, "3-D motion estimation for positioning from 2-D acoustic video imagery," in *Pattern Recognition and Image Analysis*, pp. 80–88, 2007.

[14] S. Negahdaripour, P. Firoozfam, and P. Sabzmeydani, "On processing and registration of forward-scan acoustic video imagery," *Proceedings - 2nd Canadian Conference on Computer and Robot Vision, CRV 2005*, pp. 452–459, 2005.

[15] S. Negaharipour, "On 3-D scene interpretation from F-S sonar imagery," *OCEANS 2012 MTS/IEEE: Harnessing the Power of the Ocean*, 2012.

[16] "BlueView M900-2250-130 series." http://www.teledynemarine.com/M900-2250%20Dual%20Frequency%20Series. (Accessed 21-May-2018).

[17] H. Medwin, "Speed of sound in water: A simple equation for realistic parameters," *The Journal of the Acoustical Society of America*, vol. 58, no. 6, pp. 1318–1319, 1975.

[18] N. Hurtós, *Forward-Looking Sonar Mosaicing for Underwater Environments*. PhD thesis, University of Girona in, 2015.

[19] B. Steensma, *Ship hull cleaning robots: obstacle detection using a forward looking sonar system*. Msc thesis, Delft University of Technology, 2018.

[20] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, I. D. Reid, and J. J. Leonard, "Past , Present , and Future of Simultaneous Localization And Mapping : Towards the Robust-Perception Age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[21] T. Tuytelaars and K. Mikolajczyk, "Local Invariant Feature Detectors: A Survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2007.

[22] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Alvey vision conference*, vol. 15, no. 50, pp. 147–151, 1988.

[23] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[24] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[25] 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.

[26] E. Dubrofsky, *Homography Estimation*. Msc thesis, The University of British Columbia, 2009.

[27] S. Negahdaripour, M. D. Aykin, and S. Sinnarajah, "Dynamic scene analysis and mosaicing of benthic habitats by FS sonar imaging - Issues and complexities," *Proc. OCEANS*, pp. 1–7, 2011.

[28] N. Hurtos, S. Nagappa, X. Cufi, Y. Petillot, and J. Salvi, "Evaluation of registration methods on two-dimensional forward-looking sonar imagery," *OCEANS 2013 MTS/IEEE Bergen: The Challenges of the Northern Dimension*, 2013.

[29] M. D. Aykin and S. Negahdaripour, "On Feature Matching and Image Registration for Two-dimensional Forward-scan Sonar Imaging," *J. Field Robotics*, vol. 30, pp. 602–623, 2013.

[30] P. Biber and W. Strasser, "The normal distributions transform: a new approach to laser scan matching," *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2743–2748, 2003.

[31] M. Kaess, H. Johannsson, B. Englot, F. Hover, and J. J. Leonard, "Towards Autonomous Ship Hull Inspection using the Bluefin HAUV," *Symposium on Technology and the Mine Problem*, 2010.

[32] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. J. Leonard, "Advanced perception, navigation and planning for autonomous in-water ship hull inspection," *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1445–1464, 2012.

[33] B. Zitová and J. Flusser, "Image registration methods: A survey," *Image and Vision Computing*, vol. 21, no. 11, pp. 977–1000, 2003.

[34] J. P. Lewis, "Fast Normalized Cross-Correlation Template Matching by Cross-," *Vision Interface*, vol. 1995, no. 1, pp. 1–7, 1995.

[35] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós, "A comparison of loop closing techniques in monocular SLAM," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1188–1197, 2009.

[36] J. Sawas, "Automatic Target Recognition in Sonar Imagery Using a Cascade of Boosted Classifiers," no. May, 2015.

[37] J. Neira and J. Tardos, "Data Association in Stochastic Mapping Using the Joint Compatibility Test," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 890–897, 2001.

[38] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardos, "Mapping Large Loops with a Single Hand-Held Camera," *Robotics: Science and Systems III*, 2007.

[39] D. Ribas, P. Ridao, J. Domingo Tardós, and J. Neira, "Underwater SLAM in Man-Made Structured Environments," *Journal of Field Robotics*, vol. 25, no. 1, pp. 888–921, 2008.

[40] J. Sivic and A. Zisserman, "Video Google: a text retrieval approach to object matching in videos," *IEEE International Conference on Computer Vision*, pp. 1470–1477, 2003.

[41] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[42] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós, "An image-to-map loop closing method for monocular SLAM," *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pp. 2053–2059, 2008.

[43] S. Thrun and J. J. Leonard, "Simultaneous Localization and Mapping," *Springer handbook of robotics*, pp. 871–889, 2008.

[44] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[45] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35—-45, 1960.

[46] J. Aulinas, Y. Petillot, J. Salvi, and X. Lladó, "The SLAM problem: A survey," *Frontiers in Artificial Intelligence and Applications*, vol. 184, no. 1, pp. 363–371, 2008.

[47] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, "Simultaneous localization and mapping with sparse extended information filters," *International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 693–716, 2004.

[48] M. R. Walter, R. M. Eustice, and J. J. Leonard, "Exactly sparse extended information filters for feature-based SLAM," *International Journal of Robotics Research*, vol. 26, no. 4, pp. 335–359, 2007.

[49] M. Walter, F. Hover, and J. Leonard, "SLAM for ship hull inspection using exactly sparse extended information filters," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1463–1470, 2008.

[50] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/nongaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 723–737, 2002.

[51] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," *Proc. of 8th National Conference on Artificial Intelligence*, vol. 68, no. 2, pp. 593–598, 2002.

[52] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2 .0 : An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges," *Proceedings of IJCAI 2003*, pp. 1151–1156, 2003.

[53] M. Montemerlo and S. Thrun, "FastSLAM 2.0," *Springer Tracts in Advanced Robotics*, vol. 27, pp. 63–90, 2007.

[54] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A Tutorial on Graph-Based SLAM," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[55] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[56] V. Ila, L. Polok, M. Solony, and P. Svoboda, "Highly Efficient Compact Pose SLAM with SLAM++," *arXiv preprint arXiv*, no. 2010, pp. 1–21, 2016.

[57] H. Foroosh, J. B. Zerubia, and M. Berthod, "Extension of phase correlation to subpixel registration," *IEEE Transactions on Image Processing*, vol. 11, no. 3, pp. 188–199, 2002.

[58] J. Ren, J. Jiang, and T. Vlachos, "High-Accuracy Sub-Pixel Motion Estimation From Noisy Images in Fourier Domain," *IEEE Transactions on Image Processing*, vol. 19, no. 5, pp. 1379–1384, 2010.

[59] M. Mansourpour, M. Rjabi, and J. Blais, "Effects and Performance of Speckle Noise Reduction Filters on Active Radar and SAR Images," *International Journal of Technology And Engineering System (IJTES)*, vol. 2, no. 1, pp. 111–114, 2011.

[60] S. Karabchevsky, *Real-Time Underwater Obstacle Detection using Forward Looking Sonar and FPGA*. Msc thesis, Ben-Gurion University of the Negev, 2011.

[61] 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. 4, no. 2, pp. 157–166, 1982.

[62] H. S. Stone, M. T. Orchard, E. C. Chang, and S. A. Martucci, "A fast direct Fourier-based algorithm for subpixel registration of images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 10, pp. 2235–2243, 2001.

[63] M. Balci and H. Foroosh, "Subpixel estimation of shifts directly in the Fourier domain," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1965–1972, 2006.

[64] F. Meyer and S. Beucher, "Morphological segmentation," *Journal of Visual Communication and Image Representation*, vol. 1, no. 1, pp. 21–46, 1990.

[65] B. Srinivasa Reddy and B. Chatterji, "An FFT-Based Technique for Translation, Rotation, and Scale-Invariant Image Registration," *IEEE Transactions on Image Processing*, vol. 5, no. 8, pp. 1266–1271, 1996.

[66] N. Hurtós, X. Cufí, and J. Salvi, "Rotation Estimation for Two-Dimensional Forward-Looking Sonar Mosaicing," in *ROBOT2013: Advances in Robotics, Vol. 1*, pp. 69–84, Springer International Publishing, 2014.

[67] K. Cassee, *Wheel slip and orientation drift correction for the relative localization system of a ship hull cleaning robot.* Msc thesis, Delft University of Technology, 2018.

[68] Blueview, "Proviewer 4 Software Handbook." URL to handbook: http://www.teledynemarine.com/Lists/Downloads/ProViewer-4-Software-Handbook.pdf. (Accessed 21-May-2018).

[69] Xsens Technologies, "MTi 10-series and MTi 100-series." URL to manual: https://content.xsens.com/mti-10-manual. (Accessed 20-July-2017).

[70] Kuebler Group, "RLI200." URL to datasheet: https://www.kuebler.com/k2016/pdf?RLI200_en.pdf. (Accessed 12-July-2017).

[71] STS Sensors, "ATM.1ST/N - Submersible level transmitters." URL to datasheet: https://aesensors.nl/assets/uploads/2018/04/Datasheet-ATM.1ST_N_Ex-High-Precision_kl.pdf. (Accessed 21-May-2018).

[72] STS Sensors, "ATM.1ST - Pressure transmitters." URL to datasheet: https://aesensors.nl/assets/uploads/2018/04/Datasheet-ATM.1ST-High-Precision-Transmitter_kl.pdf. (Accessed 21-May-2018).

[73] TWK Elektronik, "Electromagnetic absolute encoders TBX36." URL to datasheet: http://www.twk.de/data/pdf/11713de0.pdf. (Accessed 12-July-2017).

[74] D. Borota, "Robot dimensions used for software development," tech. rep., Fleet Cleaner, 2017.

[75] Kharagpur, Indian Institute of Technology, "Frost filter." URL to source code: https://nl.mathworks.com/matlabcentral/fileexchange/35073-frost-s-filter. (Accessed 16-April-2018).

Glossary

List of Acronyms

AUV	Autonomous Underwater Vehicle
DOF	Degree of Freedom
EKF	Extended Kalman Filter
ENL	Equivalent Number of Looks
ESEIF	Exactly Sparse Extended Information Filter
FFT	Fast Fourier Transform
FLS	Forward Looking Sonar
GPS	Global Positioning System
IFFT	Inverse Fast Fourier Transform
IMU	Inertial Measurement Unit
NDT	Normal Distribution Transform
MSE	Mean Square Error
PCS	Phase Correlation Surface
PSR	Peak to Sidelobe Ratio
RANSAC	RANdom SAmple Consensus
SEIF	Sparse Extended Information Filter
SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SNR	Signal-to-Noise-Ratio
SURF	Speeded Up Robust Features