# Sea Ice Field Analysis Using Machine Vision

Andrei Sandru

**School of Electrical Engineering**

Master's thesis.

Espoo 17.09.2018

**Thesis supervisor:**

Prof. Arto Visala

**Thesis advisor:**

M.Sc. Heikki Hyyti

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Andrei Sandru

Title: Sea Ice Field Analysis Using Machine Vision

Date: 17.09.2018 Language: English Number of pages: 7 + 89

Department of Electrical Engineering and Automation

Professorship: Automation technology Code: AS-84

Supervisor: Prof. Arto Visala

Advisor: M.Sc. Heikki Hyyti

Sea ice field analysis has motivation in various areas, such as environmental, logistics or ship maintenance. Among other methods, local ice field analysis from ship-based visual observations are currently done by human volunteers and therefore are liable to human errors and subjective interpretations.

The goal of the thesis is to develop and implement a complete process for obtaining dimensions, distribution and concentration of sea-ice floes, which aims at assisting and improving part of the aforementioned visual observations. Such process involves numerous, organized steps which take advantage of techniques from image processing (lens calibration, vignetting removal and orthorectification), robotics (transformation frames) and machine vision (thresholding and texture analysis methods, and morphological operations).

An experimental system setup for collecting the required information is provided as well, which includes a machine vision camera for image acquisition, an IMU device for determining the dynamic attitude of the cameras with respect to the world, two GPS sensors providing a redundant positioning and clock data, and a desktop computer used as the main logging platform for all the collected data.

Through a number of experiments, the proposed system setup and image analysis methods have proved to provide promising results in pack ice and brash ice conditions, thus encouraging further research on the topic. Further improvements should target the accuracy of ice floes detection, and over and under-segmentation of the detected sea-ice floes.

Keywords: sea ice field analysis, attitude estimation, image processing, machine vision

# Preface

First I would like to thank Professor Arto Visala for presenting me with the opportunity of an equally challenging and rewarding subject for my master's thesis, in addition to his never-ending patience and support throughout the entire process. I would also like to thank instructor Heikki Hyyti for all the constructive talks, guidance, support and comments on improving the structure and contents of this thesis.

Special thanks go to Pentti Kujala from the Marine Technology department at Aalto University and Dr. Annie Bekker from Stellenbosch University, who made possible a once in a lifetime trip for data collection. I would also like to thank Jakke Kulovesi for his assistance during the project, Miko Suomalainen for helping with the on-site system setup and Keith Soal for logistics and presenting the beauty of Cape Town.

And last but not least, I would like to thank all the people that in one way or another gave their support for achieving the results presented in the present work.

Otaniemi, 17.09.2018

Andrei Sandru

# Table of Contents

# List of acronyms and symbols

| | |
|---|---|
| API | Application Program Interface |
| DCM | Direct Matrix Cosine |
| DOF | Degrees Of Freedom |
| DT | Dynamic Thresholding |
| FMI | Finnish Meteorological Institute |
| FOV | Field Of View |
| GCP | Ground Control Points |
| GNSS | Global Navigation Satellite System |
| GPS | Global Positioning System |
| GPU | Graphics Processing Unit |
| GVF | Gradient Vector Flow |
| IMU | Inertial Measurement Unit |
| INS | Inertial Measurement System |
| MEMS | Microelectromechanical Systems |
| PNT | Positioning Navigation and Timing |
| ROI | Region Of Interest |
| SAR | Synthetic-Aperture Radar |
| SDK | Software Development Kit |
| TA | Texture Analysis |
| UTC | Coordinated Universal Time |

| | |
|---|---|
| $c_x$, $c_y$ | Image centre point in x and y directions. |
| ct | Centroid in the K-means and Dynamic Thresholding algorithms. |

| | |
|---|---|
| °C | Degrees centigrade. |
| DV | De-Vignetting matrix. |
| e | Entropy value. |
| $f$ | Focal length in mm. |
| g | A function. |
| G | A class in the K-means and Dynamic Thresholding algorithms. |
| h, $h$ | Height of a camera with respect to a horizontal plane. |
| H | Homography matrix. |
| I | Intensity image. |
| j | Observation. |
| J | Data set. |
| $k_1$, $k_2$ | First and second radial distortion coefficients. |
| K | Camera matrix containing intrinsic parameters. |
| L | Image containing labelled sea-ice floes. |
| m | Mean value. |
| M | Rotation matrix provided by the Inertia-Link® IMU device. |
| N | Total number of elements in a series. |
| $p_1$, $p_2$ | First and second tangential distortion coefficients. |
| q, Q | A point projection in the camera image frame and the same point in world frame. |
| $q_{x,y}$ | Value of a pixel point at coordinates (x,y). |
| R | A rotation matrix. |
| s, S | Scaling factors. |
| t | A translation vector. |
| T | Thresholding value. |
| $v$ | Horizontal physical size of the imaging sensor in a camera. |

| | |
|---|---|
| U, V | Image dimensions in the horizontal and vertical axis. |
| x, y, z | Coordinates in image plane. |
| X, Y, Z | Coordinates in World frame. |
| $\tau$ | Optical Field of View angle. |
| $\lvert\overrightarrow{qc}\rvert$ | A vector magnitude. |
| $\mu$ | Physical size of a pixel. |
| $\alpha$ | Angular aperture between the principal axis and segment formed by the center of projection and a point in image plane, in the y-axis. |
| $\gamma$ | Angular aperture between the principal axis and segment formed by the center of projection and a point in image plane, in the x-axis. |
| $\phi$ | Roll angle in degrees. |
| $\theta$ | Pitch angle in degrees. |
| $\psi$ | Yaw angle in degrees. |

# 1. Introduction

## 1.1 Objectives and scope

The present thesis' main goal was set to automate part of the visual observations which have been carried on over the years, by correctly identify ice floes boundaries in front of an ice breaker and obtain statistics such as their dimensions, area and coverage or concentration. Ice floes are a part of the surface visible ice field and the result of larger sea-ice sheets, which fracture due to winds and waves. They are commonly found in the Arctic and Antarctic regions and their dimensions vary from few tens of meters up to more than 10km across.

To achieve the aforementioned goal, an extensive process is required, which takes care of preparing or pre-processing the images. Then, those images need to be analysed by means of machine vision techniques and algorithms, which in the present work are narrowed down to thresholding based approaches using intensity and texture based features. The data required to develop and test algorithms was collected on board of the ice breaker S.A. Agulhas II, during its relief voyage to Antarctica 2017-2018.

Ice field analysis has motivation in multiple areas, such as environmental (e.g. determine seasonal changes in ice coverage and properties, or improve weather models), logistics (e.g. path planning through an ice field), or maintenance of ships. It is the latter one of special relevance for the present work, since cruising through an ice field can have severe impact on the structural integrity of a ship's hull. This makes a correct evaluation of the ship's status and maintenance crucial, especially when it is destined to cruise through remote areas.

## 1.1 Background on ice field analysis methods in Antarctic waters

Ice field analysis may be performed by different means. In each case, however, because of the non-uniformity and even arbitrariness of the measured natural environment, a number of challenges arise. One such challenge is the proper segmentation of ice floes from their background (slush and water). Another challenge is the proper division between adjacent ice floes, which may be difficult even for a human observer, yet alone for machine based techniques. Some of the means and their methods are presented next.

For instance, ship-based visual observation have been carried out in the past, ever since the days of the earliest explorers Cook and Bellingshausen [1] and encounters with sea ice have been recorded in the ship's log books. However, the sparsity and irregularity of such data makes it hard to obtain meaningful conclusions. In more recent years, such observations have been aimed to be

standardized, for example in [2] a detailed procedure for making ship-based observation is given. Such standardizations greatly improve the quality of the data recorded and their usefulness as validation data for the latter presented methods, however it might fall short for certain applications and may still be biased, since they are based on a human observer.

Satellite based ice field analysis are another manner of obtaining characteristics about the ice field, classified under remote sensing. In [3], the authors propose a thresholding method using values from a bi-Gaussian distribution over the grey levels in the satellite image. In [4], the authors use Synthetic-Aperture Radar (SAR) satellite images together with the watershed algorithm in order to segment the images and obtain ice floes characteristics. However, they noted that the watershed algorithm tends towards over-segmentation, which they tried to compensate through region merging based on intensity and that may introduce the opposite effect as well, by joining separate ice floes with similar intensity values. Satellite based ice field analysis may provide easy to compare results among measurements, however those measurements have a low spatial and temporal resolutions.

Ice field analysis may be performed on a local scale using aerial images or ship-based. Examples of the first case can be found in [5] and [6], where the authors use a combination of thresholding and morphological operations in order to detect sea ice floes. In the second example, the authors use the K-Means algorithm (explained later on in the methods parts of the present work) in order to automate the thresholding method and in addition, they propose a modified version of the snake algorithm called Gradient Vector Flow snake [7] for improving the outer boundaries of the detected ice floes. Unfortunately, their results are given only in pixel dimensions and are not compared with a ground truth.

It is the latter case, ship-based sea-ice field analysis of special relevance for the present work. In [8], the authors use a video camera to capture images, which are then digitized and a simple thresholding method applied either online for ice concentration (single row analysis at a determined distance from the ship) or offline, which already includes an orthorectification method and provides rough floe shapes. Their results were modest, since the development of digital cameras and computers was at an early stage. Next, in [9] the authors explore a full process of sea-ice parameters acquisition from digital images which includes orthorectification through Ground Control Points (GCP) and Delauney Triangulation [10]; image processing through band thresholding, unsupervised classification (K-means algorithm), Region Of Interest (ROI) delineation and directional convolution filters to extract textural information from images are presented and compared to human estimates. Their results showed good agreement between their

image analysis techniques and human observations, however the process presented is not fully automatic and the orthorectification method can greatly be improved.

In the present work, a number of ideas from the aforementioned examples are implemented, such as thresholding by K-Means algorithm and texture analysis. In addition, a number of novelty approaches are introduced as well, such as the use of an IMU device and robotics transformation frames in the orthorectification process for ship-based sea ice field analysis and an autonomous thresholding method based on the K-Means algorithm called Dynamic Thresholding.

In the ice field analysis, the classification is done in different categories of sea ice forms, such as pancake ice (circular pieces with raised rims), brash ice (ice fragments with diameters not larger than 2 metres), ice cake (floating flat ice smaller than 20 metres across) or pack ice (flat floating ice floes larger than 20 metres across). In the present work, only pack ice conditions (individually identifiable and flat sea ice floes) and brash ice conditions (ice floes surrounded by brash ice) are considered. Examples of both conditions are presented in Figure 1.

Figure 1. Examples of pack ice conditions (left image) and brash ice conditions (right image).

## 1.2 Structure

The work is structured as follows: in *Chapter 2*, the proposed instrumentation required for the present work and its installation on a ship is presented. In addition, the characteristics of the sensors are described, as well as their calibration process. Lastly, the overall coordinate system used is presented.

Next, in *Chapter 3* a number of image processing techniques are presented, conforming a complete process for performing sea-ice field analysis. Each subsection starts with a problem statement, followed by a background research on the state-of the art and continues with a description of the selected method or methods.

In *Chapter 4*, previously described methods for image orthorectification are aimed to be improved through attitude estimation, provided by an IMU device. The chapter starts with a background on sensor fusion between image and IMU, then continues with a background and methods selected on attitude estimation from IMU, and ends with a description of the Image-IMU sensor fusion process.

In *Chapter 5* the results of the described methods in *Chapters 3* and *4* are presented. Then, in *Chapter 6* the methods and their results are discussed, as well as their limitations.

Lastly, *Chapter 7* presents an overall conclusion on the present thesis work and recommendations on future work.

A total of three appendices are attached at the end. *Appendix A* depicts additional figures of the installation of sensors on the ship. *Appendix B* presents the reader with an extract depicting surface albedos as a function of light wavelength and *Appendix C* includes additional results figures from *Section 5.3: Experiment C.*

# 2. Field instrumentation work

In the present chapter, at first a full system disclosure is presented of the instrumentation proposed and later on, performed on the S.A. Agulhas II during its relief voyage 2017-2018. In addition, a theoretical background on the type of sensors proposed is introduced, together with a description of the exact ones which were installed, followed by their calibration procedures. Next, their installation on board of the vessel is shown and the structure of the collected data is presented and commented. Lastly, the coordinate system is introduced and its application to the system setup described.

## 2.1 System setup (experimental setup)

As main sensors for measuring sea-ice floes, a pair of machine vision cameras are proposed in a stereo setup, which can provide dense and precise information about the environment with high data rates. Since a camera is an exteroceptive (i.e. reacts to environmental stimuli) and passive (measures environmental energy, without altering it) sensor [11], it does not affect the environment by any means, which is crucial when navigating through protected areas such as Antarctica [12]. One additional important aspect is that remote sensing does not interfere in any case with the ship's own mission.

Together with the main sensors proposed, two additional sensors are required, namely an IMU and a Global Navigation Satellite System (GNSS) sensor (e.g., a GPS sensor). On one hand, the IMU is essential in determining the attitude of the ship and cameras with respect to the fixed earth coordinate system, which in turn permits geometrically rectifying images without a "wobbling" effect due to the natural movement of the ship. On the other hand, by installing a GNSS sensor, images (and the data contained within them) can retroactively be position and time stamped with satellite accuracy, broadening their application area.

All together, the proposed setup forms a loosely-coupled system [13], where each sensor works independently from one another and measures different physical quantities, which are processed individually. However, their measurements can be fused together through a timestamp provided from a single clock source (i.e. PC clock).

An overview of the system arrangement is presented in Figure 2, which includes the cameras, IMU and GPS sensors, and computer.

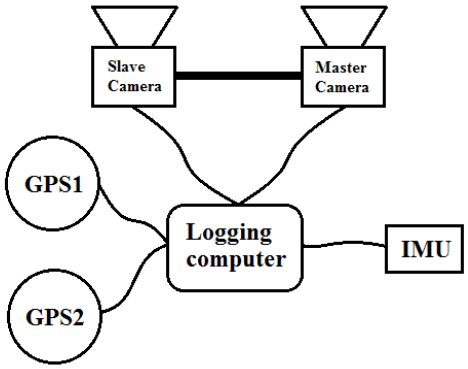


Figure 2. Abstract system overview.

As can be observed in Figure 2, a dual GPS antenna setup is proposed. Among other reasons, the main ones would be redundancy (against signal or device loss) and, by installing them at different locations on the ship, their accuracy and precision may be determined by comparing them to the ship's own GPS, which is assumed to provide higher quality measurements regarding the ship's position.

## 2.2 Sensors

In the present section, a theoretical description of the proposed type of sensors is provided. Following each subsection, the exact sensor model installed is introduced and described to the extent of requirements fulfilment.

### 2.2.1 Digital and machine vision cameras

For the last 30 years, digital cameras and photography have experienced an explosion in image quality, features, reduced size and price, to mention just a few, and the trend is nowhere near stopping [14] [15].

A digital camera works somewhat similar to the human eye, whereas instead of cones and rods there is a matrix array on a silicon chip, comprised of a large, ever-increasing number of photosites (i.e. pixels) which are sensible to light. The environment light is processed through a series of lenses before reaching the sensor, where it is integrated to form an image, effectively discretizing the scenery. An example of the image formation process through a camera perspective projection of a single lens is presented in Figure 3. The larger the amount of photosites or pixels, the larger the amount of details an image theoretically contains, although other factors discussed later may greatly influence it.

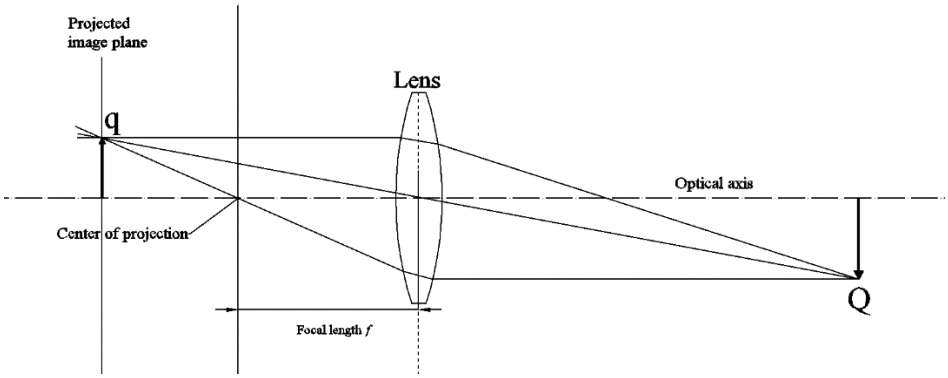


Figure 3. Image formation from perspective projection through a single lens.

A number of important camera characteristics need to be taken into account when determining the requirements for a specified purpose. The most important among these characteristics are shortly described next.

The first distinction to be made is whether a colour or monochrome sensor is needed. Similar to the human eye, a monochrome camera presents higher sensitivity to light than a colour sensitive one, hence it is more suitable for environments with diminished luminosity. However, colour information can be crucial and present numerous advantages in machine vision applications.

Another important aspect is shutter speed, which can be mechanical (physically obstructing the light from reaching the sensor die) or electronic. In both cases, the end result is to limit the amount of time the light is allowed to be integrated by the sensor. Longer integration times result in images with higher luminosity, however at the expense of introducing higher motion blur (photons reaching the same pixel may not be emitted from the exact same physical location), and therefore distorting the image. One additional source of distortion is created by the effect called *rolling shutter* [16], in which the sensor is read one line at a time. In circumstances with high vibration levels a wobbling may occur. Therefore, a global shutter is desirable, where the image is captured at once.

In a similar fashion, the aperture value determines the amount of light with respect to time that reaches the sensor by increasing or decreasing a hole opening in front of the sensor. A small aperture (large f-value, e.g. f/24) value will increase the depth of field (i.e. objects at different distances from the sensor are in focus), while a larger (smaller f-value, e.g. f/2.8) aperture value allows more light to enter the camera and therefore is suitable where high shutter speeds or low light conditions are present, at the cost of a reduced depth of field or focus distance, explained later on.

Continuing with aspects related with image luminosity, gain is an artificial increase in image luminosity, done at sensor level by increasing its sensitivity to light. However, this comes at the expense that noise will be amplified as well, hence a minimum amount of gain is desirable.

Lastly, it is worthwhile mentioning and discussing focus and lens focal length from a camera's point of view. The ideal camera model or pinhole model presents an infinite focal range, meaning that light rays entering the camera from different directions do not mix. This is not the case with real cameras, where there is always a focus range or distances. The light rays coming from objects located in this range do not mix, however outside the range they do, creating a blurring effect.

The above parameters are common for commercial and machine vision cameras. However, machine vision cameras present additional specialized features. Two such features are of particular interest in this case and the reason for using machine vision cameras for the present thesis work. The first one is the presence of an Ethernet data port, which allows high speed data transfers and makes it possible to capture and transmit uncompressed images (i.e. raw). The second feature is external triggers for image capture and exposure time, making it possible to tightly synchronize two or more cameras.

For the present thesis, a pair of Basler aviator model avA2300-25gc were used, each coupled with a Tamron 16mm fixed lens. Table 1 sums up the most important camera setup characteristics. For a full product description, refer to [17].

Table 1. Camera setup characteristics.

| | |
|---|---|
| Shutter type | Global shutter |
| Sensor size (horizontal x vertical) | 12.8 mm x 9.6 mm |
| Resolution (horizontal x vertical) | 2330px x 1750px (2330px x 1440px used) |
| Pixel size | 5.5 µm x 5.5 µm |
| Mono/Colour | Colour |
| Interface | GigE |
| Pixel Bit Depth | 12 bits |
| Synchronization | Hardware trigger<br>free-run<br>Ethernet connection |
| Exposure control | programmable via API<br>hardware trigger |
| Lens | Tamron 16 mm |

### 2.2.2 Inertial Measurement Unit

An Inertial Measurement Unit (IMU) is an electronic device which, attached to a body, is capable of measuring its specific force, angular rate and in some cases, the surrounding magnetic field. To do so, it uses a combination of multi-axis accelerometers, gyroscopes and where applies, magnetometers. As with many other developments, military applications were among the first to thrive and make use of IMU devices, for instance in aiding 1954 fighter pilots with navigation [18] or later on in the control of guided missiles [19].

Nowadays however, IMUs can be found in a large variety of applications (military and civilian), due to recent developments in the fabrication of microelectromechanical systems (MEMS) which have made them extremely affordable and more reliable [20]. Examples of such applications include attitude estimation in smartphones [21], Virtual Reality headsets [22], ships [23], satellites [24], or even body motion capture [25].

The data obtained from an IMU, when integrated, can be used to track a body's trajectory and current position, given an initial known position and using a method named dead reckoning. However, one major drawback of IMU sensors is that they suffer from accumulated error: since dead reckoning principle implies a continuous integration over time, any and all errors are accumulated. This behaviour causes a low-frequency drift, a continuous increase in the difference between the actual attitude and/or position of the system and what the calculus results tell. In addition, MEMS devices introduce additional bias and errors, which are temperature dependent [26].

Most recent developments combine IMU with GNSS data [27][28], commonly by means of a Kalman filter or one of its derivatives. In this kind of arrangements, IMU data is utilized to estimate the attitude and/or position of a body in between updates from a GNSS sensor. Then, GNSS data aids in compensating for the drift present in the output from the Inertial Measurement System (INS).

The IMU utilized to determine the attitude of the ship and cameras was an Inertia-Link 4200-1076, which provides 6-DOF (six degrees of freedom) due to its 3-axis gyroscope and 3-axis accelerometer. In addition, it can output a 3x3 estimated rotation matrix, with values fully compensated for temperature [29]. Specifications and datasheet of the device can be found in [30].

This section continues first with the sensor calibration in *Section 2.3.2* and then with attitude estimation in *Section 4.1*.

### 2.2.3 Global Navigation System

The Global Navigation System (GPS) [31] is a utility owned by the U.S. and which can provide positioning, navigation and timing (PNT) all around the world. As with IMUs, it was initially aimed at military applications and afterwards released for civilian application in the 1980s. Nowadays, a great number of civilian application areas rely on GPS data for their correct or improved operation. A limited example of such areas includes agriculture, marine, search and rescue missions, recreation, mapping or timing. A brief working principle is presented next.

The GPS system [31] is conformed of a number of satellites (31 at the time of writing this thesis, expected to be expanded in the near future) orbiting the Earth at a height of approximately 20200 km. Each satellite is equipped with a highly precise atomic clock, synchronized among satellites and with ground stations. This clock, together with the satellite's own position is continuously broadcasted to Earth, where a GPS-enabled device can calculate its position or synchronize its clock with respect to the Coordinated Universal Time (UTC). To do so, a clear path to at least four satellites is required, from whose data a total of four unknowns are calculated (the receiver's latitude, longitude, altitude and clock bias with respect to satellite clock). In optimal conditions (clear sky and direct path to the satellites), a smartphone may achieve an accuracy of up to 4.9m [31]. Although satellites output highly accurate data, the signals may suffer from atmospheric distortion or other ground distortions (such as blocked or reflected signals, the later causing an effect called "multipath"), effectively degrading accuracy. Additional signal carriers help mitigating the effect of such distortions, however capable receivers are usually bulky and expensive, relegating their use for professional applications only.

For the current thesis, the most important aspect from the GPS utility is timing: correctly synchronizing the computer clock to UTC enables the recorded data to be related with other data sources, also external to the ship (e.g. satellite images). For this purpose, two distinct and low-cost GPS receivers were used, namely a u-blox® LEA-5H and a GlobalSat® BU-353S4. Their specifications can be found at [32] and [33], respectively.

## 2.3 Sensor calibration

In the current subsection, the models assumed for the cameras and IMU devices are presented, followed by their calibration procedures and calibration results.

### 2.3.1 Camera calibration

#### *2.3.1.1 The* camera *model*

As mentioned earlier, a camera captures and discretizes the scenery, which can be considered as a mapping between the 3-dimensional world and the captured 2-dimensional image through a camera model. If such mapping is known, the inverse process becomes feasible: information about the scenery can be extracted from its 2-dimensional image. The high complexity of obtaining the mapping model constrains it to only an approximation, therefore some amount of error will still remain.

For a parametric camera model, its parameters can be optimized through a calibration process in order to obtained the best possible mapping. In the present work, the chosen and described camera model is based on perspective projection, and more specifically, the pinhole camera model [34], which is depicted in Figure 4.

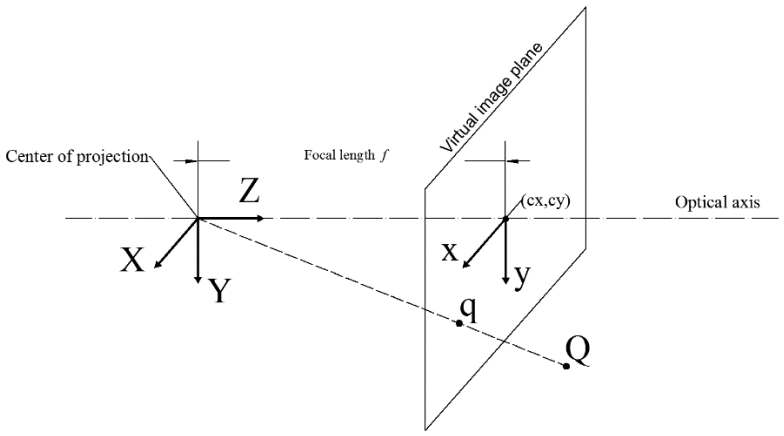


Figure 4. Pinhole camera model.

The pinhole camera model is a commonly used mapping model, in which a projected point $q$ with pixel coordinates $(x,y)^T$ in the image plane is related to a point $Q=[X,Y,Z]^T$ in the 3-D world by:

$$x = S_x f_x \frac{X}{Z} + c_x$$

$$y = S_y f_y \frac{Y}{Z} + c_y \quad (1)$$

where $S_x, S_y$ are scaling factors given by the physical dimensions of a pixel, $f_x, f_y$ are the focal distances and $(c_x, c_y)$ are the coordinates (in pixels) of the image centre. This can be rewritten as a projective mapping:

$$q = \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = CQ \tag{2}$$

where $s$ is an arbitrary scaling factor and matrix $C$ is the projection matrix expanded in (3), and containing the camera intrinsic parameters (matrix $K$) and extrinsic (camera pose in world coordinates given by *[R t],* where $R$ is a rotation matrix and $t$ is a translation vector).

$$C = [K]x\,[R\ t] = \begin{bmatrix} \frac{f_x}{S_x} & 0 & c_x & 0 \\ 0 & \frac{f_y}{S_y} & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x\,[R\ t] \tag{3}$$

This section continues in the following subsection, and is built upon as well in *4.2: IMU-Image fusion*.

*2.3.1.2 Lens distortion estimation*

In an ideal world, the image formed on the camera's sensor would be an exact representation of the captured scenery. Nevertheless, this is not the case within the actual world we live in: captured images are altered by a number of artefacts, e.g. dead pixels, previously explained vignette effect, noise, blooming or lens distortion [35]. All of them degrade the image quality and alter the image to different degrees. Because of the operation of geometric transformation used at a later stage, it is of high importance to address and correct the lens distortion effect, since a small alteration in the original image may be enlarged in the processed one.

The first step is to obtain the camera intrinsic parameters, which can be done through the Matlab camera calibrator app as described in [36]. Images were captured of a calibration target composed of 10x6 black and white, 100mm side squares, at different angles and distances. The camera calibrator app would firstly detect the feature points in the images (intersection corners between four connected black and white squares); then, using the closed form solution proposed by Zhang [37] it would estimate the five intrinsic camera parameters and extrinsic parameters described in the previous chapter. Lastly, it would refine found estimates by minimization (maximum-likelihood estimation). The resulting parameters from the calibration are presented in Table 2.

Table 2. Estimated camera model parameters

| | | |
|---|---|---|
| U<br>V | 2332 pixels<br>1440 pixels | Image total width.<br>Image total height. |
| $f_x/S_x$<br>$f_y/S_y$ | 2933.706 pixels<br>2933.684 pixels | Camera focal length in *x-axis*.<br>Camera focal length in *y-axis*. |
| $c_x$<br>$c_y$ | 1173.042 pixels<br>895.998 pixels | Image centre in *x-axis*.<br>Image centre in *y-axis*. |
| $k_1$<br>$k_2$ | -0.154490<br>0.9605035 | Second degree radial distortion coefficient.<br>Fourth degree radial distortion coefficient. |
| $p_1$<br>$p_2$ | $9.855 \times 10^{-4}$<br>$-7.016 \times 10^{-4}$ | First tangential distortion coefficient.<br>Second tangential distortion coefficient. |

By using the Matlab function *undistortImage*, together with the previously estimated camera parameters, captured images can be corrected for radial and tangential distortions. Such correction function follows the equation:

$$x' = x(1 + k_1 r^2 + k_2 r^4) + 2p_1 xy + p_2(r^2 + 2x^2)$$
$$y' = y(1 + k_1 r^2 + k_2 r^4) + 2p_2 xy + p_1(r^2 + 2y^2) \quad (4)$$

where *(x,y)* are the coordinates of a pixel in the original image, *(x',y')* are the coordinates of the same pixel in the rectified image, $r = \sqrt{x^2 + y^2}$, $(k_1, k_2)$ are the radial distortion coefficients and $(p_1, p_2)$ are the tangential distortion coefficients.

An example of a performed correction is presented in Figure 5.

Figure 5. Original image (left), undistorted image (centre) and combination of both images with differences marked in green/pink (right).

*2.3.1.3 Vignette estimation*

Vignetting effect refers to a systematic flaw which affects optical rays with larger span-off angle from the camera's principal axis by attenuating their intensity. In other words, pixels which are

further from the image's optical centre tend to become darker, with a peak attenuation effect on the images' corners [38]. There can be various reasons for the vignette effect to occur, including mechanical, optical, natural or pixel related vignette [39]. Since it is a systematic optical defect which can be modelled, commercial cameras may include an in-camera de-vignetting option, with pre-loaded parameters for different lenses [40].

In order to estimate the vignette effect and obtain a de-vignetting mask, a total of two methods are proposed: one ready-made and publicly available and a third method developed in this work.

The first algorithm tackled is proposed by Zheng et al. [41]. Their method, aimed at estimating vignetting effect on a single image, is based on the assumption that the radial gradient (gradient lengthwise from the image centre) has a symmetric distribution for images without vignetting effect and it is skewed for those which present a vignetting effect. Once the asymmetry of the radial gradient has been identified, they propose two methods to remove it: one of them uses a least-squares approach at discrete intervals along the radius to determine the amount of vignetting and remove it, while the other tries to fit a vignetting model to the image using non-linear optimization. As mentioned earlier, a ready-made implementation of their algorithm was used, which can be found at [42].

This method does not require access to the camera used to capture the image, nor the camera's specifications; however, assuming that both of them are available, a completely different approach can be explored. Based on selected ideas from [43], [41] and [44], the following assumptions are made: the vignette effect can easily be extracted from a flat, texture less surface, and it is radial around the optical centre of the image.

An assumption is made that in addition to aiming the camera to a flat, texture-less surface, the vignetting effect can be empirically obtained by covering the lens of the camera with a semi-transparent, homogeneous material and in controlled lighting conditions. A number of images are captured in this manner, converted to grey scale and then pixel-wise summed and divided by the number of images, effectively obtaining an averaged de-vignetting mask. This operation is presented in (5), where $I^n$ is the $n^{th}$ image of the set, $N$ is the total amount of images in the set, $(x,y)$ are pixel coordinates in the standard image coordinate system and $DV$ is the obtained de-vignetting mask. The vignette effect can be easily removed from an image by simply pixel-wise dividing it by the de-vignetting mask, followed by a normalization operation.

$$DV_{x,y} = \frac{1}{N}\sum_{n=1}^{N} I_{x,y}^{n} \qquad \forall x \in I^n, \forall y \in I^n \tag{5}$$

However, a more formal method is required/desired. The proposed method is based on the aforementioned assumption of a radial gradient vignetting effect, known intrinsic parameters of the camera (specifically, the optical centre of an image) and the above method of empirically obtaining a de-vignetting mask, which will henceforth be referred to as simply *DV*.

In order to test the assumption of a radial gradient, a plot of pixel intensity versus radial distance to the optical centre is obtained. Radial distance is calculated as Euclidean distance between two pixels. Furthermore, the image is divided in four sectors and for each sector the pixel intensity versus radial distance is plot, as seen in Figure 6.

From Figure 6, it can be seen that the vignette effect behaves in an approximately radial fashion. Next, a parametric equation is fit to the data from Figure 6 in Matlab, namely a smoothing spline with a '*SmoothingParam*' set to 0.0005, and the results can be seen in the same figure superimposed as a red line.

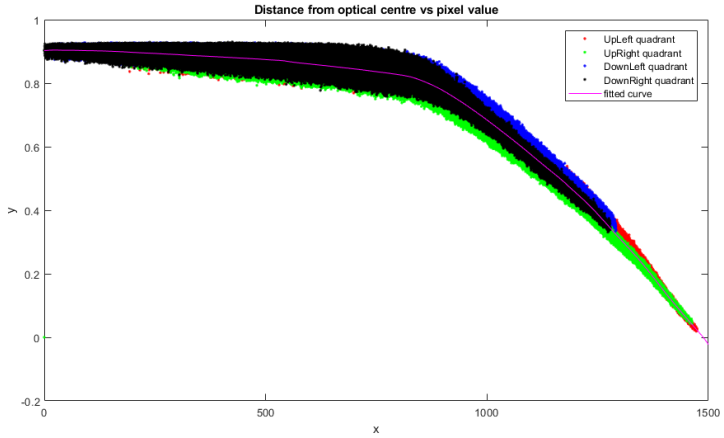


Figure 6. De-vignetting mask (DV) plot as pixel intensity vs. radial distance to the optical centre. Superimposed, the parametric equation *smoothing spline* fit to the pixel intensity vs. radial distance is plot as a red line.

Lastly, the inverse procedure is followed to obtain a de-vignetting mask: The Euclidean distance of the pixels of a grey image *I* are fed into the fit parametric equation, obtaining their pixel value as an output.

A comparison of the proposed vignetting effect estimation and removal methods is presented in *Section 5.1.1*.

### 2.3.2 IMU calibration

The calibration of a MEMS IMU sensor is a demanding task and its parametric values might change over time, but nevertheless it is essential.

In the case of the used sensor, its calibration had already been performed and is fully described in [26], by using a temperature dependent calibration model to estimate the gyroscopes biases for a range of temperatures (and derived a linear model for such range).

## 2.4 Installation

Figure 7 shows the previously described system setup mounted on the ship. *Appendix A* offers additional figures and a more detailed installation of the cameras, IMU and GPS devices with respect to the ship.

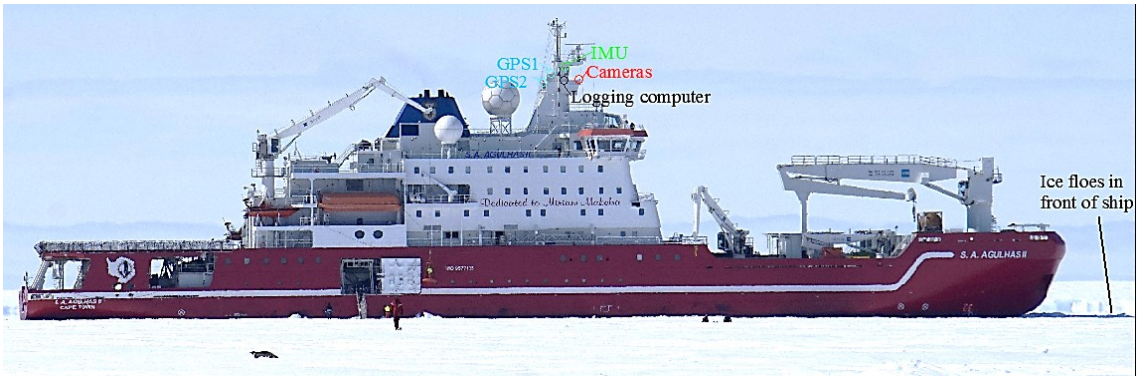


Figure 7. System installation on board of S.A. Agulhas II.

Because of size limitations during transportation, the mounting structure for the stereo cameras was sent in parts. On site measurements and planning derived in arguably the best possible design, given the initial idea. Originally, a screw and nut system was thought as to make the system modular, however after further considerations and taking into account the amount of vibrations the metallic structure would be exposed to, it was decided to weld it in one piece at Stellenbosch University.

The decision of having one single, welded structure however created additional complications: To protect the cameras from the Antarctic climate, a protective case had been made for them during previous voyages. The mounting inside the case must be done at room temperature, in a dry environment. With the designed setup, this was not possible, resulting in severe fogging of the images. The issue was finally overcome by allowing air to circulate inside the housing through a small gap.

The cameras were mounted at angles that allowed the horizon line to be in sight at any time, which could be used at a later stage as a baseline in determining the ship's roll and pitch.

Regarding the GPS antennas, these were installed at different locations over the crow's nest. The reasoning behind it, is that steel is nowadays used as the main material in ships construction for a number of reasons [45], which hinders the ability of a GPS antenna to function normally by, for example, obstructing the satellite signal or creating a multipath environment. In this kind of environment, presented in Figure 8, the same satellite signal may reach the GPS antenna at different time points effectively altering its position estimation.

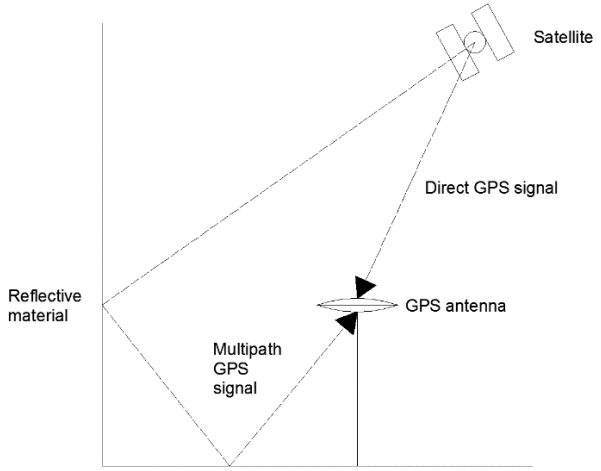


Figure 8. Example of multipath in GPS positioning.

Lastly, the IMU sensor together with the logging computer were installed inside a room in the crow's nest. Inside the room, two electric heating elements were set at 24°C. Unfortunately, no recording sensor was used to monitor the actual temperature in the room and the IMU device does not report its internal temperature, hence it will be assumed to be constant at 22°C ±4°C.

### 2.4.1 Data recording and saving

All the control and data logging was performed on a generic desktop computer, securely attached in the crow's nest. Its basic characteristics include an i7 processor, 6Gb of RAM and integrated graphics card. However, the essential characteristic for the project are a network card with two Gigabit Ethernet ports for the Basler cameras, and sufficient mounting space for a total of four hard disks, not including the system disk. Each hard disk had a capacity of 8Tb, where two of them were used as the main data logging platform, and the remaining two were used as a backup. Next, a description of each individual component and software used to record its data is presented.

As mentioned previously, the two Basler cameras were attached to the computer through two ruggedized Ethernet cables, in a master-slave configuration: the "right eye camera" was considered as master, and the "left eye camera" as slave. The provided software from the

manufacturer, Basler pylon camera software suite [46], was used during the installation process to check system integrity, however it was deemed not suitable for long periods of data logging. Instead, a slightly modified version was used of an implementation of the camera's own SDK developed at Aalto University, Autonomous Systems research group. The implementation was adapted to allow recording of raw images, taking advantage of the full 12-bit colour depth.

The image capture software would firstly set-up parameters (such as those described in *Section 2.2.1*) in both cameras. The master camera was set to capture images with 1Hz, and through a digital I/O pin, it would trigger a capture in the slave camera as well as limit its exposure time. In such approach, the obtained images present identical parameters and are tightly synchronized. In addition, the full vertical resolution of the cameras was not used, since it would not provide meaningful information (i.e. the sky) and by cropping them, disk storage can be saved.

Each pair of images were saved successively based on their frame count in an hour named folder, inside a date named folder: even images were saved in one hard disk, and odd ones to the other one. In this way, data rate in each hard disk is halved (0.5Hz), however, data loss is further avoided since if one hard disk fails, data can still be used from the other hard disk. Even more, both hard disks were manually backed up every few days, decreasing the chances of data loss to a minimum. The images naming is as follows: *image_"frame count"_"time stamp since started recording"_"ticks since started recording"_"isMaster (0 or 1)".tiff*

IMU and GPS devices were accessed and their data collected by means of two, command-line programs developed at Aalto University, Autonomous Systems research group. Similar to the images, IMU and GPS data were manually backed up to other hard disks intermittently. The data was saved as plain text, using space as column separator and new lane as data index. Table 3 presents the data structuring of the IMU in the file, while Table 4 does the same with the GPS data.

Table 3. IMU parameters in file.

| Data | Data Row Index | µs since POSIX (PC) | IMU ticks | accx | accy | accz | gyrox | gyroy | gyroz |
|---|---|---|---|---|---|---|---|---|---|
| Column # | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | | | | | | | | | |
| Data | M11 | M12 | M13 | M21 | M22 | M23 | M31 | M32 | M33 |
| Column # | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |

Table 4. GPS parameters in file.

| Data | sec. since POSIX (PC) | µs since POSIX (PC) | Latitude | Longitude | Altitude | Groundspeed | Course |
|---|---|---|---|---|---|---|---|
| Column # | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| | | | | | | | |
| Data | phop | hdop | vdop | fixmode | quality | satellites | hour (sat.) |
| Column # | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| | | | | | | | |
| Data | minute (sat.) | second (sat.) | µs since POSIX (PC) | | | | |
| Column # | 15 | 16 | 17 | | | | |

It is worth noting that in Table 4 there are two PC –computer– clocks, both representing microseconds since UNIX Epoch time, also known as POSIX time [47]. The first one, comprising the first and second columns, refers to the time when the message was received (transferred from the GPS device), while the second one specifies the time when the message was about to be written on the hard disk.

## 2.5 The coordinate system

In this section, the coordinate system used in the present thesis work is described. There are a total of five coordinate systems, which can be related to each other through homogeneous transformation matrices, as extensively detailed in [48]. These coordinate systems can be divided into device specific (cameras and image, and IMU), local (ship) and absolute (world). The first two types specify the attitude of their respective frames, and the last one is an absolute, static reference. In robotics, attitude is defined as the orientation (in 2D or 3D) of an object or frame. All coordinate systems presented here, except the world one, are assumed to be static with respect to each other, hence a static transformation matrix is sufficient to relate them.

Figure 9 shows the previously described system setup mounted on the ship, with the superimposed coordinate system for each frame.

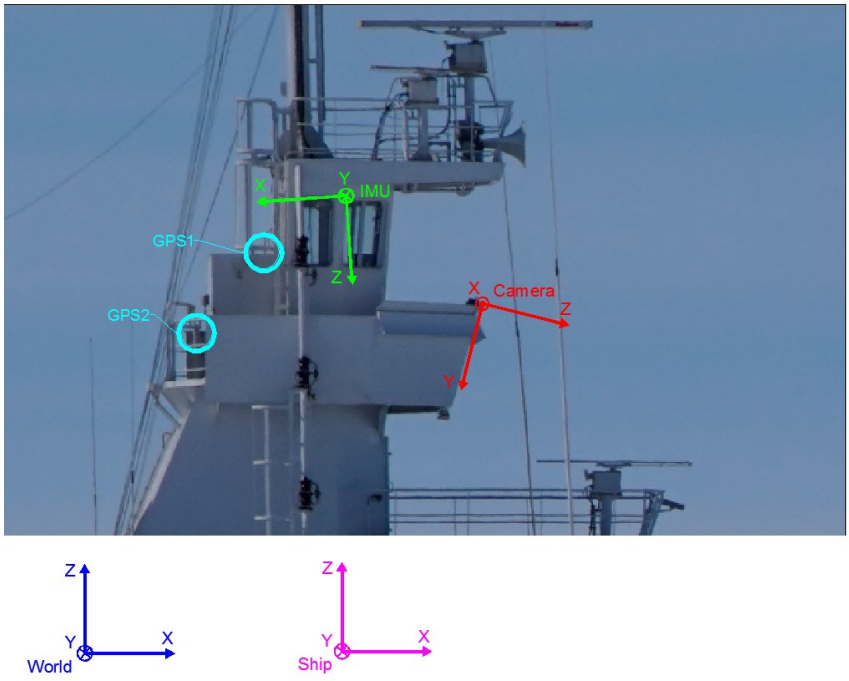


Figure 9. Coordinate frames from IMU, cameras, ship and world.

### 2.5.1 Camera coordinate system

The extrinsic camera coordinate frame is given by its centre of projection, as previously shown in Figure 4. Applied to the mounted cameras, their coordinate frames can be seen in Figure 10 below.


Figure 10. Coordinate frames of the mounted cameras.

### 2.5.2 IMU coordinate system

The IMU coordinate system is determined by the device itself, and is given by its construction (i.e. its internal arrangement of gyroscopes, accelerometers and magnetometers). In the particular case of the device used in this work, it was not possible to determine the coordinate frame of the device directly from the manufacturer. Through empirical tests however, aforementioned frame was determined and is presented in Figure 11. The results are supported by the manufacturer's specifications for other similar devices, in particular the model 3DM-GX3® [49].

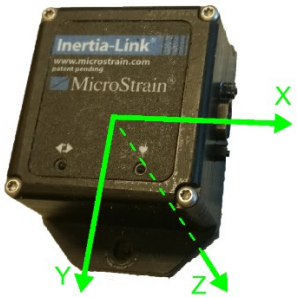


Figure 11. Inertia-Link MicroStrain® 4200-1076 (3DM-GX2).

### 2.5.3 Ship coordinate system

Along the present thesis, when referring to attitude changes of the ship rotations around the yaw axis (i.e. heading) correspond to rotation around the Z axis in the world coordinate system, pitch to Y-axis and roll to X-axis. Aforesaid notation has been previously presented in Figure 9.

The origin of the ship's coordinate system is assumed to be at the waterline and centred between its bow and aft (i.e. amidships), as well as from above. By using this assumption, the Z-axis from the ship's coordinate system approximately passes through the centre of projection of the cameras, as can be seen in Figure 12.

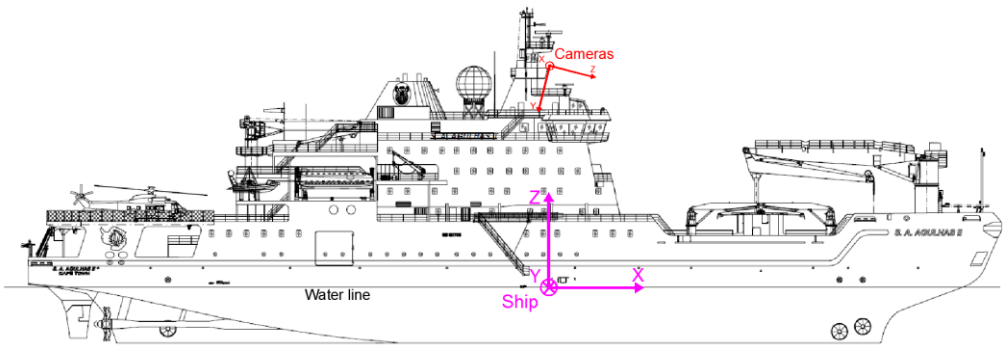


Figure 12. Ship coordinate system superimposed on the technical drawing of S.A. Agulhas II.

### 2.5.4 World coordinates

Previous coordinates systems are local and in this case define dynamic frames, which move and rotate in relation to an absolute frame called world frame. In the present work, a geodetic system is assumed as the world frame, in particular the World Geodetic System [50].

# 3. Method 1: Ice field analysis from camera images

In this section various machine vision techniques applied throughout the process of floes detection and ice field analysis from camera images are presented and described. Firstly, with a brief introduction to machine vision, followed by image pre-processing operations, ice and slush detection methods, image post-processing and then floes detection and analysis. Lastly, by means of two different algorithms the shape of the detected floes is aimed to be improved. Figure 13 presents a flow diagram of the aforementioned methods, in the order they should be applied.

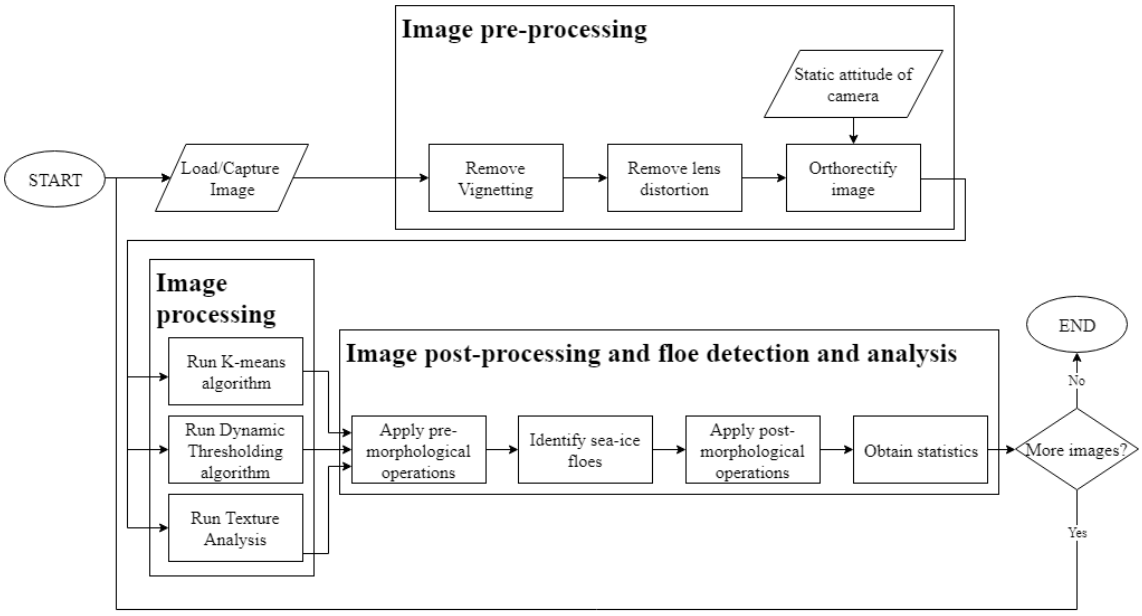


Figure 13. Flow diagram of the proposed steps for ice field analysis using camera images.

## 3.1 Background on machine vision

It is, perhaps, worthwhile clarifying the differences between image processing, computer vision, robotic vision and machine vision. Figure 14 offers a clear, overall view of their relation, as described in [51] and [52]. Firstly, image processing, which is a subsection of signal analysis, consists in transforming an image in a number of ways with the aim of improving its quality, highlight certain properties (e.g. by increasing contrast or combining colour channels) or prepare it for further processing. Secondly, computer vision is used to extract useful information from an image, mimicking the human eye (e.g. object detection, pose estimation, optical flow, etc.). Thirdly, in robotic vision an additional dimension is added by means of machine learning, together with techniques from the area of robotics, such as transformation frames or robot control. Combined, they may be used for pattern recognition, robot navigation or object grasping. Lastly,

machine vision commonly refers to the practical application of image processing, and may include certain techniques from robotics, such as transformation frames.

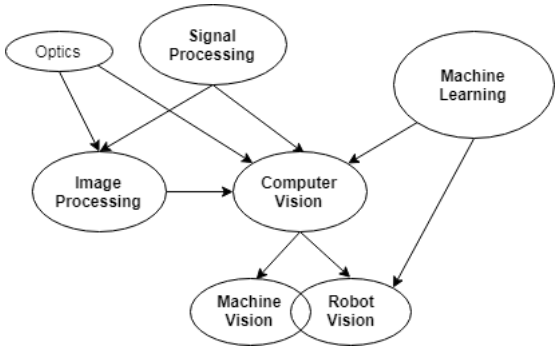


Figure 14. Relation between image processing, computer vision, robotic vision and machine vision.

Nowadays, machine vision has grown to a by no means negligible size: it agglomerates an extensive body of algorithms. New developments in machine vision have been driven by new and/or more efficient ways to process data (i.e. in a wide extend of the word, images).

## 3.2 Image pre-processing

Before ice floes can be detected and measured, images need to undergo a series of operations which would remove camera artefacts and rectify them such that they would display the "true horizontal shape" of the floes (i.e. as seen perpendicularly from above). First, the intensity related artefacts are removed, specifically the vignette effect–dividing the image by a de-vignetting estimated mask–and in addition, a black mask to cover the ship is added (such that it would not influence the detection algorithms). Next, the lens distortions are removed by means of the intrinsic camera parameters obtained from a calibration data set, as detailed earlier in *Section 2.3.1.2*. Once the artefacts are removed, images are geometrically transformed by means of a projective 2-D transform, described next.

### 3.2.3 Geometric transformations

Since images need to be taken at an angle with respect to the normal axis of the horizontal plane (sea water in the present case), the weak perspective projection of the camera would distort the elements in the images (e.g. an ice floe would increasingly shrink the further away is from the camera). Geometric transformations can recover the true shape of the captured environment, considering certain limitations.

Two methods for the geometric rectification of images are proposed, and their results later on compared and discussed. The first method is based purely on a geometric orthorectification around the pitch angle, derived in the present thesis work and based on the geometric representation presented in Figure 15 (weak perspective of pinhole camera) from the work proposed in [6]. Using it as a starting point, a set of equations are derived for geometric orthorectification. The core distinction from the equations proposed in [6] is that they use a forward transform approach, that is, for each pixel in the original image its coordinates in the orthorectified plane are calculated. The derived approach in the present work is based on an inverse transform: for each pixel of the orthorectified plane, its value is calculated from the original image. Since the calculated coordinates in the original image will almost surely lie between pixels, an interpolation method is required to obtain the pixel value. The second method proposed follows a more standardized approach based on the camera coordinate system, transformation matrixes from robotics and the concept of homography presented in [53] and [38]. Homography, applied to computer vision, states that two projections of the same planar surface by a pinhole camera model can be directly related though a homography matrix.

*3.2.3.1 Image rectification through geometric relations*

In this method, the original image contained within a virtual plane is first geometrically transformed to another plane called orthorectification plane, where the camera height is determined by the focal length *f*. Then, a scaling factor *s* (pixels per world unit in millimetres) is obtained from the ratio between the new virtual camera height and its world height, which is applied to the aforementioned orthorectification plane in order to scale it to world coordinates.

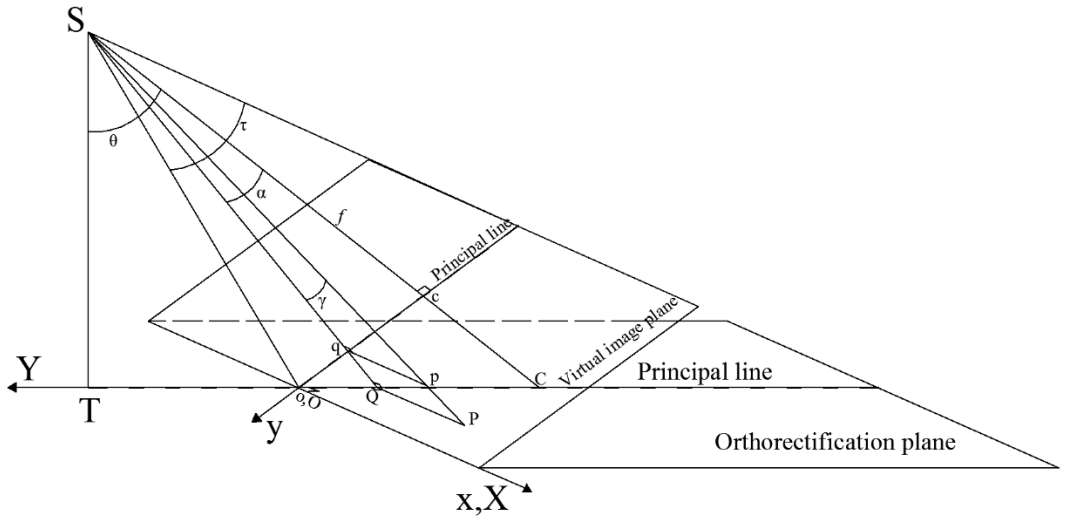


Figure 15. Geometric orthorectification representation.

In Figure 15, the principal line passes through *c*, which is the optical centre of the original image and is perpendicular to the camera's principal axis.

The optical field of view (FOV) angle $\tau$ of a camera can be determined by:

$$\tau = 2 \cdot \tan^{-1}\left(\frac{v}{2f}\right) \tag{6}$$

where *v* is the (horizontal) physical size of the sensor and *f* is the focal length. Both dimensions must be in the same unit, generally millimetres.

From Figure 15, together with (6) it can be derived that $|\overrightarrow{cq}|$, the distance increment from the optical centre *c* of the original image in the *y* direction and the orthogonal projection *q* of a point *p* on the principal line in the virtual image plane, is:

$$|\overrightarrow{cq}| = f \cdot \tan(\alpha) \tag{7}$$

where *f* is the focal length (in millimetres) and $\alpha$ is the angular increment from the principal axis in the *y* direction defined in (9), and:

$$|\overrightarrow{qp}| = \frac{f}{\cos(\alpha)} \cdot \frac{\Delta X \cdot \cos(\theta - \alpha)}{|\overrightarrow{ST}|} \tag{8}$$

where $|\overrightarrow{qp}|$ is the distance increment between a point *p* and its orthogonal representation *q* on the principal line , $\Delta X$ is the increment from the optical centre of the orthorectified image in the *X* direction, $\theta$ is the angular opening between the normal axis of the orthorectified plane and the principal axis of the camera, and lastly $|\overrightarrow{ST}|$ is defined in (10).

$$\alpha = \theta - \tan^{-1}\left(\frac{|\overrightarrow{TO}| + \Delta Y}{|\overrightarrow{ST}|}\right) \tag{9}$$

In (9), $\Delta Y$ is the increment from the optical centre of the orthorectified image in the *Y* direction.

$$|\overrightarrow{ST}| = \frac{f}{\cos(\tau)} \cdot \cos\left(\theta - \frac{\tau}{2}\right) \tag{10}$$

$$|\overrightarrow{TO}| = \frac{f}{\cos(\tau)} \cdot \sin\left(\theta - \frac{\tau}{2}\right) \tag{11}$$

The aforementioned distance increments $|\overrightarrow{cq}|$ and $|\overrightarrow{qp}|$ can be related to the standard camera image coordinates through:

$$p_y = c_y + \frac{|\overrightarrow{cq}|}{\mu}$$
$$p_x = c_x + \frac{|\overrightarrow{qp}|}{\mu} \tag{12}$$

where $(p_y,p_x)$ are pixel coordinates of a point $p$ in the image, $(c_y,c_x)$ is the pixel coordinates of the image's optical centre and $\mu$ is a scaling factor given by the physical size of one pixel, which is usually square and given as *mm/pixel*.

The calculus and memory required for small distance increments $\Delta Y$ and $\Delta X$ in the orthorectified plane greatly increases as the angle between the camera's principal axis moves away from the normal vector of the orthorectification plane. In order to improve efficiency, a scale value, *S*, is introduced. It is based on the idea that, since the camera height and inclination are large, the rectified plane will be large as well, hence it is safe to assume that a great number of near pixels from the rectified plane will request a value from the same pixel in the original image (assuming no interpolation, only nearest method). With this scaling factor, the pixel size in the rectified plane is effectively increased by a factor of S. Now, the distance increments $\Delta Y$ and $\Delta X$ can be calculated from pixel coordinates of the orthorectified image as follows:

$$\Delta Y = |\overrightarrow{CQ}| = \frac{P_y - C_y}{\mu} \cdot S$$
$$\Delta X = |\overrightarrow{QP}| = \frac{P_x - C_x}{\mu} \cdot S \tag{13}$$

where $(P_y, P_x)$ are the pixel coordinates of a point *P* in the orthorectified plane, and $(C_y, C_x)$ are the coordinates of the optical centre of the orthorectified image.

Lastly, using similar triangles, an absolute scaling factor can be obtained, which directly relates measured distances in the orthorectified image (pixel wise), with real world measurements:

$$s = \left(\frac{h}{ST}\right) \cdot \mu \cdot S \tag{14}$$

where *s* is the scaling factor from pixels to real world dimensions in *mm/pixels*, *h* is the height of the camera in millimetres from the horizontal world plane, $\mu$ is side length of one pixel and *S* is a positive integer scaling factor.

The interpolation method proposed is based on the distance from a calculated point $q$ on the original image, to its four surrounding pixels' locations $q_i$:

$$Irect_{q(x,y)} = \sum_{i=1}^{4} I_{q_{i(x,y)}} \frac{\sqrt{\left(q_{i(x,y)} - q_{(x,y)}\right)^2}}{\sum_{i=1}^{4}\sqrt{\left(q_{i(x,y)} - q_{(x,y)}\right)^2}} \tag{15}$$

where $Irect_{q(x,y)}$ is the pixel value of point $q$ in the orthorectified image and $I_{qi(x,y)}$ is the pixel value of one pixel in the original image surrounding point $q$.

*3.2.3.2 Image rectification through homogeneous transform*

An example relation between a planar surface in the environment, captured image and rectified image is presented in Figure 16.

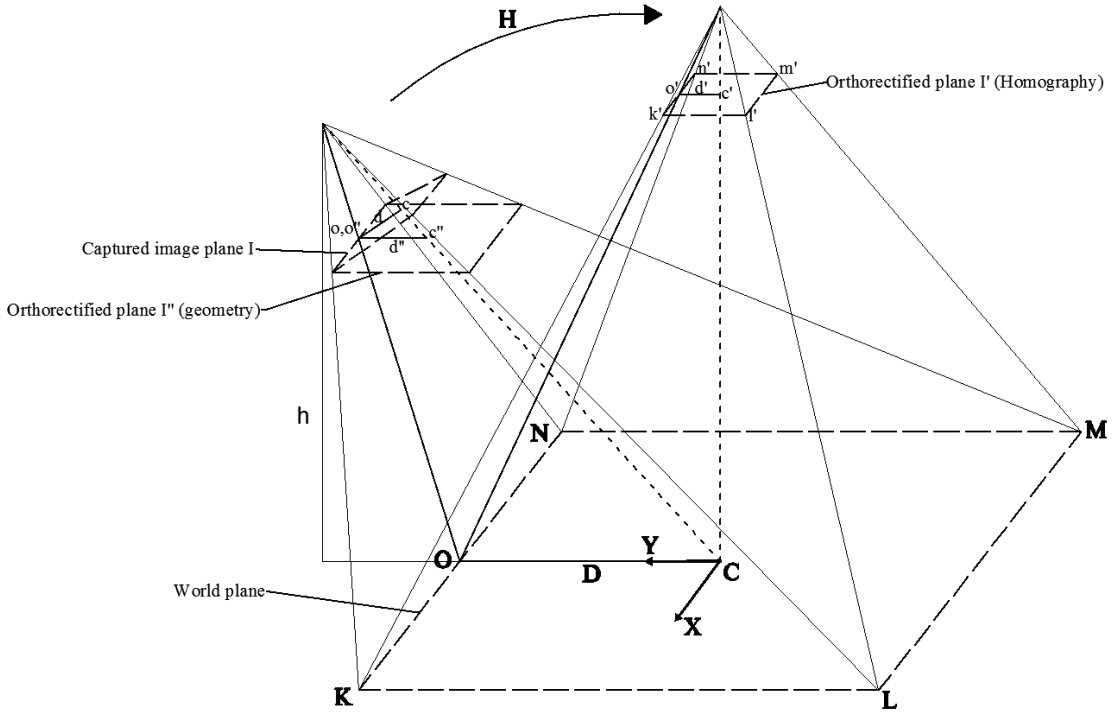


Figure 16. Example relation through homography between captured image *I*, planar surface (environment) and rectified image.

The relation between one projective image *I* of a planar surface (e.g. captured image), and another projective image of the same surface *I'* ("rectified" image) is given by:

$$I' \cong HI \tag{16}$$

where *H* is the homography matrix. Such matrix provides the required projective transformation up to a scaling factor, because homogeneous coordinates are used. In the ideal case of a complete planar surface, such relation becomes strongly strict (i.e. accurate). However, if the surface is not completely planar, 3D objects are altered to different degrees, depending on their height, due to the fact that new information cannot be created (i.e. the camera cannot see behind an obstacle, hence that data is occluded and lost when image is geometrically rectified). In the present work, it is assumed that ice floes floating in sea water form an approximate planar surface.

There are several methods for obtaining the homography matrix, which include for example RANSAC estimators and minimization and require a pair of images [54][55] or point correspondence between two planes [56]. However, as derived in [53], the Euclidean homography $H_E$ and projective 2D homography *H* are related through the camera parameter matrix *K* as:

$$H_E \cong K^{-1}HK \tag{17}$$

Furthermore, the Euclidean homography is given by:

$$H_E \cong R + \frac{t}{d}n^T \tag{18}$$

where *R* is a 3x3 rotation matrix between two camera perspectives and the term $\frac{t}{d}n^T$ represents the translation between same perspectives.

Combining (17) and (18), and removing the translation term, the homography equation can be derived:

$$H = KRK^{-1} \tag{19}$$

Hence, if the camera parameter matrix–containing its intrinsic parameters–and the rotation performed between the two perspectives are known, the 2D perspective homography matrix can directly be obtained.

The camera parameter matrix *K* is defined in (3), *Section 2.3.1.2.*

As for the rotation matrix, in this section only a static rotation around the world Y-axis (pitch) is considered, and is given as:

$$R = R_Y(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{20}$$

where $\theta$ represents the angular rotation.

An image can thereafter be orthorectified through the Matlab functions *projective2d* (use the homography matrix *H* to create a 2-D geometric transformation encapsulated in an object) and *imwarp* (applies the specified geometric transformation to an image).

Lastly, the scaling factor for the orthorectified plane, required for directly relating pixel and world dimensions can be estimated from:

$$s' = \frac{D}{d'} = \frac{d \cdot s}{d'} \tag{21}$$

where *s'* is the scaling factor in [mm/pixels] unit and from Figure 16, $D = |\overrightarrow{CO}|$. In the case of $d = |\overrightarrow{c''o''}|$, it can be obtained by means of the previously derived geometric equation, specifically (7) and (8), and *s* is the scaling factor obtained from (14). Lastly, $d' = |\overrightarrow{c'o'}|$ and the coordinates of points *o'* and *c'* can be directly calculated with the homography matrix:

$$[o' \quad c'] = H[o \quad c] \tag{22}$$

## 3.3 Image processing: ice and slush detection

Once an image has been through the pre-processing phase, useful information can be extracted from it, for instance discern between floating ice and slush, and open water. This process is called image segmentation, and as the name indicates, the aim is to correctly portion or identify the parts of the image containing ice, those containing slush and lastly open water.

Numerous attempts have been made over the years, since almost the beginning of digital image analysis, to implement a fast and reliable image segmentation technique, for example [57] or [58]. Nowadays, the development of such techniques has been mostly focused on optimization, rather than development of new methods.

In more recent years, image segmentation has been applied as well to satellite images. For instance, in [59] the authors present several methods and combination of them used to classify or group features in an image, however in a broad sense (i.e. any kind of features which have in common their colour.

In the present work, a total of three distinct methods are proposed. The first algorithm relies on pixel values to perform an intensity analysis on the image by means of a standard algorithm. The second method is an optimized derived algorithm from the first one, which follows the intensity analysis at its core. Then, the third algorithm was derived in this work and performs a local texture analysis on the image.

### 3.3.1 Intensity analysis

One of the most common techniques for image segmentation is that of thresholding: pixels are classified or grouped together based on their value or intensity according to a threshold, which can be manually or automatically set.

Several studies of the surface albedo (i.e. measurement of the surface reflectance) of different types of ice have been carried on over the years, including for example [60], [61], [62] or [63]. The first two are of special interest, as they examine the ice zone in Antarctic waters. All in all, the main idea remains constant among the studies: the reflectance index of ice (with or without a snow cover), slush and open water creates a clear distinction among them, as it can be noted from Table B1, in Appendix B. In addition, Figure B1 from the same appendix presents another factor which further increases the contrast between classes: the albedo of the surfaces under study is dependent on the wavelength of the incident rays. During the preliminary study carried out for the present thesis, it was hypothesized that a near-UV sensitive camera would present the greatest contrast among the classes of snow, ice and open water; however, a combination of short preparation time and an already existing equipment for image capturing did not allow for further considerations in this aspect.

Using a thresholding method in sea ice analysis is not a novelty, previous works have already explored such approach. In [3], the authors propose a local dynamic threshold based on the estimation of a bi-Gaussian distribution function over grey levels. In [6] the authors continue the process of identifying sea-ice floes by applying a thresholding method, as well as one additional algorithm: K-Means.

Since the main goal of the present thesis is to automate the process of sea-ice field analysis, and in this case the sea-ice identification, the K-Means algorithm is explored and applied. However,

such algorithm is particularly computationally expensive, even with an optimized implementation. Subsequently, another method is derived, based on the K-Means algorithm and thresholding, which can potentially process an image in a duration one order of magnitude smaller.

*3.3.1.1 K-Means*

K-Means is an iterative clustering algorithm, originally from signal processing. Its goal is to group a set of *j* observations into *i* clusters, each having a mean or *centroid* (hence the name). The idea behind the algorithm dates back to 1957, by Hugo Steinhaus, yet the standard version of the algorithm was suggested by Stuart Lloyd in the same year [64].

It is beyond scope to disclose a full explanation of the algorithm, however a short introduction to the algorithm proposed by Lloyd [65] is presented first, followed by an improved update by Arthur and Vassilvitskii [66].

In Lloyd's algorithm, *k* initial centres are chosen randomly. Those are the starting points or seeds, used by the algorithm to try and find the best possible cluster means iteratively, by minimizing (23). That is, the algorithm calculates the sum of the distances between all the observations *j* belonging to a class $G_i$ and the mean $m_i$ of said class, among all classes *N*. There are two ways in which an observation is assigned to one of the classes: either assign the observation to the class with the closest centroid, or if by reassigning the observation to different classes, the total sum within the class is decreased. Lastly, an updated mean within each class is computed and used as the new centroids. Those steps are repeated either until class assignments of the observations do not change or a specified maximum number of iterations is reached. Other methods for calculating distances between observation and centroids are also possible, as described in [67].

$$\arg_{\mathrm{G}} min \sum_{i=1}^{N} \sum_{j \in G_i} \left|\left|j - m_i\right|\right|^2 = \arg_{G} min \sum_{i=1}^{N} |G_i| \, Var \, G_i \tag{23}$$

As mentioned earlier, in their revision Arthur and Vassilvitskii proved the advantages of carefully choosing the seeds for the K-Means algorithm by using a heuristic, achieving faster convergence to a smaller sum of distances within a class [66]. In their approach, an observation is chosen randomly from a uniformly distributed data set *J* and used as the first centroid $ct_1$. Then, the distance between each observation $j \in J$ to $ct_1$ is calculated and denoted as $d(J_j,ct_1)$. Additional

centroids may be randomly selected, in which case a probability distribution for the data set *J* is calculated according to:

$$P(j \in J) = \frac{d^2(J_j, ct_l)}{\sum_{i=1}^{N} d^2(J_j, ct_l)} \tag{24}$$

Up to this point, the algorithm has been initialized with *N* centroids corresponding to observations. Next, aforementioned centroids are refined iteratively by first calculating the distances between observations and each centroid, and assigning each observation to its nearest centroid.

In the current case, each pixel (and its intensity value) in an image is considered an "observation", and can be processed as such with the K-Means algorithm implementation in Matlab, by making use of the function *kmeans*. Such function includes the revision by Arthur and Vassilvitskii, which is key for not only automating the image segmentation part, but also improving its efficiency.

Once the centroids are obtained from the *kmeans* function, an image is segmented according to the nearest centroid rule (i.e. the half distance between two centroids is considered the thresholding level, marked as a dashed line in Figure 17). The classes used in the segmentation are 1 for open water, 2 for slush, 3 for ice/snow and 0 for anything other (such as the ship or frame around the rectified image):

$$q'_{x,y} = \begin{cases} 1, & 0 < q_{x,y} < T_1 \\ 2, & T_1 \leq q_{x,y} < T_2 \\ 3, & T_2 \leq q_{x,y} \leq 1 \\ 0, & q_{x,y} \leq 0 \ \cup \ q_{x,y} > 1 \end{cases} \tag{25}$$

where $q_{x,y}$ is the pixel value of the original image at coordinates *x,y*, $T_1$ and $T_2$ are thresholding values, and $q'_{x,y}$ is the pixel value in the processed image.

Figure 17 presents a run example of the K-Means algorithm on an image, compared to the histogram of the image, showing a correlation between centroids and centres of mass for ice, slush and water.

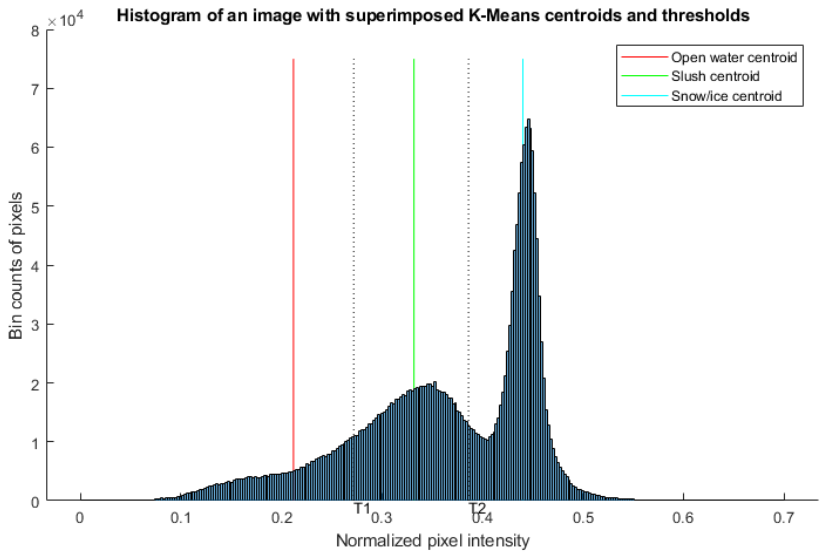


Figure 17. Histogram containing peaks for open water, slush, and ice/snow and found centroids though K-means algorithm and derived thresholding levels T1 and T2.

*3.3.1.2 Adaptive Thresholding*

As mentioned earlier, K-Means algorithm can automatically and successfully segment an image based on pixel intensity. However, such process comes at a cost: it requires high processing times. Even with Arthur and Vassilvitskii's revision, which improves speed and accuracy, it remains a "slow algorithm" [66].

Since the data captured consists of a sequence of images with small time increment between them, and following the idea behind the update of centroids innate to the standard K-Means algorithm, a similar approach was implemented in order to reduce the computation time required for one image. Under certain conditions, such approach has proved to rival in precision with the standard K-Means algorithm for each individual image, as later presented in *Chapter 5*.

The proposed algorithm runs as follows. For the first image, the "correct" centroids for each class are obtained using the function *kmeans* in Matlab. Next, these centroids are used as previously explained to threshold the following image in the sequence. Once the classes have been determined in an image, a mean value for each class is calculated using (26), which become the new or updated centroids:

$$ct^i = \frac{\sum_{l=1}^{N^i} q_l^i}{N^i} \tag{26}$$

where $ct^i$ represents a centroid belonging to class $i$, $N^i$ is the total number of pixels in said class and $q_l^i$ is the value of a pixel which belongs to class $i$ and disregarding its position in the image.

The proposed modification of the algorithm requires that an extensive enough sample of pixels for each class is present (i.e. there are a certain number of pixels classified in one class, such that they can effectively be used to calculate a meaningful new centroid for the same class). Such number is pre-defined manually as a threshold, and can be individual for each class based, for example, on an *a priori* information of the overall expected concentration of pixels for each class.

To overcome situations where a sample large enough is not available for a number of classes, one possible solution would be to update centroids of those classes based on the update performed on a contiguous centroid. This is, if the number of pixels in a class does not overcome a certain threshold, the centroid for such class can be updated by the same amount as the nearest centroid of another class. Equation (27) describes such relation:

$$ct_{k+1}^i = \begin{cases} \frac{\sum_{l=1}^{N_k^i} q_{k,l}^i}{N_k^i}, & N_k^i > T \\ ct_k^i + \Delta ct_{k+1}^{i\pm1}, & N_k^i < T \end{cases} \tag{27}$$

where $N_k^i$ is the number of pixels in class $i$ at time instant $k$, $q_{k,l}^i$ is the value of one pixel $l$ belonging to class $i$ and at time instant $k$, $T$ is an integer specifying the thresholding for updating the centroid based on the pixel values, $ct_k^i$ is the current centroid value of class $i$, $ct_{k+1}^i$ is the updated centroid at time instant $k+1$ and $\Delta ct_k^{i\pm1}$ is the value increment at time instant $k+1$ for one of the contiguous centroids, chosen randomly or based on availability.

### 3.3.2 Texture analysis

Even though intensity analysis techniques can provide highly accurate results, they can also fail in disastrous manners. Certain ice conditions, with high local intensity variation and ice concentration, such as those of brash ice, can trick the intensity analysis algorithms into over-segmentation and under-detection of sea ice floes. In such situations, it was noted that, while the slush surrounding the ice floes mixes with sea water and presents an irregular texture in the image, sea ice floes usually present a smooth surface texture.

Texture analysis methods in computer vision consider complex visual patterns in an image in order to extract meaningful information from it. It has numerous and distinct applications, from detecting young spruce trees in the forest [68], to differentiating between fresh and frozen-thawed

fish fillets [69]. According to [70], texture analysis methods can be classified in four distinct categories: structural (based on pre-defined primitive textures and their spatial arrangement), statistical (represent the textures by the non-deterministic relationships between pixels), model-based (analyse structures by means of fractal and stochastic models) and transform methods (reinterpret an image in a distinct space, such as Fourier or wavelet transform, where texture properties can easily be extracted).

Based on the problem statement and its characteristics, the author proposed the following approach: by statistically analysing the local variation in texture, it is possible to distinguish ice floes in an image. This analysis can be done by measuring the local Shannon entropy [71] in an image. Entropy, applied to a digital image, is a statistical measure of randomness present in it. By measuring this randomness or entropy on a local scale (i.e. neighbourhood), it is possible to determine how "smooth" a texture is in the specified neighbourhood.

To perform aforementioned algorithm, the function *entropyfilt* present in Matlab is used [72]. Such function returns an image of the same size as the input image, where each pixel value corresponds to the entropy measured around a neighbourhood of specified size and shape. The entropy value of a pixel in the function is defined as:

$$e = -\sum p \cdot log_2 (p) \tag{28}$$

where $p$ contains the local normalized histogram counts inside the specified neighbourhood.

Lastly, in order to automate the classification of the entropy image, once again the K-means algorithm can be used with only two centroids, effectively dividing the image in two classes: smooth (corresponding to ice floes) and not smooth (corresponding to slush and/or water).

## 3.4 Image post-processing: morphological operations

Once open water, slush and ice/snow are discerned, a new challenge arises: correctly identifying individual ice floes and their characteristics, such as dimensions, area and distribution. One heuristic technique, which can be used to ease the task of extracting such information from an image, involves morphological operations (i.e. process images based on shapes).

Image processing using morphological operations is based on mathematical morphology, which is a collection of non-linear spatial operators. Each pixel in the output image corresponds to a function applied to a subset of pixels around the same pixel in the original image; relation shown

in (29) from Corke's book [53] in page 316. Since the language of mathematical morphology is set theory, it can be applied directly to binary images [73].

$$I'_{x,y} = g\left(I_{x+i,y+l}\right), \qquad \forall (i,l) \in \delta, \qquad \forall (x,y) \in I \tag{29}$$

In (29), *x* and *y* represent pixel coordinates of *I'* (output image) and *I* (input image). The function applied to the subset of pixels is in fact a structuring element, $\delta$. It can be thought of as a kernel, as it works in a similar fashion, except that the function $g(\cdot)$ is applied only to a subset of pixels inside the window specified by the structuring element (hence such term is more appropriate for convolution operations). It is important to note, that the specified centre of a structuring element does not need to be the same as the geometric centre of that shape contained within.

In binary morphology, which is a certain case of lattice morphology, an image is treated as a binary matrix over which is applied a structuring element (also a binary image). The basic binary morphological operators, which include erosion, dilation, opening and closing, are shift-invariant (i.e. translation invariant).

Assuming a binary image with objects represented by ones, in common words an erosion operation would remove pixels (set them to zero) on the object's boundaries, whether those are external or internal (i.e. a single zero pixel inside an object will increase its size according to the structuring element used). Hence fore, an erosion operation slides the structuring element across an image and sets the pixel under consideration to zero each time another pixel in its neighbourhood is zero. In the case of grayscale images, a pixel will be set to the minimum value of its neighbourhood defined by the structuring element. Both approaches are exemplified in Figure 18.

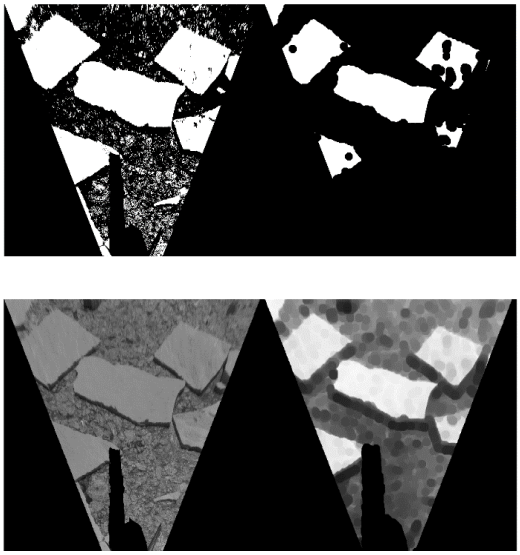

Figure 18. Morphological erosion of a binary image (top) and grayscale image (bottom) using *'disk'* (a circular shaped) structuring element.

Dilation, on the contrary, is an operation by which the object boundaries (once again, internal and external) increase depending on the size and shape of the structuring element used. In other words, a dilation operation slides the structuring element across an image and sets the pixel of interest to one each and every time another pixel's value is one in its neighbourhood. In the case of grayscale images, a pixel is set to the maximum value of its neighbourhood defined by the structuring element. Both approaches are exemplified in Figure 19.

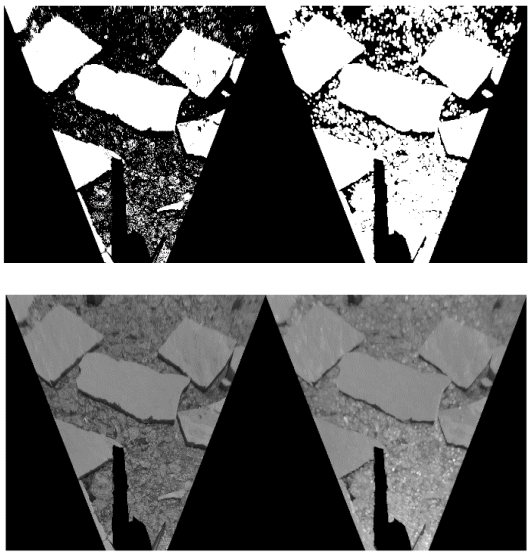

Figure 19. Morphological dilation of a binary image (top) and grayscale image (bottom) using *'disk'* (a circular shaped) structuring element.

The above two basic operations can be combined together to obtain simple, yet powerful results. One such composite operator is opening, which consists in an erosion operation, followed by a

dilation one. As the name indicates, an opening operation is well suitable where removing spikes or separating objects is desired.

Another composite operator is closing, an antagonistic approach to opening: a dilation operation is run on the image at first, followed by an erosion one. As the name indicates, a closing operation is widely appropriate for closing gaps inside objects or its boundaries. However, this operation can create additional challenges in object detection, as neighbouring objects may be fused together.

In the present study the main goal is to identify individual sea ice floes. The first step required before running any morphological operations is to threshold the image into a binary one, where every identified pixel as belonging to the class of ice would be set to "1", and any other pixel to "0".

Next, depending on the image processing algorithm applied in discerning ice/snow and open water from *Section 3.3*, two different approaches are followed. The main difference between the two lies in the type of ice condition images they target.

On one hand, K-Means and Adaptive Thresholding algorithms perform well in ice conditions similar to pack ice, where ice floes are clearly visible and their boundaries defined. In such conditions, morphological operations are used to split adjacent ice floes, separated in some cases by only a thin line of pixels; and one skilful approach to do so is by using a modified version of morphological opening. The main idea behind this method is that as long as there is even a thin, non-continuous line separating two ice floes, it will be expanded though erosion, effectively splitting an ice floe. Then, instead of continuing with the opening operation and directly applying dilation to the whole image, floes are detected as per explained in the following *Chapter 3.5*. Once detected, each floe is dilated (i.e. expanded) one by one, by the same amount as it was eroded and marked with a unique identifier. In this fashion, even if two floes do not present any separation line between them after the dilation operation, they are still identified as distinct.

On the other hand, in the case of the Texture Analysis algorithm the post-processing approach varies slightly. Texture Analysis algorithm performs well and is aimed at brash ice conditions, where a large intensity variation is present along an image's texture. Recalling the operating principle of the algorithm, it tries to detect such areas with a smooth local texture which, in theory, should correspond to an ice floe. However, an ice floe's surface may present as well arbitrary random changes in texture, which the algorithm may erroneously identify as non-belonging to the ice floe. As such, the output from the algorithm can present ice floes with gaps or, worse, fragmented. To overcome this issue, the image needs to first undergo a morphological closing

operation with a small structuring element, which should fill in gaps and join closely identified objects. Next, the same procedure as with K-Means and Adaptive Thresholding is followed.

One of the largest drawbacks of morphological operations applied in this context is the trade-off between the need of correctly splitting floes where needed and maintaining their true shape: a large structuring element will increase the chances of separating two adjacent floes, at the cost of losing their true shape; however, a small structuring element may not correctly split floes, but will maintain their true shape to a high degree.

## 3.5 Floe detection and analysis

After an image has been through all the processing steps above described, individual ice floes should become apparent. In order to identify these objects and their characteristics, two readily implemented MATLAB functions from the *Image Processing Toolbox* [74] were used. As a result, only a brief description of such functions is provided below.

The first function required takes care of individually labelling in an image each object detected, described as a series of connected pixels:

$$L = bwlabel(BW) \tag{30}$$

where $L$ is a return image with each pixel labelled with an index depending on the object boundaries it belongs and $BW$ is a binary input image to be analysed.

As mentioned earlier, once each floe has been individually labelled, they are dilated one by one, by the same amount as they were eroded using morphological operations. Then, a second function is used to extract useful information, namely:

$$stats = regionprops(L, properties) \tag{31}$$

where *stats* is a *struct array* containing a *struct* data structure for each labelled object in the image $L$, where the requested *properties* are included. The required properties for the current thesis work are “Centroid” (specifies the centre of mass of the object in pixel coordinates), “Area” (pixel count belonging to the object), “MajorAxisLength” and “MinorAxisLength” (scalar length in pixels of the major and minor axis respectively belonging to an ellipse which has the same normalized second central moments as the object).

Lastly, by means of the scales previously described in (14) and (21), the above obtained properties can be directly related to world dimensions as:

$$D_w = s \cdot d_I$$
$$A_w = s^2 \cdot a_I \tag{32}$$

where $D_w$ is the length in world units, $d_I$ is the length in pixel units in image $I$, $s$ is the scaling factor specified as *[world units/pixel]*, $A_w$ is the area in world units and $a_I$ is the area specified as pixel count in image $I$.

# 4. Method 2: Augmenting floe detection with inertial measurements

Sensor fusion between vision and inertial data has been comprehensively studied in the past, and numerous algorithms and approaches developed, see for example [75], [76], [77], [78], [79]. A tutorial introduction in [34] presents the fundamentals of visual and inertial perception from the biological and robotic systems point of view. In addition, their complementarity is discussed and few fundamental fusing approaches presented.

According to [79], there are two issues that need to be addressed when fusing vision and inertial measurements, namely: 1) Define all the measurable physical quantities; and 2) Define an efficient, yet reliable method to extract these physical quantities from the raw sensor data.

Based on the above and assuming an unknown and dynamic environment, two key issues arise for the estimation of visual attitude and/or odometry: there are no static landmarks available (the ship and cameras are considered one rigid body, and everything else dynamic) and the complexity of the dynamics of the environment makes it prohibitive to even attempt to estimate such dynamics. Moreover, weather conditions such as rain or fog further aggravates the possibility of obtaining a reliable source of attitude estimation from camera images. Consequently, a reliable method for attitude estimation is needed, based solely on raw measurements provided by an IMU device. Fusion between vision and inertial measurements can be classified into Correction, Colligation, and Fusion, where in the first category information extracted from one sensor is used to correct or verify another, in the second category different parts of the sensors' data are merged, and lastly in the third category information containing same physical quantities is fused [78].

The methods described in the present thesis fall under the aforementioned category of *Correction*, rather than a strictly *Fusion* approach, where estimated pitch and roll from an IMU sensor is utilized in image geometric rectification. An augmented version of the previously described workflow in *Method 1* is presented in Figure 20, which highlights the addition of an IMU sensor.

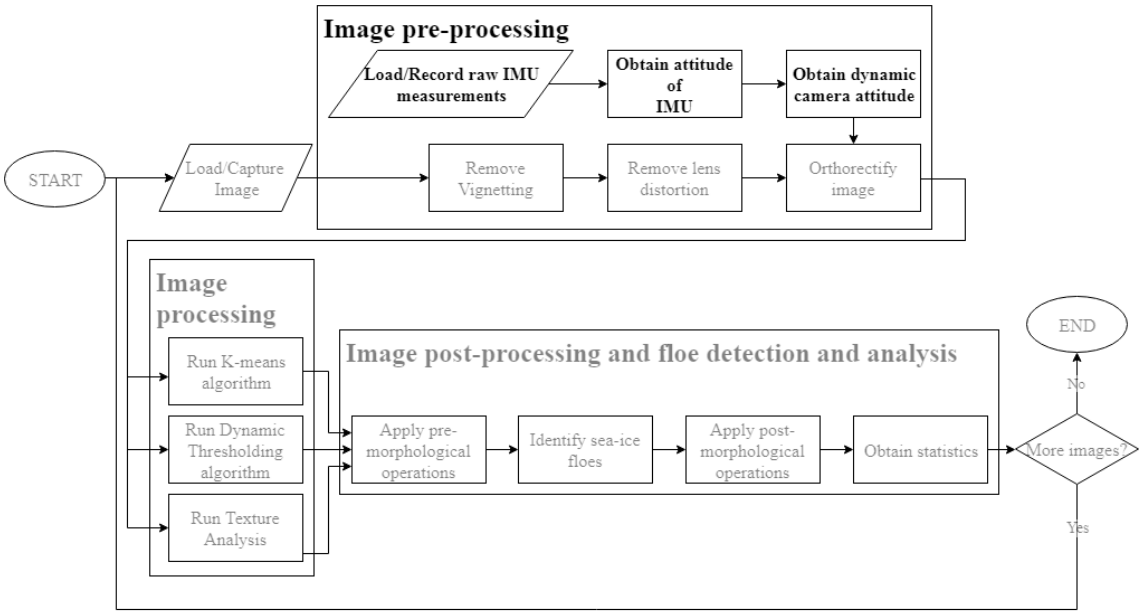


Figure 20. Flow diagram of the proposed steps for ice field analysis using camera images, with the inertial measurements highlighted.

## 4.1 Background on attitude estimation

### 4.1.1 Attitude from internal estimated rotation matrix

The IMU utilized to determine the attitude of the cameras is able to compute and output a 3x3 estimated rotation matrix, with values fully compensated for temperature [29].

The aforementioned rotation matrix, presented in (33) [29], describes the orientation of the IMU in regard to the fixed earth coordinate system previously described in *Section 2.5*

$$M = \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,3} \\ M_{2,1} & M_{2,2} & M_{2,3} \\ M_{3,1} & M_{3,2} & M_{3,3} \end{bmatrix} \tag{33}$$

The rotation matrix M satisfies (34), meaning that through dot matrix multiplication, a vector expressed in fixed earth coordinate system ($V_{FE}$) can be transformed to the same vector but expressed in the sensor's own local coordinate system ($V_{IL}$).

$$V_{IL} = M \cdot V_{FE} \tag{34}$$

Furthermore, Euler angles for pitch, roll and yaw can be extracted from the rotation matrix M using (35) [26], which follow the ZYX formulation of Euler angles and makes use of the two argument inverse tangent *atan2*, used to distinguish angles in all four quadrants.

$$Pitch = \theta = \arcsin(-M_{31})$$

$$Roll = \phi = \operatorname{atan2}(M_{32}, M_{33}) \tag{35}$$

$$Yaw = \psi = \operatorname{atan2}(M_{21}, M_{11})$$

From (35) it is clear that with a small amount of effort, it would be possible to obtain the attitude of the cameras with respect to the Fixed-Earth coordinate system as Euler angles (given, that the extrinsic correlation between the cameras and IMU is known).

### 4.1.2 DCM based attitude estimation algorithm

The exact same IMU device had previously been used (and fully characterized) by Hyyti and Visala while developing a novel attitude estimation algorithm for MEMS [26]. Their research yielded promising results, hence both the device and their algorithm were made use of in the present thesis. A thorough description of the algorithm is available in their paper, therefore the reader is presented only with a few thick brushes about the algorithm in this thesis, which serve an introductory and fitness purpose.

The research in [26] is based on the direction estimation of the gravity vector, and its relation to measured accelerations. In order to estimate aforementioned vector, angular velocities are integrated using a partial Direction Cosine Matrix (DCM), presented in (36). Unlike other DCM-type filters, their approach updates only the bottom row of the matrix, hence only pitch and roll angles can be directly derived. This is in accordance with the requirements specified for the current thesis, since yaw angle (heading) is not required for image rectification. If needed, absolute heading may be obtained with high precision from the on board instrumentation of the ship, or derived from GPS data.

$${}^{n}_{b}C = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} = \begin{bmatrix} \theta_c\psi_c & -\phi_c\psi_s + \phi_s\theta_s\psi_c & \phi_s\psi_s + \phi_c\theta_s\psi_c \\ \theta_c\psi_s & \phi_c\psi_c + \phi_s\theta_s\psi_s & -\phi_s\psi_c + \phi_c\theta_s\psi_s \\ -\theta_s & \phi_s\theta_c & \phi_c\theta_c \end{bmatrix} \tag{36}$$

In (36), *C* indicates the direction cosine matrix (rotation matrix) from the fixed body frame *b* to the navigation frame *n*. Inside the matrix, an *s* subscript refers to sine and *c* to cosine; $\theta$ refers to pitch, $\phi$ to roll and $\psi$ to yaw in Euler angles. Since the rotation matrix *C* is equivalent to the

previously presented rotation matrix *M* in (33), same equations from (35) may be used to extract roll, pitch and yaw Euler angles from *C* matrix.

The last row of the direct cosine matrix and the gyroscope bias (one for each axis) are interpreted as states and updated by means of an Extended Kalman Filter (EKF). In addition, the standard version of an EKF has been improved in a number of ways in their work, out of which the following are relevant for the present thesis: computational feasibility, since the aim of the thesis was to develop a real-time approach for ice field analysis; robustness against rapid accelerations, or in other words, natural vibrations present on a ship; and lastly, the formulation of the filter can tolerate changes in sampling rate or jitter, which may occur on a PC based setup.

Additionally, it is important to emphasize the changes in bias and gain terms of accelerometers and gyroscopes due to temperature fluctuations, when using low-cost MEMS IMUs. This is of special relevance while using DCM IMU, since it heavily relies on accelerometer readings to estimate gyroscope biases online. In their research, a measurement model is provided for estimating bias and gain terms based on temperature; and it was found that near room temperatures such terms can be linearized.

Lastly, Figure 3 from [26] is a comparison of the DCM IMU algorithm with other state-of-the-art algorithms, and which presents a clear fitness of the algorithm for the current thesis: on one hand, the algorithm handles well sudden accelerations, and on the other hand, drift in angle measurements is removed or, at least, greatly reduced. The same figure also suggests that the internally estimated rotation matrix from the Inertial-Link™ device presents a lack of accuracy and slight drift under controlled conditions in the attitude estimation. Therefore, such matrix has been recorded for the present work, but not utilized.

## 4.2 IMU-Image fusion

As previously described in Section *2.5*, only the rotation extrinsic parameters between the IMU and camera are considered in the present work, specifically the roll and pitch ones. Yaw rotation is deemed not useful for the present work; however, it is included for completeness. In addition, for the derived geometric orthorectification method, only the pitch Euler angles are used as detailed next. The camera pitch in world coordinates can be obtained with:

$$ {}^{W}_{CAM}\theta' = {}^{W}_{CAM}\theta + {}^{W}_{IMU}\theta - \theta_b \tag{37} $$

where ${}^{W}_{CAM}\theta'$ represents the dynamic or updated pitch angle of the camera *CAM* in world coordinates *W*, ${}^{W}_{CAM}\theta$ represents the static absolute pitch angle of the camera with respect to the world, ${}^{W}_{IMU}\theta$ is the current dynamic or updated angles of the IMU with respect to the world coordinate, and lastly, $\theta_b$ is the bias angle in static conditions of the IMU with respect to the world coordinate.

Next, in the case of the homogeneous transformation, pitch $\theta$ and roll $\phi$ angles are extracted from the estimated rotation matrix *C* from the DCM algorithm as detailed earlier. Then, by using a ZYX convention and setting the yaw angle to 0, a new rotation matrix is built containing only pitch and roll:

$$
\begin{aligned}
{}^{W}_{IMU}R &= R_Z(0)R_Y(\theta)R_X(\phi) \\
&= \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}
\end{aligned}
\tag{38}
$$

Which is multiplied with the transformation matrix between the camera and IMU frames and can be directly used through transformation frames:

$$
{}^{W}_{CAM}R = {}^{W}_{IMU}R\,{}^{IMU}_{CAM}R \tag{39}
$$

where ${}^{W}_{CAM}R$ is the rotation matrix providing the attitude of a camera in world coordinates, ${}^{W}_{IMU}R$ is the rotation matrix providing the attitude of the IMU (pitch and roll) in the world frame and lastly ${}^{IMU}_{CAM}R$ is a (static) rotation matrix relating the camera attitude of a camera to the attitude of the IMU.

The aforementioned rotation matrix is combined with the camera matrix *K* in order to obtain the homography matrix *H*, as detailed in (19). As explained earlier, an image can thereafter be orthorectified through the Matlab functions *projective2d* and *imwarp*.

# 5. Experiments and results

In this chapter, each subsection presents the reader with an experimental implementation of the previously described methods, as well as the obtained results. Later on, in the following chapter those results are examined and discussed.

## 5.1 Experiment A: Image pre-processing

### 5.1.1 Vignetting removal

In this experiment a de-vignetting mask is estimated through the various methods proposed earlier, which can be used to remove a vignetting effect of an image of the same size as the mask.

For the first algorithm of vignetting estimation, proposed by Zheng et al. [41], an implementation of their algorithms for Matlab [42], publicly available, was used on a captured image on board S.A. Agulhas II. Such image was selected which presented an overall illumination as smooth as possible in the region of interest (i.e. below the horizon line), shown in Figure 21. The resulting de-vignetting mask is presented in Figure 22.

Figure 21. Image presenting a vignette effect.

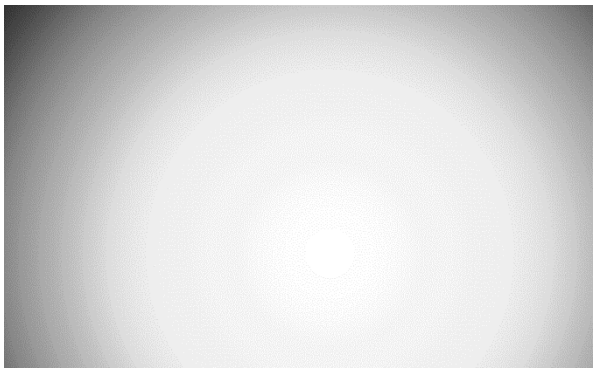

Figure 22. De-vignetting mask obtained with the method proposed by Zheng et al. in [41].

Next, the resulting masks based on the empirical methods proposed in the present work are shown in Figure 23 (averaging intensity mask) and Figure 24 (radial fit with smoothing spline).

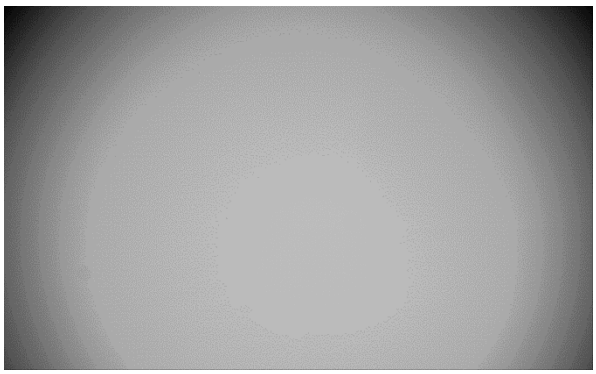

Figure 23. De-vignetting mask obtained by averaging a series of intensity images of a plain surface in controlled light conditions.

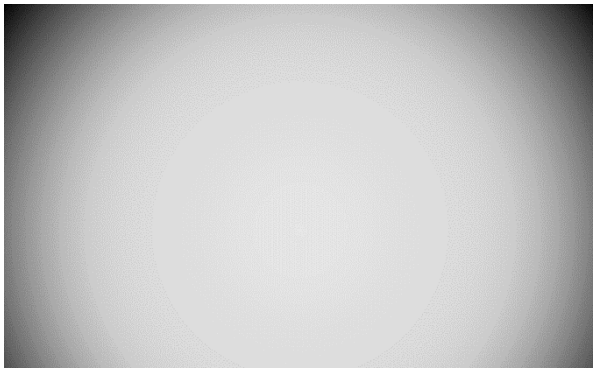

Figure 24. De-vignetting mask obtained through the fitted function method. In this case, a smoothing spline was used.

Lastly, the image in Figure 21 was pixel-wise divided by all three masks and the results of the operations are shown in Figure 25. The method and examples described in [41] seemed promising at first, however its implementation failed to fully remove the vignetting effect as can be noted from the aforementioned figure. Furthermore, a certain amount of halo is introduced (i.e., rings around the centre of the image which present higher brightness). On the other hand, the proposed and derived empirical methods described in the present thesis work have managed to successfully remove the vignetting effect on a visual scale. The mean mask however increases the overall intensity of the image, which can be undesired in machine vision applications.

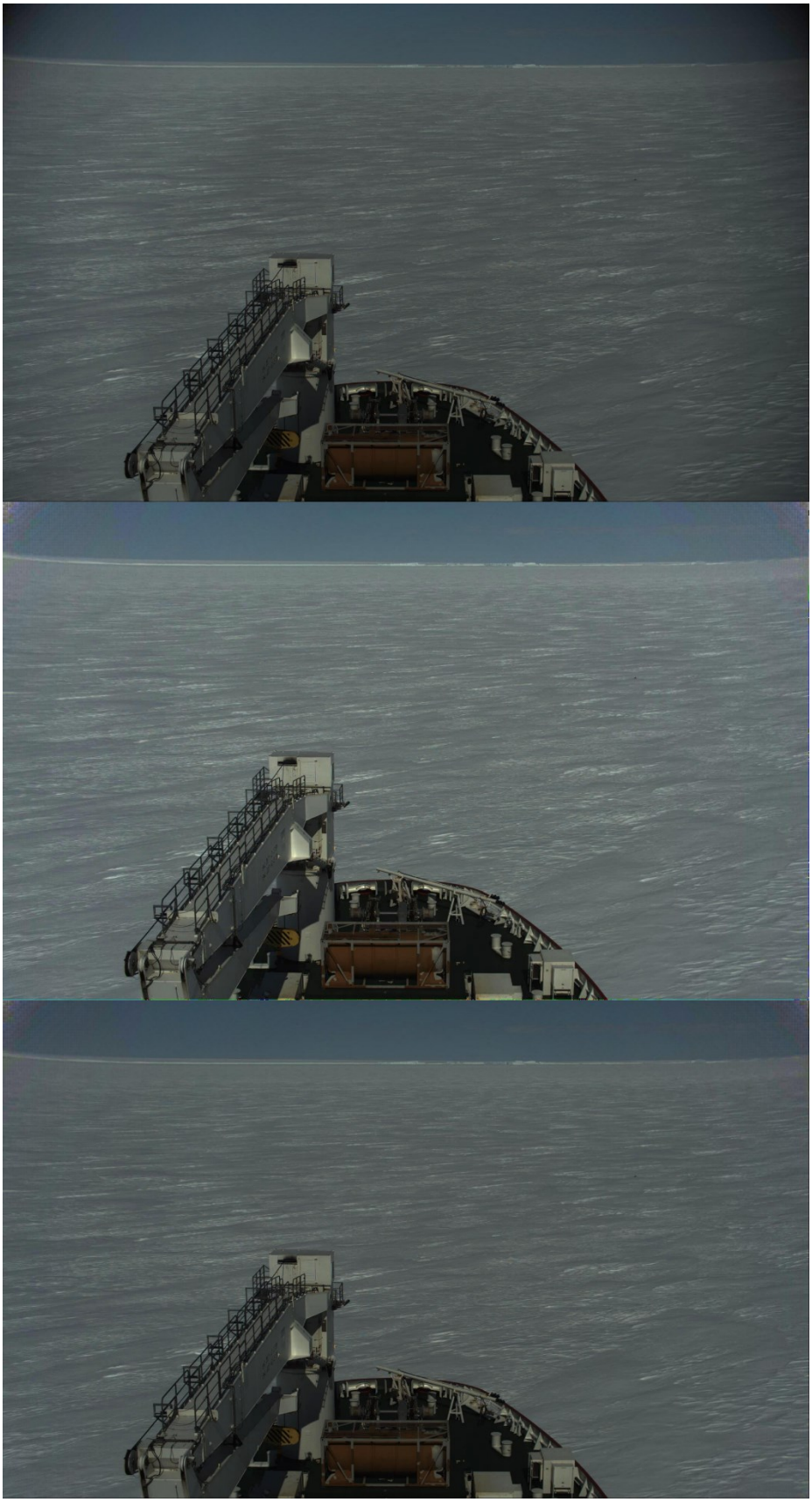

Figure 25. De-vignetting operation using three different masks from Zheng's et al. method (top), average intensity mask (middle) and fitted function mask (bottom).

### 5.1.2 Image geometric transformations: accuracy and speed

The two methods proposed for orthorectifying images were implemented in Matlab, and put to test in order to determine their accuracy in rectifying a flat surface, but also their expected error margin. At first, a miniature experimental setup was implemented where a camera was mounted on a tripod at a known height between the floor and the centre of the sensor, facing downwards at a known angle *theta* (i.e. angle between the camera's principal axis and normal axis of the floor) and capturing the image of a calibration board (checker board). The details of the setup are summarized in Table 5. Height was obtained with a measuring tape and the inclination angle from a phone inclination measuring app.

Table 5. Small scale camera setup parameters

| | |
|---|---|
| Camera model | Sony a5000 |
| Image size (HxV) | 5456x3632 |
| Calibrated optical centre (HxV) | 2712.4375x1809.6414 |
| Height of camera (wrt ground) | 140.5cm |
| Theta angle | 77.5° |
| Side of square (checker board) | 100mm |

The original image of the small scale dataset is presented in Figure 26, together with the orthorectified version of it by means of both methods in Figure 27. As can be observed, the checkerboard recovers its true shape in both orthorectified examples. A particular case occurs when orthorectifying a flat surface parallel to the orthorectification plane but does not lie on it. For instance, in Figure 26 the table on the left does recover its true shape (circular) in the orthorectified images from Figure 27, but not its scale. All the remaining objects in the image which grow above the orthorectifying plane (i.e. in this case, the floor) and are not parallel to it become elongated (e.g. chairs, drawers or shelves).

Then, in Table 6 the accuracy results are presented. First, the distance between a number of squares from the checker board were manually measured and then multiplied with the calculated scale, obtaining an average distance for the horizontal and vertical sides of a square. Second, the Matlab function *detectCheckerboardPoints* was used in order to obtain subpixel position of all the corners of the checker board through interpolation. Then, all distances between adjacent points in vertical and horizontal axis were calculated and the mean value are presented in the same table.

Figure 26. Toy image used for comparison of methods.

Figure 27. Toy dataset with known parameters orthorectified with geometric method (left) and transformation frame (right).

Table 6. Accuracy results of two orthorectifying methods using small scale data

| | Method | |
|---|---|---|
| Parameter | Geometric | Transformation frame |
| Scale [mm/px] | $2.17522 \times 10^{-4}$ | $2.97901 \times 10^{-4}$ |
| Mean meas. manual horizontal | 97.88mm (45px) | 98.31mm (33px) |
| Mean meas. manual vertical | 97.88mm (45px) | 98.31mm (33px) |
| Mean meas. subpixel horizontal | 97.240mm | 97.264mm |
| Mean meas. subpixel vertical | 96.797mm | 96.815mm |
| True value | 100mm | 100mm |
| Error integer (horizontal axis) | -2.12mm, -2.12% | -1.69mm, -1.69% |

| | | |
|---|---|---|
| Error integer (vertical axis) | -2.12mm, -2.12% | -1.69mm, -1.69% |
| Subpixel error horizontal | -2.760mm, -2.76% | -2.736mm, -2.736% |
| Subpixel error vertical | -3.203mm, -3.203% | -3.185mm, -3.185% |
| Expected error horizontal | -2.12%±(1pixel*scale)mm | -1.69%±(1pixel*scale)mm |
| Expected error vertical | -2.12%±(1pixel*scale)mm | -1.69%±(1pixel*scale)mm |

There are no checker boards floating around in Antarctic waters, hence a direct estimate of the measurement/scale error using ground truth is not possible at water level. However, it is possible to use the width of the ship as a reference for an ad-hoc accuracy test over a larger distance compared to the previous small scale setup. To do so, the horizontal plane (and hence the distance height of the camera) is set to the handrail level. From the ship's drawings it is possible to estimate the distance between the two side handrails present in Figure 28, as well as the height of the camera from the specified plane.

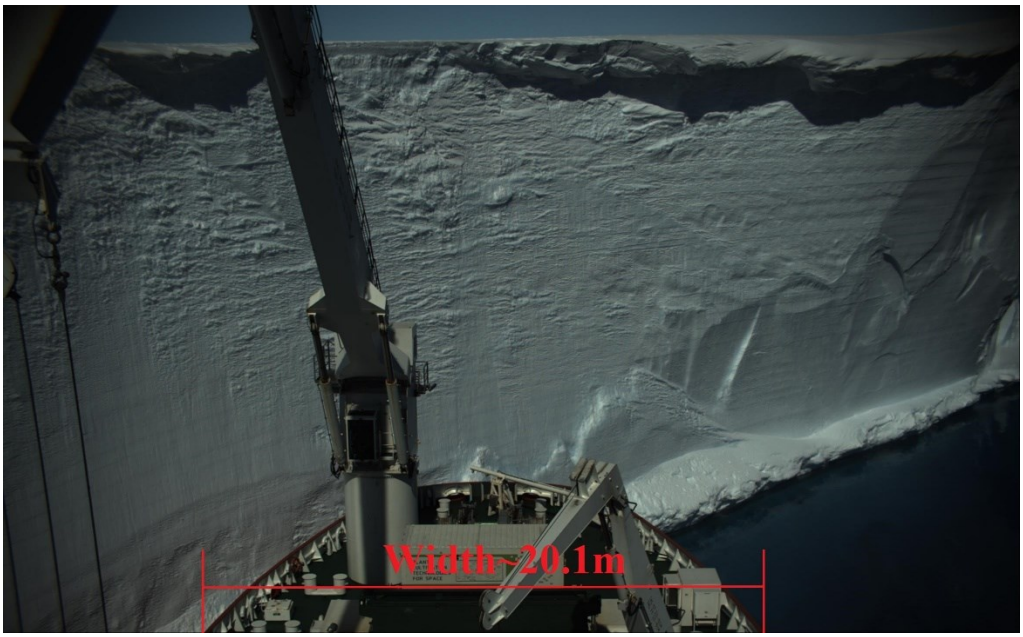


Figure 28. Approximate width of the ship at the last pixel row of the image.

Table 7. Accuracy results of the two orthorectifying methods using the ship's width as a reference.

| | Method | |
|---|---|---|
| Parameter | Geometric | Transformation frame |
| Theta angle | 76° | 76° |
| Height | ~20m | ~20m |
| Scale [m/px] | 0.0515937193 | 0.0633334356 |

| | | |
|---|---|---|
| Manual measurement | 20.79m (403px) | 20.9m (330px) |
| True value | ~20.1m | ~20.1m |
| Measurement error | 0.69m, 3.43% | 0.8m, 3.98% |
| Expected error | 3.43%±0.052m | 3.98%±0.063m |

Next, a total of 10 images taken from the large scale setup were run through both algorithms. An example output from the runs is shown in Figure 29, and then these orthorectified images are overlapped and their pixel wise difference shown in Figure 30.

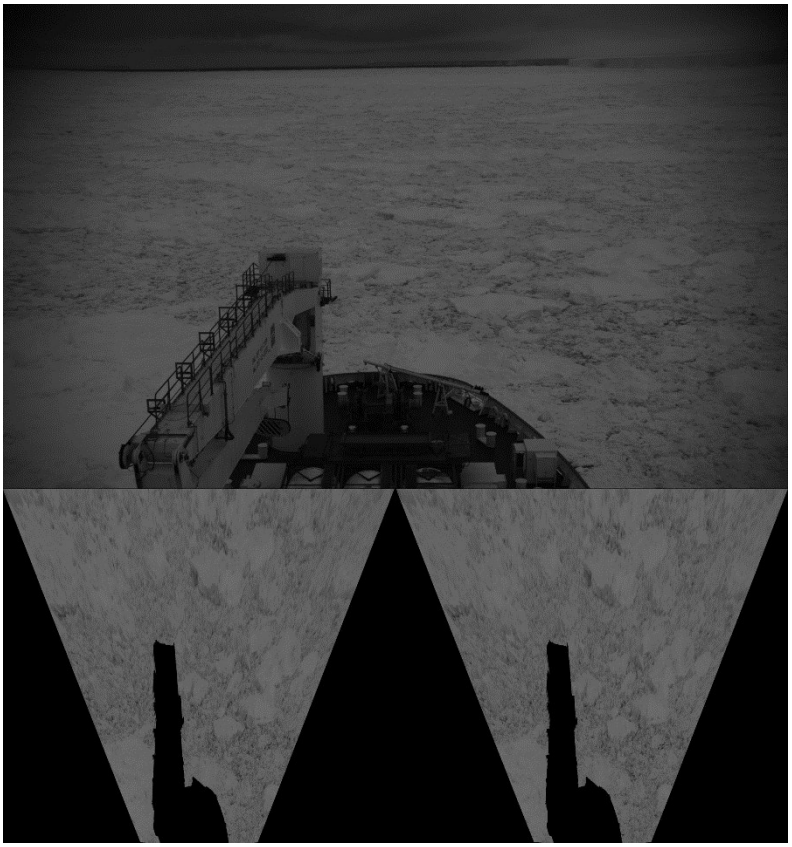

Figure 29. Original image (top) and partial orthorectified images through geometric method (bottom left) and transformation frame (bottom right).

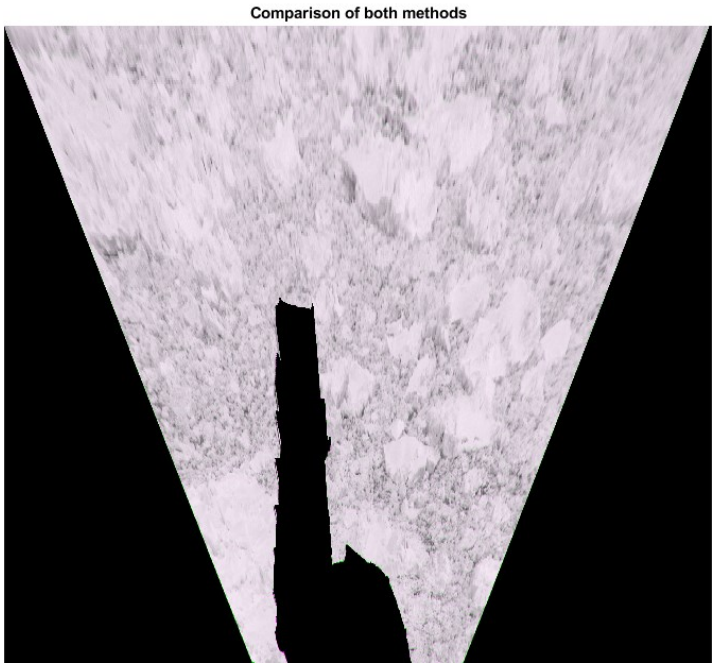


Figure 30. Pixel wise difference between same orthorectified image through geometric and transformation frame methods. Difference appears as green or pink.

Lastly, Figure 31 presents the reader with a comparison of the processing time between both methods for the dataset of 10 images. It is important to note, that in the present case both rectifications took into account only a static transformation (i.e. angle of the camera with respect to the horizontal plane did not change). In the case of the geometric method, an additional ~8s need to be added at the beginning, for computing the rectification matrix. The method based on transformation frames accepts rotations around the camera's principal axis, while the geometric method does not in its current stage. From figure 31 it can be noted that the geometric based orthorectification is at least one order of magnitude slower.

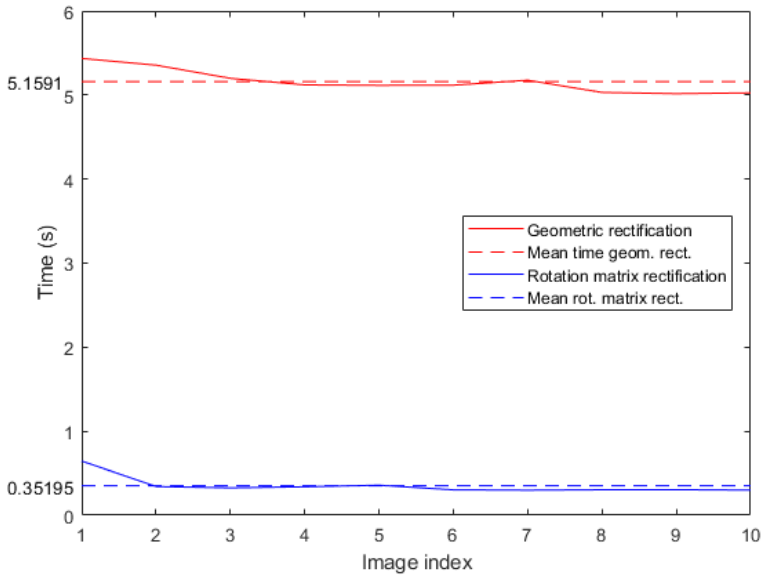


Figure 31. Comparison of required processing time per image and in average between the developed geometric rectification and by means of a rotation matrix and homography.

## 5.2 Experiment B: Intensity and texture analysis comparison

In this section, the three proposed algorithms for discerning ice/snow from slush and water are compared against each other in two common test scenarios, and their respective processing time recorded. For the K-Means algorithm, the function *kmeans* from Matlab is used, as well as for the initialization of the Dynamic Thresholding method and in the Texture based analysis algorithm. The images are pre-processed with methods described in this work and their intensity channel used.

First, both methods based on intensity analysis are compared against each other for consistency, because they share a similar approach regarding the centroids thresholding scheme but not the process to (continuously) obtain them. Their centroids values for water, slush and ice/snow in a series of consecutive images are plot and compared in Figure 32, in the two aforementioned case scenarios: when identifying pack ice in clearly separated ice floes and brash ice conditions. The fourth centroid is not plotted, as it is defined to be 0 at any time instant for both methods.

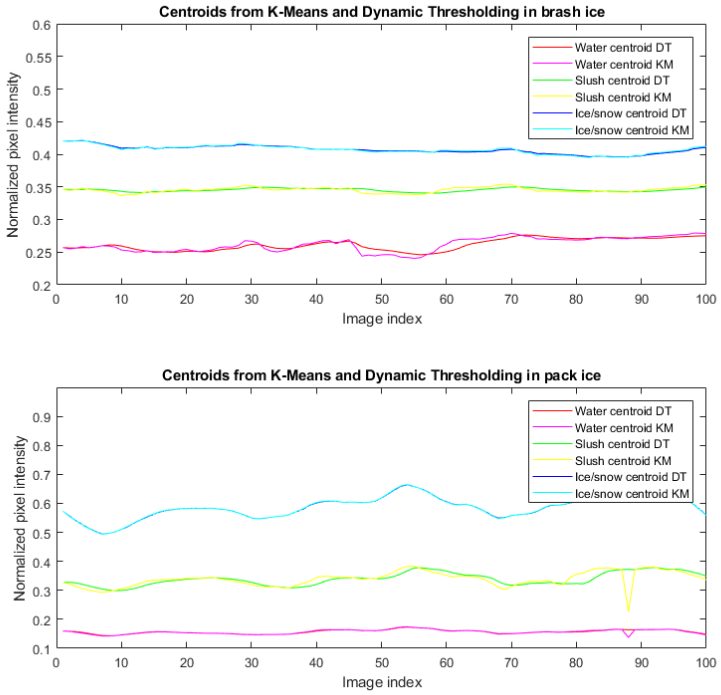


Figure 32. Comparison of identified centroids using K-Means and Dynamic Thresholding methods, for a sample of 100 images in brash ice (top) and pack ice (bottom).

It is important to note, that both intensity based algorithm require the first image to contain all four classes as described earlier, so that the centroids for each class can correctly be identified. In addition, regarding the Dynamic Thresholding method in the case study of ice, slush and open water partitioning, if for example there is no or not enough ice detected in the image, but there is slush, its centroid will be updated by the same amount $\Delta ct_{k+1}^{i-1}$ in (27) as the centroid for the slush class. Therefore, the order in which the classes for other, open water, slush and ice are located with respect to one another matters, and is given by their position on the histogram of an image (from darker to lighter). It was chosen to update the centroid of a class based on the one from a lower intensity centroid at first, following the idea that if nothing else, there will always be open water.

In the case of the Texture Analysis method, the function *entropyfilt* from Matlab was used to obtain the pixel-wise entropy of the image. The function uses a neighbourhood around the pixel under consideration to compute its entropy. The shape of the neighbourhood used is *disk*, since it presents, arguably, the best fitting shape inside a generic ice floe from brash ice. Regarding its size, a radius value of 9 pixels was found empirically to present the best results in the current study and for the used image resolution. Figure 33 presents an example of an image analysed with *entropyfilt*, where it can be noted that areas with a smoother texture in the original image become

darker in the analysed one, as the entropy or randomness is low. On the contrary, in such areas where there is a large variation around a pixel's neighbourhood, its entropy value increases and therefore its pixel intensity becomes lighter in the output image. As a result, the initial theorem for the possibility of successfully using texture based analysis in sea-ice floe identification is sustained (that is, at least for the presented case study).

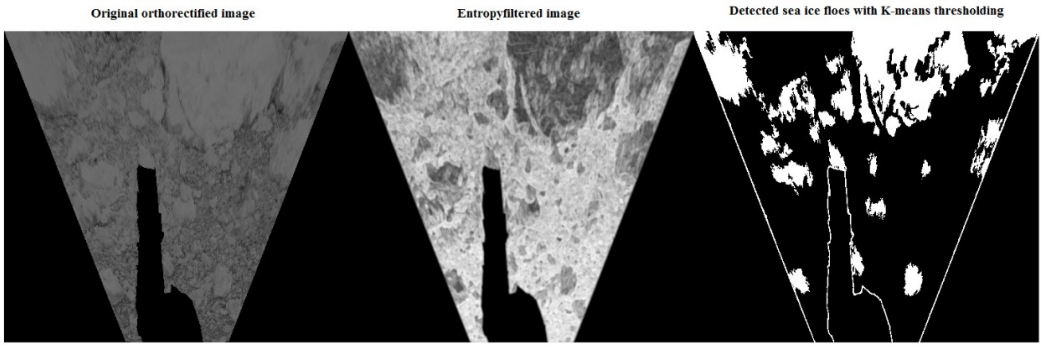


Figure 33. Original image (left), result after running *entropyfilt* algorithm on it (centre) and detected floes (right).

Next, the reader is presented in Figure 34 with the identification results from all three methods in examples of the two most common ice conditions, that is, brash ice conditions in the top row and pack ice conditions in the bottom row. Figure 34 highlights the accuracy of all three methods in discerning between open water (marked as blue for the K-means and Dynamic Thresholding algorithms, and black in the case of Texture Analysis), slush (marked as green and only identifiable by the K-means and Dynamic Thresholding methods) and lastly snow or ice (defined as white in all three algorithms).

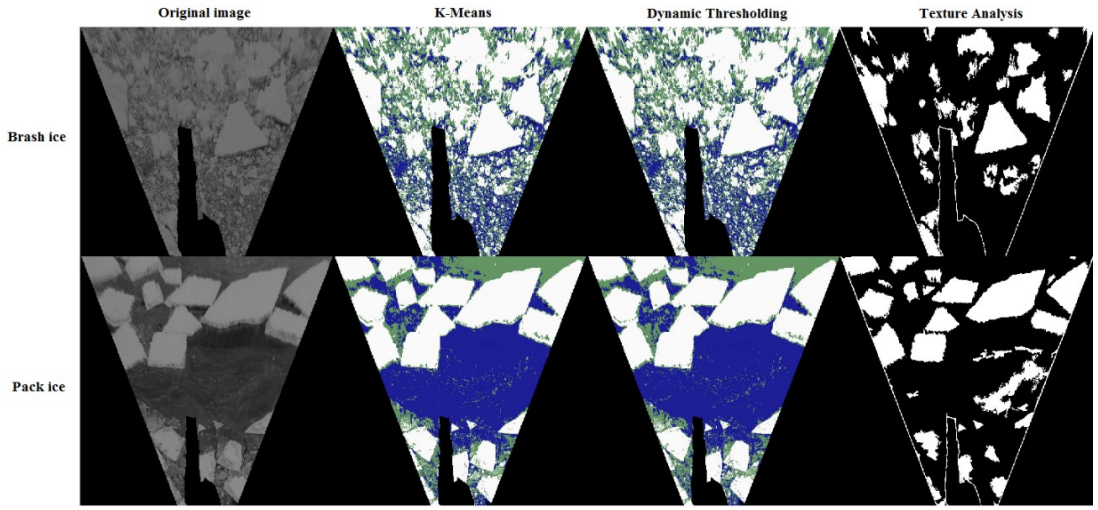


Figure 34. Original images (first column) and their RGB segmentation using K-Means (second column) and Dynamic Thresholding (third column) where blue represents open water, green slush and ice/snow is presented as white. The fourth column corresponds to Texture Analysis, where black represents open water and white ice/snow.

During the previous operations, the processing time of individual images was recorded as well, and is plot in Figure 35 as a comparison of the processing footprint of each method. From the figure, it is clear that Dynamic Thresholding outperforms the K-means algorithm in both of the conditions in which they were put to test, by at least an order of magnitude in the processing times. Furthermore, the processing time for K-Means behaves in a non-deterministic manner, while Dynamic Thresholding maintains a stable one, which may be affected only by external agents (i.e. the operating system or other running applications). In a similar fashion, Texture Analysis maintains a deterministic processing time during the experiment, which may be altered once again only by external factors.

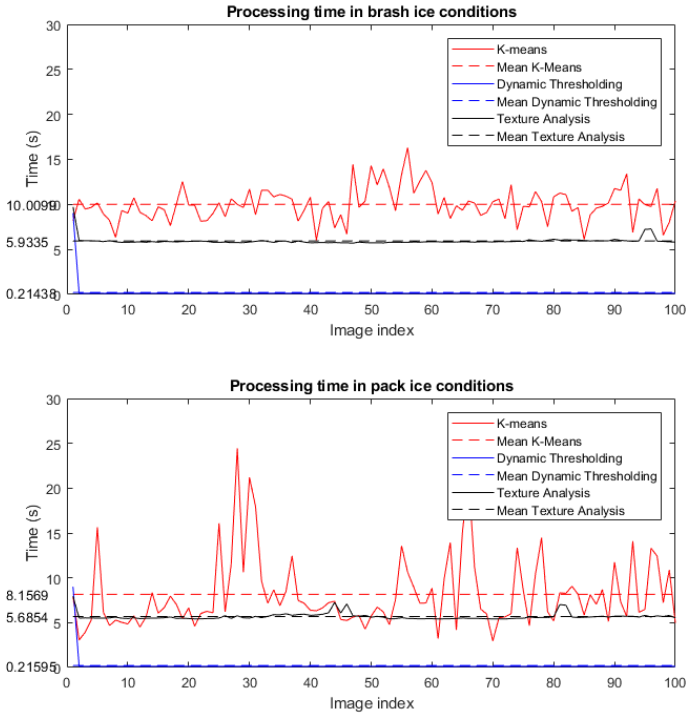


Figure 35. Comparison of processing times for individual images in a sequence, for K-Means and Dynamic Thresholding methods.

Lastly, it is worth mentioning that initially the proposed intensity based algorithms (K-Means and Dynamic Thresholding) were tested on a toy image data set provided by M. Lensu from FMI. The images were taken during 2016 from a drone by J. Lehtiranta. Aforesaid data had already been pre-processed and therefore did not display any vignette effect. However, it was soon noted a vignette effect on the images captured specifically for the present thesis work by means of the procured cameras. Such effect greatly influences the floes detection ability of pixel-intensity based algorithms. As can be seen in Figure 36, an orthorectified image which presents a vignetting effect induces the algorithm into wrongly classifying snow or ice on the border of the image as

slush or even open water, which does not occur when the vignetting effect is removed and therefore the image presents an overall normalized illumination.

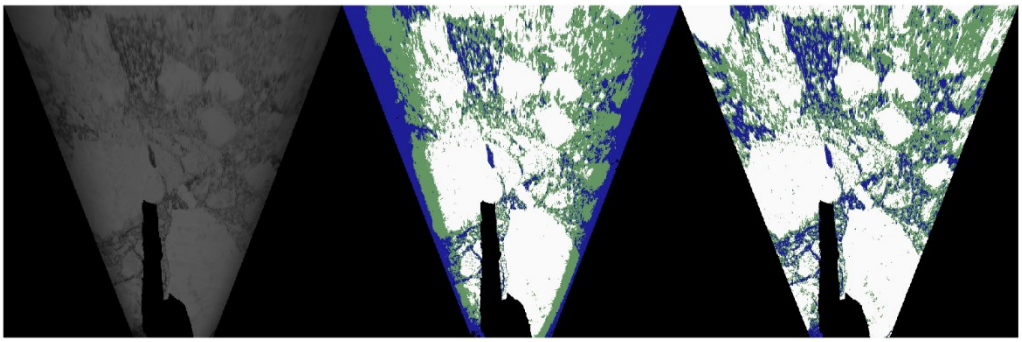

Figure 36. Orthorectified image (left) and example of the K-Means algorithm run on the original image (middle), and on the same image with the vignetting effect removed (right).

## 5.3 Experiment C: Floes detection accuracy

In this experiment, the proposed algorithms of Dynamic Thresholding and Texture analysis are compared against manually identified sea ice floes in a small number of randomly selected orthorectified images. The goal of the experiment is to determine the accuracy of both methods in identifying sea ice floes, compared to the manually selected ones, which are used as the ground truth. For illustration purposes, only the results of two cases (i.e. images) are presented here, while additional results can be found in *Appendix C*.

Based on the assumption that the orthorectification methods produce accurate results, as previously proven and since there were no means to obtain any ground truth in the form of physical dimensions of ice floes, a different approach was engaged in order to determine the accuracy of the overall boundaries of identified ice floes by manually identifying the floes in an orthorectified image. As such, a simple program in Matlab was developed where the user is presented with the original orthorectified image and asked to identify ice floes, one by one, by selecting as many as possible points belonging to the floe's perimeter. Then, all individually identified and labelled sea-ice floes are combined in one image and analysed using the same algorithm proposed in the post-processing method.

Figure 37 presents the reader with the output results of identifying sea-ice as a comparison between the manually identified floes and through two of the proposed algorithms, in two cases corresponding to images 3 and 6 from the total set analyzed. As it may be noted, Dynamic thresholding identifies sea-ice floes in *Image 3* to a high degree of accuracy, while it suffers from a small amount of over-segmentation in certain cases. For the same image, Texture Analysis

presents a poorer performance in correctly identifying the boundaries of sea-ice floes and may even add holes inside them. By contrast, in *Image 6* Dynamic Thresholding suffers heavily from over-segmentation and fails to correctly identify the sea-ice boundaries, while Texture Analysis presents a closer match overall to the manually identified figure.

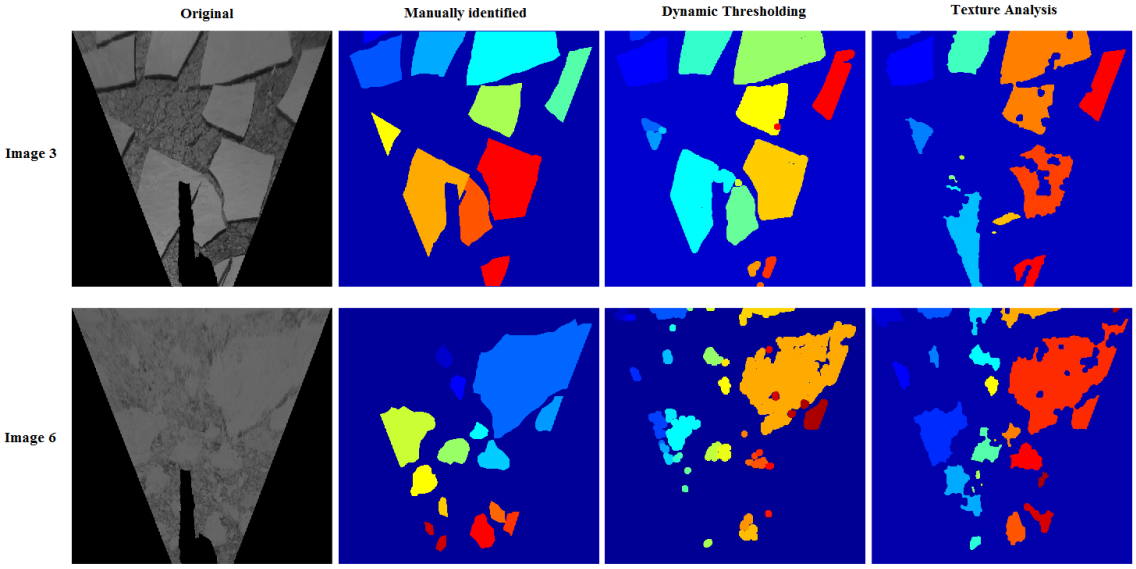


Figure 37. Sea-ice floes identifications results in two different ice conditions. First column presents the original image, second one the manually picked floes, and third and fourth columns presents the results of the Dynamic Thresholding and Texture analysis algorithms, respectively.

Next, statistics of the identified sea-ice floes in the above two cases are presented in the form of histograms for the Major-axis (Figure 38), Minor-axis (Figure 39) and Areas (Figure 40).

In Figure 38, in the top row (pack ice conditions, image 3) it can be appreciated that the Dynamic Thresholding histogram plot for the major axis of the detected floes follows closely the histogram from the manually identified ice floes between value of 400-800 pixels in length. However, under 400, Dynamic Thresholding suffers from over-segmentation. In the case of Texture Analysis, it suffers both from over-segmentation and under-segmentation (detection of inexistent large ice floes). Nevertheless, in the bottom row corresponding to image with index 6, Texture Analysis has values for the histogram closer to the manually picked one, while Dynamic Thresholding presents an extreme case of over-segmentation (large amount of detected ice floes with a major axis smaller than 200 pixels in length). A similar case occurs as well in Figure 39, where the minor axes of the detected sea-ice floes are plot as histograms.

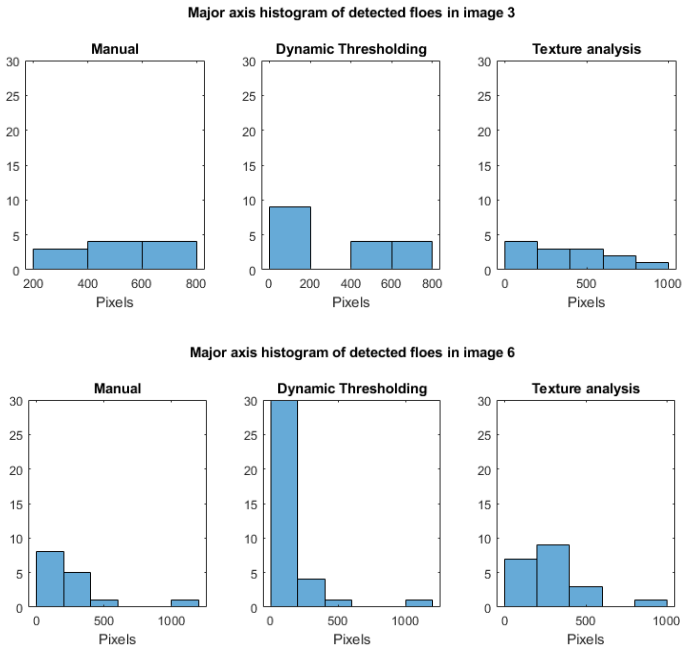


Figure 38. Histograms of the detected major axis in images 3 (top) and 6 (bottom) from the experiment test set, using a bin width of 200 pixels.

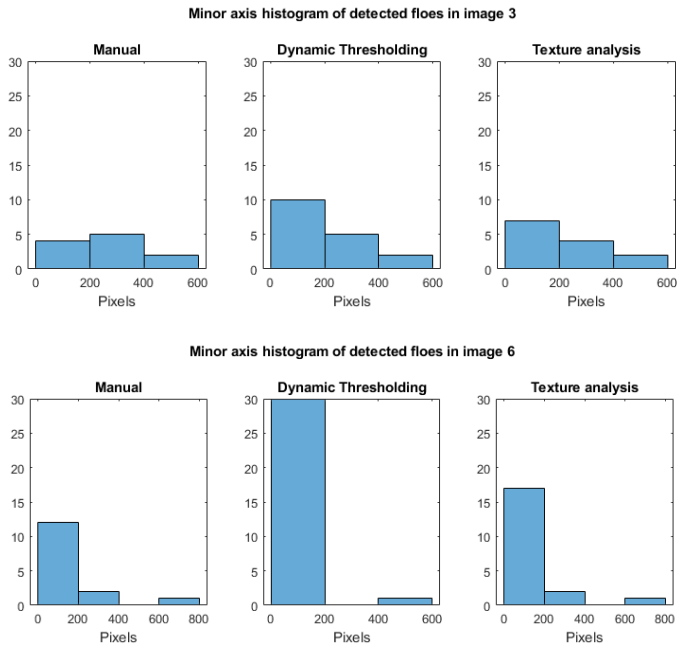


Figure 39. Histograms of the detected minor axis in images 3 (top) and 6 (bottom) from the experiment test set, using a bin width of 200 pixels.

Following with the area estimates in Figure 40, it presents fairly expected results based on the estimated major and minor axes as previously described: in the case of pack ice (first row),

Dynamic Thresholding offers more reliable results (although with a slight over-segmentation tendency), while in brash ice conditions (second row) Texture Analysis presents better estimates.

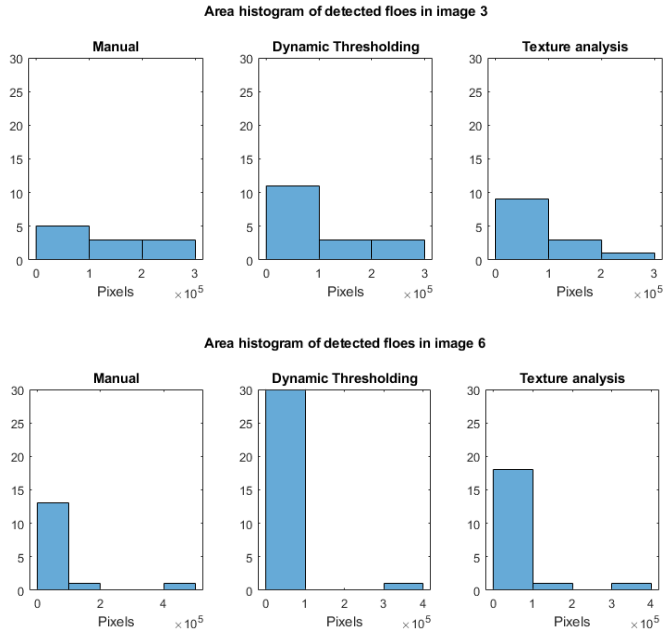


Figure 40. Histograms of the detected areas in images 3 (top) and 6 (bottom) from the experiment test set, using a bin width of 100000 pixels.

Lastly, in Figure 41 a comparison is presented of the mean values obtained from the whole set of analyzed images for major axis, minor axis, area and sea-ice floes concentration. The difference in values between the analysis of the manually identified sea-ice floes, Dynamic Thresholding and Texture Analysis varies according to the ice conditions present in the image. As such, in average terms the results from the proposed algorithms are closest to the manually estimated ones in images 8-10, while they differ the most in image 2. It is important to keep in mind the average nature of the results presented in Figure 41, since over and under-segmentation can greatly influence them and therefore the previously described histogram comparison presents more accurate results (i.e. image 2 presents rather “easy” ice conditions for identifying floes, but since both algorithms falsely detect small floes in addition to the correct ones, averages are pulled towards smaller values thus differing the most from the ground truth).

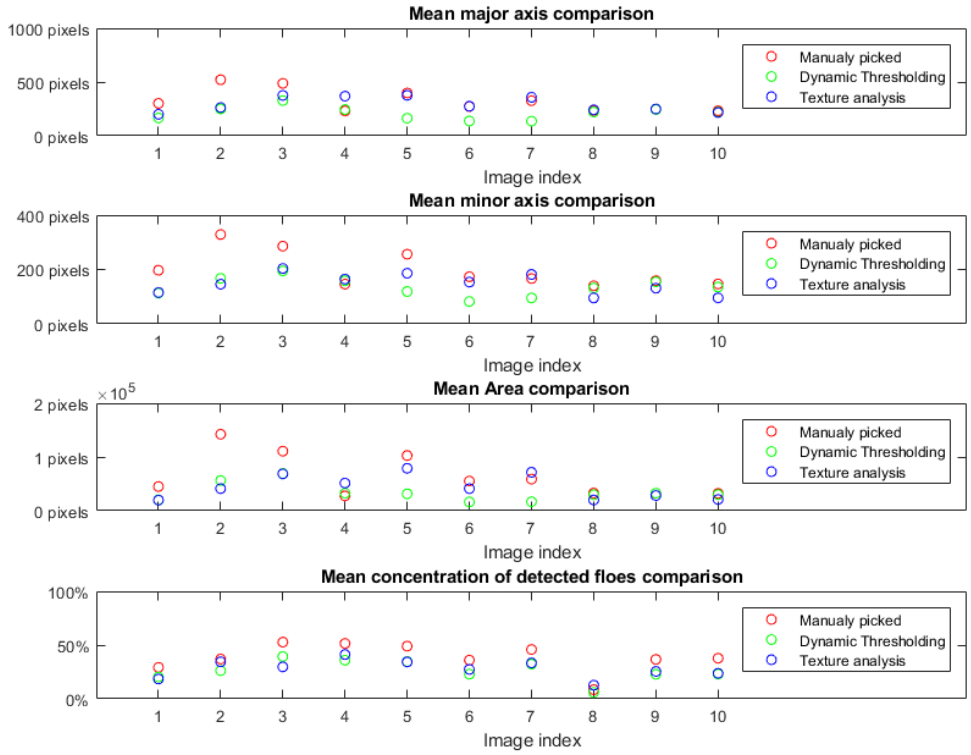


Figure 41. Comparison of the manually selected sea-ice floes (red), Dynamic Thresholding (green) and Texture Analysis (blue) algorithms.

## 5.4 Experiment D: Visual observations vs. Machine vision measurements

During the relief voyage 2017-2018 of the S.A. Agulhas II, a set of ice parameters were observed (while travelling in ice conditions) and annotated at a rate of one observation per minute, which are then averaged over a 10-minutes time window. From these parameters, the important ones for the current case study are floe size and concentration, which will be used in the current section as a comparison reference with the results obtained through the machine vision methods and techniques presented earlier.

In Figures 43-44 and 46-47, visual observations are plot as rectangles of which width corresponds to a time frame of 10 minutes and height corresponds to the current class the rectangle represents; while their colour intensity corresponds to the number of observations lying inside the rectangle's area on a scale from 0 (no observations) to 10 (maximum number of observations during a time period of 10 minutes). For ice concentrations (Figures 43 and 46), the height of the rectangles corresponds to percentages in tens from 0% up to 100% ice concentration and values equal or lower than 0% are considered open water. Concentration classes for manual observations are summed up in Table 8.

Table 8. Concentration classes for manual observations

| Class | Open water | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Concentration | 0% | 0-10% | 10-20% | 20-30% | 30-40% | 40-50% | 50-60% | 60-70% | 70-80% | 80-90% | 90-100% |

In the case of sea ice floes dimensions (Figures 44 and 47), the classes for the heights of the rectangles are divided as presented in Table 9.

Table 9. Sea ice floes diameter classes for manual observations

| Class | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Floe size diameter | 0-20m | 20-100m | 100-500m | 500-2000m | 2000-5000m | 5000m< |

From all the possible time frames of the voyage, two in particular were chosen displaying brash ice and pack ice conditions. These represent, arguably, the best circumstances for comparing the approaches defined in the Dynamic Thresholding and Texture Analysis methods. Regarding the K-Means method, it was left out because of its similarity to the Dynamic Thresholding method, and computational time requirements.

In Figure 42, an example image capture from the first case scenario is presented as an output from both algorithms. For each algorithm and image to be analysed, a mosaic is generated displaying the original image (top-left), its segmentation into classes (top-right), the detected sea ice floes (artificially RGB coloured based on their label as a distinction mark among ice floes, at the bottom-left) and lastly, on the bottom-right average statistics of the performed ice field analysis and image timestamp are displayed.

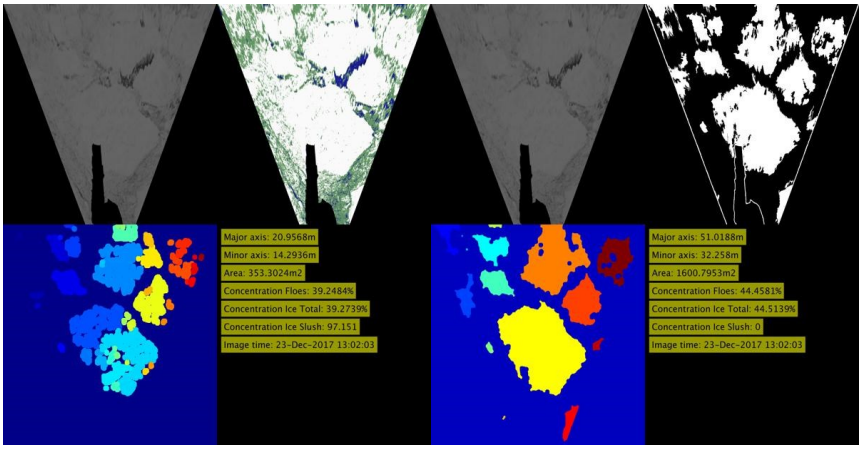


Figure 42. Resulting image from running Dynamic Thresholding (left mosaic) and Texture Analysis (right mosaic). In each mosaic, top left image presents the original orthorectified image, top-right presents the classified pixels in the image, bottom-left shows the detected ice floes and bottom-right presents statistics of the image.

In this first case, where brash ice conditions were present, the ice concentration in the time window between 12:00-15:00 hours on 23.12.2017 is shown in Figure 43, followed by the estimated floe sizes in Figure 44. From Figure 43 it can be seen that the total concentration (including ice floes and brash, and named *ConcentrationIceSlush*) is most accurate from the Dynamic Thresholding algorithm, compared to visual observations, which are plot as an intensity based block in tens of percentages over a timespan of ten minutes, as previously detailed. Floe concentration (green and blue lines) from both algorithms fails short when compared to visual observations. Additionally, it is worth pointing out the close correlation between the concentration from the identified and analysed ice floes after the post-processing operations (marked in blue), and the concentration of the total ice and snow obtained from the processing methods (marked in green).

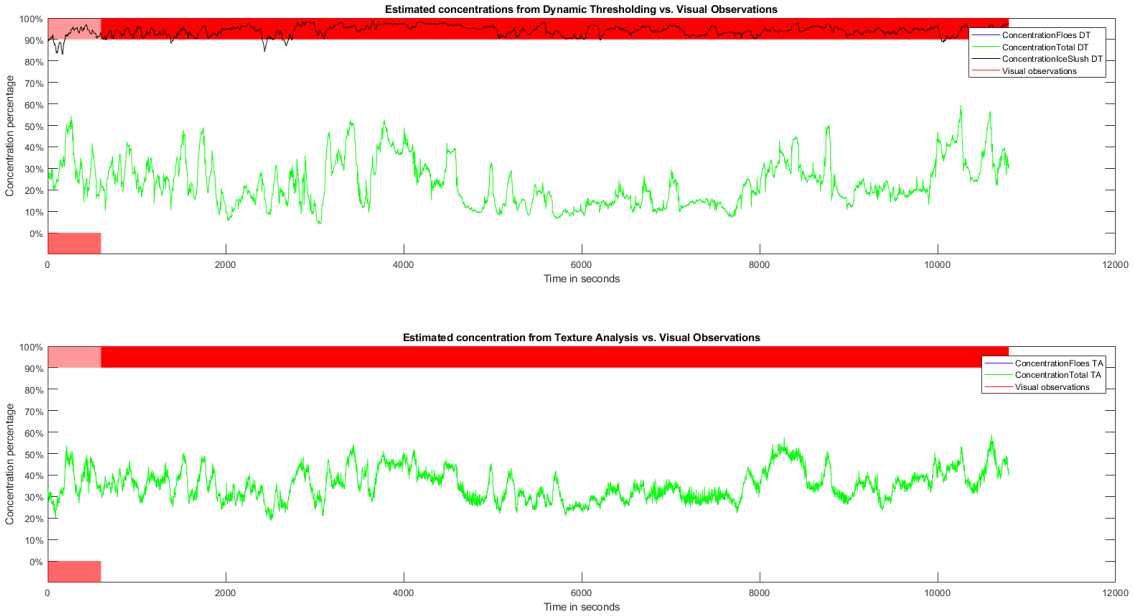


Figure 43. Estimated concentrations vs. visual observations on 23.12.2017 between 12:00-15:00. Blue represents the concentration from the identified and analysed ice floes after post-processing, green is the concentration of the total ice and snow detected in an image, black is the joint concentration between ice and slush, and lastly red is the concentration according to visual observations.

Continuing in the same case scenario, the ice floe dimensions from visual observations were plot in Figure 44 and compared to both algorithms under study. Data from the algorithms was plot first in raw (major and minor axes of each detected floe in a time instant, i.e. in an image) and then averaged over the same time instant. Overall, the mean value from Texture Analysis (dark blue and green) correlates best with visual observations, while Dynamic Thresholding presents a low average during the test.


Figure 44. Estimated ice floe dimensions vs. visual observations on 23.12.2017 between 12:00-15:00. Dark blue represents the mean major axis dimension and green the minor axis one for a time instant, while cyan and yellow represent dimensions for major axis and minor axis respectively for individual ice floes and time instants. Red rectangles represent visual observations.

An example capture of the second case scenario is presented in Figure 45, where pack ice conditions with clearly identifiable sea ice floes are present. Then, the ice concentration in the time window between 14:00-17:00 hours on 5.1.2018 is shown in Figure 46, followed by the estimated floe sizes in Figure 47.

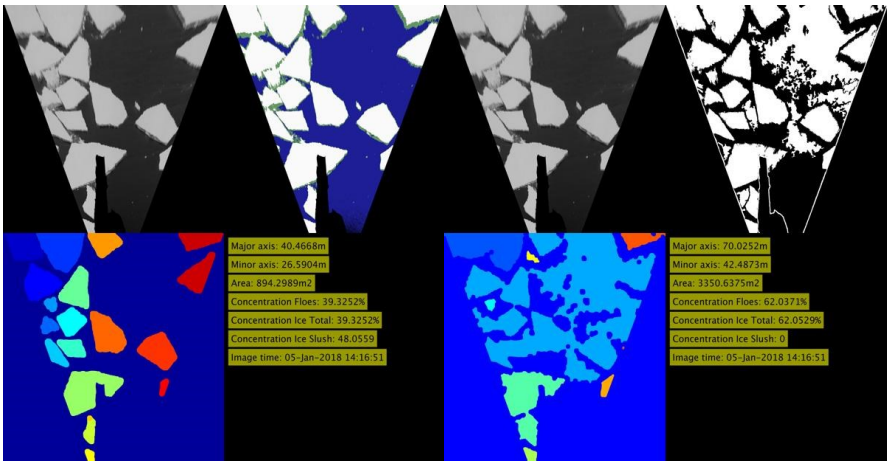


Figure 45. Resulting image from running Dynamic Thresholding (left mosaic) and Texture Analysis (right mosaic). In each mosaic, top left image presents the original orthorectified image, top-right presents the classified pixels in the image, bottom-left shows the detected ice floes and bottom-right presents statistics of the image.

From Figure 46 it can be observed that the concentration reported by the Texture Analysis algorithm vaguely follows the one from visual observations only at the very beginning of the experiment (first three columns of the visual observations, i.e. 30 minutes), after which behaves

in a random fashion when compared to visual observations. On the other hand, the floe concentration (green line) reported by the Dynamic Thresholding method presents a closer correlation to visual observations, while the total concentration reported of ice and slush (black line) only correlates to visual observations for the first half of the experiment, after which presents severe overshot in some cases. Lastly, once again the identified floe concentration and total ice and snow detected in an image highly correlate for both algorithms.


Figure 46. Estimated concentrations vs. visual observations on 5.1.2018 between 14:00-17:00. Blue represents the concentration from the identified and analysed ice floes after post-processing, green is the concentration of the total ice and snow detected in an image, black is the joint concentration between ice and slush and lastly, red is the concentration according to visual observations.

Following with the sea-ice floes dimensions depicted in Figure 47 for pack ice conditions, Texture Analysis presents values which can be considered rather random, while Dynamic Thresholding reports sensible values which correlate with the visual observations.


Figure 47. Estimated ice floe dimensions vs. visual observations on 1.5.2018 between 14:00-17:00. Dark blue represents the mean major axis dimension and green the minor axis one for a time instant, while cyan and yellow represent dimensions for major axis and minor axis respectively for individual ice floes and time instants. Red rectangles represent visual observations.

# 6. Discussion

In the present chapter, the results of the experiments performed and described previously are discussed individually.

## 6.1 Image pre-processing: vignetting

As previously shown, the results from the method in [41] and its implementation [42] continued to present a vignetting effect. The reason theorized behind this culprit was that the implemented algorithm assumes the optical centre of the image and the "physical" one to lie on the same point. This theory is supported by the fact that the captured images were cropped before saving them and because of the halo effect formed in the image. The source code does not allow specifying an image optical centre, and numerous attempts to modify it were unsuccessful: the image would tend to overexposure, while the vignette effect remained.

Since the resulting image from the algorithm using a radial fit on an empirically obtained de-vignetting mask, presented no noticeable vignetting effect on a visual scale, it was deemed as arguably the best method among the proposed ones. However, an extended test in controlled conditions should be performed in order to determine the actual accuracy of the de-vignetting mask through formal methods.

## 6.2 Image pre-processing: geometric transformations

From the miniature experiment with known parameters, it was proved that both orthorectifying methods (based on pure geometrical relations and homography), successfully accomplish the task of restoring the true shape of a flat surface (calibration target lying on the floor). As expected, any and all objects which grow above the orthorectifying plane (in this case, the floor) are deformed with elongation. Then, using a scaling factor and subpixel interpolation, the world dimensions can be obtained with a high degree of accuracy. Since in the following experiments, specifically sea-ice floes identification, subpixel measurements are not possible, manual measurements of the calibration target were carried out as well, which once again indicated a high degree of accuracy. It is important to note here that the same scaling factor does not work for objects which do not lie flat on the orthorectification plane (in this case, the floor). Using the round table as an example, applying the calculated scaling factor would result in erroneous measurements, since the height between the camera and the table surface differs from the height between the camera and the floor.

The scale of the miniature experiment is one order of magnitude smaller, hence a more appropriate experiment was carried on a larger scale. In this experiment, the width of the ship was estimated from an image and compared to its true value, obtained from technical drawings. Once again, the methods proved to be highly accurate given the cameras resolution, since in the case of the homography based one it was 0.8m error versus 0.69m of the geometry one. By extension, once the height of the camera is set with respect to the water line, it is expected that measurements carried on the water level plane to be accurate. Furthermore, the expected error in such measurements can be further reduced by increasing the camera's image resolution (i.e. smaller discretization level).

Lastly, both methods were compared against each other on a series of images, and the following conclusion obtained: first, there is no noticeable pixel-wise difference between the methods for the orthorectified images making them equivalent; and second, processing time for the geometric orthorectification is at least one order of magnitude larger than the one required for orthorectification through a homography matrix. Among other reasons, a strong justification for the slower processing of the developed algorithm would be a more inefficient handling of pixel-wise manipulation and the additional geometry calculus. Therefore, orthorectification through homography is deemed to be the most sensible method. Even more, its processing time can be further optimized by using other programming languages and libraries, such as C/C++ and OpenCV [80].

## 6.3 Intensity and texture analysis comparison

In this experiment, at first both intensity based methods were compared, since they share a similar approach for discerning between ice, slush and open water. Given that both methods use centroids for thresholding among classes, it is sensible to focus the comparison on their estimated mean or centroid values for each class. From the experiment, it can be deduced that both methods have similar estimates for each class and image in the sequence, especially in the case of the snow/ice class. Therefore, it is safe to assume that both methods are equivalent. It is important to note though, that the Dynamic Thresholding method presents a smoother transition between centroid values, making it more robust against sudden changes in the overall intensity of an image, but it may also mean that classes in such images are wrongly discerned and making the method not reliable in such cases.

As it was noted in *Section 5.2*, both intensity based methods require all the classes to be present in the first image for their initialization, which certainly adds an important limitation. Then, a

more robust method is required for overcoming such situations where one or two of the classes are missing in the image sequence. Further development is necessary in order to overcome such drawbacks and reach a fully autonomous algorithm in all situations. In addition, updating certain classes' centroids based solely on others centroids induces one addition challenge which was noted while running the algorithm: if there is only open water in the image, its class will gradually pull the other classes' centroids towards itself, resulting in that the algorithm will falsely detect slush and/or ice and snow. A simple solution, which works well where open water fills the whole image for a short sequence of images, is to use the open water percentage as a thresholding method whether to update the centroids from other classes not present in the image, or leave them as they are.

Lastly, all three methods were tested in pack ice and brash ice conditions, two of the most common encountered in Antarctic waters. The results suggest that Texture Analysis performs best in such conditions with high concentration of brash ice since there is less chance of false identification of open water as ice. On the other hand, K-means and Dynamic Thresholding algorithms are more suitable for pack ice conditions where certain amount of open water is present as well as ice floes, which help to maintain their centroid updates.

As it was proved earlier their similarity in ice identification, Dynamic Thresholding becomes the most sensible choice among both algorithms. Regarding the Texture Analysis algorithm, since it is based on a "pure" version of the K-means algorithm for thresholding the entropy image, its processing times are rather high and far from the aimed real-time operation. Further analysis and development is required in order to automate the thresholding process in the Texture Analysis algorithm by other means, less expensive computationally.

## 6.4 Floes detection accuracy

The results of this experiment have the following deductions: as it can be seen in *Appendix C,* Figure D1 once again it has been proven that Dynamic Thresholding performs best in pack ice conditions where ice floes are clearly distinguishable, while it suffers from over-segmentation in brash ice conditions; in contrast to the Texture Analysis method, which performs best in situations with high concentration of ice and brash and rather poorly in pack ice conditions.

Continuing with Figure D1, its results make apparent that further improvements are required. One such improvement, which would increase the ice floe boundary detection, would be to choose the structuring element for the morphological operations based on the ice type. Since pack ice presents approximately straight lines in the ice floes contours, a *diamond* type of structuring

element would be more suitable. On the other hand, brash ice presents rounder borders, hence the *disk* type continues to be a sensible choice. In addition, an off-line method may be used in order to further tune the detection of sea-ice floes boundaries, such as an active contour model [81].

Based on the histogram results from *Section 5.3* and *Appendix C*, it becomes apparent that over-segmentation can present a challenge in accurately identifying ice floes and gathering accurate statistics of the ice field. One simple solution would be to use a thresholding method based on size (e.g. remove any detected ice floes under certain limit, for example 20 metres across). On the other hand, under-segmentation can in turn bias the acquired statistics towards higher values, and it presents its own challenges: a human observer may use other sources of information in order to discern whether two closely connected ice floes form two separate entities or one single ice floe. Sources of such extra information may be their relative movement against each other or even their shape and length of their connection point (i.e. an hourglass shaped ice floe with a narrow middle part would most certainly be composed by two distinct ice floes). All in all, further development is required in order to effectively remove, or at least minimize, over and under-segmentation.

## 6.5 Visual observations vs. machine vision measurements

In the last experiment, the collected visual observations during the voyage were compared against machine vision estimates for two time intervals and ice type conditions. Overall, the first benefit of machine vision estimates becomes apparent: they offer a much larger temporal resolution compared to visual observations.

Taking into account the tendency of the Dynamic Thresholding algorithm towards over-segmentation and the outperformance from the Texture Analysis algorithm in detecting ice floes in brash ice conditions, it is sensible to assume that the concentration of ice floes provided by the latter algorithm becomes more accurate. Even though this assumption disagrees with the visual observations, since they consider coverage as a whole without discerning between ice and brash, it is an additional benefit from machine vision estimates. In contrast, the total concentration of ice and slush from the Dynamic Thresholding correlates best with the visual observations one. Then, as it was pointed out previously, both the concentration from identified sea ice floes and total ice in an image closely correlate, indicating that the post-processing methods do not influence the estimated ice field statistics.

Continuing in the same case scenario with brash ice conditions, when comparing the sea ice floes dimensions estimates, the following conclusions can be extracted: first, Dynamic Thresholding presents a low average during the whole period of the experiment as a consequence of over-

segmentation, while Texture Analysis presents a larger variety in sea-ice floes dimensions and closer to the visual observations, which is in accordance with the results from *Section 5.2 and 5.3*; and second, in both algorithms (especially in Texture Analysis) larger sea ice floes (above 100 metres in diameter) were detected during the first half of the experiment, which are not present in the visual observations plots, hence another benefit of machine vision becomes apparent.

Next, in pack ice conditions with individually distinguishable sea ice floes and certain amount of open water, the Texture Analysis algorithm behaves in an erratic manner, since it may wrongly identify calm, open water as ice or snow. This is in accordance with previous results for this algorithm and type of ice conditions. However, when comparing the detected snow/ice concentration from Dynamic Thresholding, there is a clear correlation with the visual observations, proving the usefulness of the algorithm. In the case of the reported total concentration including snow/ice and brash ice, its accuracy and correlation to visual observation diminishes in such cases where no or a small amount of snow/ice was detected by the algorithm. An improved version of the algorithm should include a method to detect such cases and, perhaps, remove or mark as not reliable the total estimated concentration.

When comparing dimension estimates in pack ice conditions, Texture Analysis presents values which could be considered random, proving one last time that it is not a suitable algorithm for such ice conditions. On the other hand, Dynamic Thresholding does correlate with visual observations except for few cases. Such cases have been manually analysed from image footage and it was deemed that the estimates from the Dynamic Thresholding method were accurate, while the visual observations presented deficiencies (such as classifying as open water a field where ice floes were present).

All in all, the performance of both methods can be considered as acceptable in certain conditions, however there is room for improvement, especially regarding detection of sea-ice floes and most importantly, division of two or more adjacent floes. In addition, another improvement may derive from combining both methods into one, which would estimate the reliability of each individual method for a particular case scenario and output the results from the one with best expected estimates.

# 7. Conclusion

Along this thesis work a complete process is presented for obtaining dimensions, distribution and concentration of sea-ice floes. This process aims at assisting and improving part of the ice field analysis from on-board visual observations, currently done by human volunteers and therefore liable to human errors and subjective interpretations. An example system setup for collecting the required information is provided as well, which includes a pair of machine vision cameras for image acquisition, one IMU device for determining the dynamic attitude of the cameras with respect to the world, two GPS sensors providing a redundant positioning and clock data, and a desktop computer used as the main logging platform for all the collected data.

The full process for obtaining the (partial) sea-ice field analysis involves numerous, organized steps. At first, images need to undergo a pre-processing phase, where camera artefacts regarding vignetting effect (directly affecting intensity based methods) and lens distortions (undermining orthorectification accuracy) are removed. In the case of the vignetting effect, a method is derived for estimating such effect and removing it through a de-vignetting mask; while the lens distortion is estimated and removed by means of a powerful tool from the Matlab environment. In addition, during the pre-processing phase, images are geometrically orthorectified from their perspective view by means of two distinct algorithms: a derived geometry-based algorithm and a homography-based transformation using transformation frames and rotation matrices. In order to obtain the attitude of the camera with respect to the world, an IMU device is used. The orthorectification process provides a “birds eye view” of the ice field, where sea-ice floes display their true shape.

The pre-processing phase eases or even makes possible the task of discerning between open water, brash ice and snow/ice carried on during the following step. In the processing phase, three algorithms are proposed, which can be run in parallel. The first two are based on intensity analysis and thresholding, and as such interpret the image based on pixel intensity values, disregarding their position in the image or relationship with neighbouring pixels. The third algorithm explores the texture dimension present in an image by performing a local statistical analysis on the neighbourhood of each pixel based on its entropy. The theory behind this approach is that sea-ice floes present a smoother surface than their surrounding brash ice.

Once ice or snow covered floes are detected in an image, their individual perimeters must be determined. The proposed method to do so is based on morphological operations, a heuristic technique where images are processed based on shapes. Specifically, images containing ice are first eroded, a process which shrinks the total area of the detected ice floes but at the same time increases the separation among them. Then, ice floes boundaries can be detected with a simple

algorithm searching for individual objects in an image. Once all the individual sea-ice floes are detected and labelled, the opposite process is used (i.e. dilation), which restores their original area and shape up to a degree. Lastly, by using a scaling factor from the geometric orthorectification method, together with the individually labelled sea-ice floes, it is possible to perform the (partial) ice field analysis.

The comparison between the explored de-vignetting methods suggest that the derived approach performs well, however additional tests are required for validation. Regarding image orthorectification, the accuracy of both algorithms has been tested and proved, though the homography based approach outperforms in computation efficiency the geometry based one by an order of magnitude. Lastly, through a series of experiments, the performance of the full process was examined by first comparing it to manually identified sea-ice floes from a series of orthorectified images and then to the on-board visual observations performed by volunteers. The results indicate that intensity based algorithms perform well in pack ice conditions, where individual floes are distinguishable and surrounded by open water, and perform poorly in brash ice conditions with an accentuated over-segmentation problem. In contrast, the texture based algorithm performs poorly in detecting ice floes in pack ice conditions, while it manages to successfully detect ice floes in brash ice conditions, with a high concentration of ice present.

All in all, the results from this thesis work provide a solid background on the feasibility of using machine vision to perform ice field analysis and a basis for further developments and improvements. One such improvement could be the use of a different environment (e.g. object programming through C++) in order to improve the computational efficiency of the process. Additionally, the collected data for the present thesis provides a solid ground for exploring sensor fusion methods, for example between visual odometry and IMU measurements or even with the estimated ship's attitude from the horizon line. Lastly, the obtained visual data can serve as a ground truth for satellite images or in determining the ice field motion.

# References


[1] C. L. Parkinson, “Search for the little ice age in southern ocean sea-ice records,” *Annals of Glaciology*, vol. 14, pp. 221–225, 1990.

[2] A. Worby, “Observing antarctic sea ice: a practical guide for conducting sea ice observations from vessels operating in the antarctic pack ice,” *Scientific Committee on Antarctic Research, Hobart, Tasmania, Australia*, 1999.

[3] L.-K. Soh, C. Tsatsoulis, and B. Holt, “Identifying ice floes and computing ice floe distributions in sar images,” in *Analysis of SAR Data of the Polar Oceans*. Springer, 1998, pp. 9–34.

[4] T. B. Ijitona, J. Ren, and P. B. Hwang, “SAR sea ice image segmentation using watershed with intensity-based region merging,” in *2014 IEEE International Conference on Computer and Information Technology (CIT)*. IEEE, 2014, pp. 168–172.

[5] M. Paget, A. Worby, and K. Michael, “Determining the floe-size distribution of east antarctic sea ice from digital aerial photographs,” *Annals of Glaciology*, vol. 33, pp. 94–100, 2001.

[6] Q. Zhang and R. Skjetne, “Image processing for identification of sea-ice floes and the floe size distributions,” *IEEE Transactions on geoscience and remote sensing*, vol. 53, no. 5, pp. 2913–2924, 2015.

[7] C. Xu and J. L. Prince, “Snakes, shapes, and gradient vector flow,” *IEEE Transactions on image processing*, vol. 7, no. 3, pp. 359–369, 1998.

[8] K.-I. Muramoto, K. Matsuura, and T. Endoh, “Measuring sea-ice concentration and floe-size distribution by image processing,” *Annals of Glaciology*, vol. 18, pp. 33–38, 1993.

[9] B. Weissling, S. Ackley, P. Wagner, and H. Xie, “Eiscam - digital image acquisition and processing for sea ice parameters from ships,” *Cold Regions Science and Technology*, vol. 57, no. 1, pp. 49–60, 2009.

[10] B. Delaunay, “Sur la sphere vide,” *Bulletin de l’Academie des Sciences de l’URSS, Classe des sciences mathematiques et na*, vol. 6, no. 793-800, pp. 1–2, 1934.

[11] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. MIT press, 2011.

[12] “Antarctic treaty system,” Wikipedia. [Online]. Available: https://-en.wikipedia.org/wiki/Antarctic_Treaty_System [Accessed: 2018-07-24].

[13] R. B. Glassman, “Persistence and loose coupling in living systems,” *Behavioral science*, vol. 18, no. 2, pp. 83–98, 1973.

[14] K. Parulski and M. Rabbani, “The continuing evolution of digital cameras and digital photography systems,” in *Proceedings of the 2000 IEEE International Symposium on Circuits and Systems*, vol. 5. IEEE, 2000, pp. 101–104.

[15] H. Maître, *From Photon to pixel: the digital camera handbook*. John Wiley & Sons, 2017.

[16] "Rolling shutter," Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/-Rolling_shutter [Accessed: 2018-07-19].

[17] "ava2300-25gc - basler aviator," Basler. [Online]. Available: https://-www.baslerweb.com/en/products/cameras/area-scan-cameras/aviator/ava2300-25gc/ [Accessed: 2018-07-19].

[18] P. Mechanics, "Robot navigator guides jet pilots," *Popular Mechanics Magazine*, p. 87, May 1954. [Online]. Available: https://books.google.fi/-books?id=oN0DAAAAMBAJ&pg=PA87&dq=1954+Popular+Mechanics+January&hl=en&sa=X&ei=TV0mT-7WMoGftwfi4dizCg&redir_esc=y#v=onepage&q=1954%20Popular%20Mechanics%20January&f=true [Accessed: 2018-07-28].

[19] R. H. Parvin, *Inertial navigation*, ser. Principles of guided missile design. Van Nostrand, 1962.

[20] D. Titterton, J. Weston, J. Weston, I. of Electrical Engineers, A. I. of Aeronautics, and Astronautics, *Strapdown Inertial Navigation Technology*, ser. Electromagnetics and Radar Series. Institution of Engineering and Technology, 2004.

[21] P. Zhou, M. Li, and G. Shen, "Use it free: Instantly knowing your phone attitude," in *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 2014, pp. 605–616.

[22] S. LaValle, A. Yershova, M. Katsev, and M. Antonov, "Head tracking for the Oculus Rift," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 187–194.

[23] S. Küchler, C. Pregizer, J. K. Eberharter, K. Schneider, and O. Sawodny, "Real-time estimation of a ship's attitude," in *American Control Conference (ACC), 2011*. IEEE, 2011, pp. 2411–2416.

[24] A. Golovan and A. Cepe, "Small satellite attitude determination based on GPS/IMU data fusion," in *Proceedings of AIP Conference*, vol. 1637, no. 1. AIP, 2014, pp. 341–347.

[25] R. Zhu and Z. Zhou, "A real-time articulated human motion tracking using tri-axis inertial/magnetic sensors package," *IEEE Transactions on Neural systems and rehabilitation engineering*, vol. 12, no. 2, pp. 295–302, 2004.

[26] H. Hyyti and A. Visala, "A DCM based attitude estimation algorithm for low-cost MEMS IMUs," *International Journal of Navigation and Observation*, vol. 2015, 2015.

[27] A. Roberts and A. Tayebi, "On the attitude estimation of accelerating rigid-bodies using GPS and IMU measurements," in *2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*. IEEE, 2011, pp. 8088–8093.

[28] R. B. Widodo and C. Wada, "Attitude estimation using Kalman filtering: external acceleration compensation considerations," *Journal of Sensors*, vol. 2016, 2016.

[29] *3DM-GX3 Data Communications Protocol*, MicroStrain, 2009. [Online]. Available: http://www.alliantech.com/pdf/coin_des_experts/3DM-GX3_protocole_de_communication.pdf [Accessed: 2018-07-19].

[30] *3DM-GX2 Datasheet*, MicroStrain, 2007. [Online]. Available: http://-www.microstrain.com/inertial/3dm-gx2 [Accessed: 2018-07-19].

[31] "GPS: The global positioning system," U.S. Government. [Online]. Available: https://www.gps.gov/ [Accessed: 2018-07-26].

[32] "LEA-5H, LEA-5S, LEA-5A u-blox5 GPS and GALILEO modules," u-blox. [Online]. Available: http://www.mipsasoft.com/MS7/Hardware/Expansiones/Int8/GPS/-GPS%20UBLOX%20LEA-5A.pdf [Accessed: 2018-07-19].

[33] "GPS receiver BU-353S4," GlobalSat. [Online]. Available: https://-www.globalsat.com.tw/en/product-199952/Cable-GPS-with-USB-interface-SiRF-Star-IV-BU-353S4.html [Accessed: 2018-07-19].

[34] P. Corke, J. Lobo, and J. Dias, "An introduction to inertial and visual sensing," *The International Journal of Robotics Research*, 2007.

[35] J. Plumridge, "How to avoid artifacts in digital photos," 2018. [Online]. Available: https://www.lifewire.com/avoid-artifacts-in-digital-photos-493765 [Accessed: 2018-07-19].

[36] *Single camera calibrator app*, Matlab. [Online]. Available: https://-se.mathworks.com/help/vision/ug/single-camera-calibrator-app.html [Accessed: 2018-07-19].

[37] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, 2000.

[38] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

[39] "Vignetting," Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/-Vignetting [Accessed: 2018-07-19].

[40] N. Mansurov, "What is vignetting?" 2018. [Online]. Available: https://-photographylife.com/what-is-vignetting [Accessed: 2018-07-19].

[41] Y. Zheng, J. Yu, S. B. Kang, S. Lin, and C. Kambhamettu, "Single-image vignetting correction using radial gradient symmetry," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008*. IEEE, 2008, pp. 1–8.

[42] Y. Zheng, "nu_corrector," MathWorks, 2010. [Online]. Available: https://-se.mathworks.com/matlabcentral/fileexchange/27315-nu_corrector [Accessed: 2018-07-19].

[43] S. Lyu, "Estimating vignetting function from a single image for image authentication," in *Proceedings of the 12th ACM workshop on Multimedia and security*. ACM, 2010, pp. 3–12.

[44] S. B. Kang and R. Weiss, "Can we calibrate a camera using an image of a flat, textureless lambertian surface?" in *European conference on computer vision*. Springer, 2000, pp. 640–653.

[45] W. Babcock and E. Czyryca, "The role of materials in ship design and operation," *The AMPTIAC Quarterly*, vol. 7, no. 3, pp. 31–36, 2003.

[46] "Basler pylon camera software suite," Basler. [Online]. Available: https://-www.baslerweb.com/en/products/software/basler-pylon-camera-software-suite/ [Accessed: 2018-07-19].

[47] The IEEE and The Open Group, *The Open Group Base Specifications Issue 7 – IEEE Std 1003.1, 2017 Edition*. IEEE, 2017. [Online]. Available: http://pubs.opengroup.org/-onlinepubs/9699919799/basedefs/V1_chap04.html#tag_04_16 [Accessed: 2018-07-19].

[48] T. Bajd, M. Mihelj, J. Lenarcic, A. Stanovnik, and M. Munih, *Robotics*. Špringer Science & Business Media, 2010, vol. 43.

[49] *3DM-GX3 Datasheet*, MicroStrain, 2012. [Online]. Available: http://-www.microstrain.com/inertial/3dm-gx3-25 [Accessed: 2018-07-19].

[50] *World Geodetic System 1984*, National Geospatial-Intelligence Agency. [Online]. Available: http://www.unoosa.org/pdf/icg/2012/template/WGS_84.pdf [Accessed: 2018-07-19].

[51] A. Owen-Hill, "Robot vision vs computer vision: What's the difference?" 2016. [Online]. Available: https://blog.robotiq.com/robot-vision-vs-computer-vision-whats-the-difference [Accessed: 2018-07-19].

[52] "Computer vision," Wikipedia. [Online]. Available: https://en.wikipedia.org/-wiki/Computer_vision [Accessed: 2018-07-19].

[53] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms In MATLAB*. Springer, 2011, vol. 73.

[54] P. H. Torr and A. Zisserman, "Mlesac: A new robust estimator with application to estimating image geometry," *Computer vision and image understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[55] E. Vincent and R. Laganiére, "Detecting planar homographies in an image pair," in *Proceedings of the 2nd International Symposium on Image and Signal Processing and Analysis (ISPA), 2001*. IEEE, 2001, pp. 182–187.

[56] A. Goshtasby, "Image registration by local approximation methods," *Image and Vision Computing*, vol. 6, no. 4, pp. 255–261, 1988.

[57] T. Pavlidis, "Segmentation of pictures and maps through functional approximation," *Computer Graphics and Image Processing*, vol. 1, no. 4, pp. 360–372, 1972.

[58] C. R. Brice and C. L. Fennema, "Scene analysis using regions," *Artificial intelligence*, vol. 1, no. 3-4, pp. 205–226, 1970.

[59] T. Balaji and D. M. Sumathi, "Effective features of remote sensing image classification using interactive adaptive thresholding method," *arXiv preprint arXiv:1401.7743*, 2014.

[60] R. E. Brandt, S. G. Warren, A. P. Worby, and T. C. Grenfell, "Surface albedo of the antarctic sea ice zone," *Journal of Climate*, vol. 18, no. 17, pp. 3606–3622, 2005.

[61] M. C. Zatko and S. G. Warren, "East antarctic sea ice in spring: spectral albedo of snow, nilas, frost flowers and slush, and light-absorbing impurities in snow," *Annals of Glaciology*, vol. 56, no. 69, pp. 53–64, 2015.

[62] R. Lei, X. Tian-Kunze, M. Leppäranta, J. Wang, L. Kaleschke, and Z. Zhang, "Changes in summer sea ice, albedo, and portioning of surface solar radiation in the pacific sector of arctic ocean during 1982–2009," *Journal of Geophysical Research: Oceans*, vol. 121, no. 8, pp. 5470–5486, 2016.

[63] Y. Feng, Q. Liu, Y. Qu, and S. Liang, "Estimation of the ocean water albedo from remote sensing and meteorological reanalysis data." *IEEE Transactions Geoscience and Remote Sensing*, vol. 54, no. 2, pp. 850–868, 2016.

[64] "K-means clustering," Wikipedia. [Online]. Available: https://en.wikipedia.org/-wiki/K-means_clustering [Accessed: 2018-07-19].

[65] S. Lloyd, "Least squares quantization in PCM," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[66] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[67] *Matlab documentation: kmeans*, Matlab. [Online]. Available: https://-se.mathworks.com/help/stats/kmeans.html [Accessed: 2018-07-19].

[68] H. Hyyti, J. Kalmari, and A. Visala, "Real-time detection of young spruce using color and texture features on an autonomous forest machine," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2013, pp. 1–8.

[69] F. Zhu, D. Zhang, Y. He, F. Liu, and D.-W. Sun, "Application of visible and near infrared hyperspectral imaging to differentiate between fresh and frozen–thawed fish fillets," *Food and Bioprocess Technology*, vol. 6, no. 10, pp. 2931–2937, 2013.

[70] A. Materka, M. Strzelecki *et al.*, "Texture analysis methods–a review," *Technical university of lodz, institute of electronics, COST B11 report, Brussels*, pp. 9–11, 1998.

[71] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE mobile computing and communications review*, vol. 5, no. 1, pp. 3–55, 2001.

[72] *Matlab documentation: entropyfilt*, Matlab. [Online]. Available: https://-se.mathworks.com/help/images/ref/entropyfilt.html [Accessed: 2018-07-19].

[73] P. Peterlin, "Morphological operations: an overview," University of Ljubljana, 1996. [Online]. Available: http://www.inf.u-szeged.hu/ssip/1996/morpho/morphology.html [Accessed: 2018-07-19].

[74] *Image Processing Toolbox*, Matlab. [Online]. Available: https://-se.mathworks.com/products/image.html [Accessed: 2018-07-19].

[75] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to mav navigation," in *International Conference on Intelligent Robots and Systems (IROS), 2013*. IEEE, 2013, pp. 3923–3929.

[76] P. Gemeiner, P. Einramhof, and M. Vincze, "Simultaneous motion and structure estimation by fusion of inertial and vision data," *The International Journal of Robotics Research*, vol. 26, no. 6, pp. 591–605, 2007.

[77] J. Kelly and G. S. Sukhatme, “Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration,” *The International Journal of Robotics Research*, vol. 30, no. 1, pp. 56–79, 2011.

[78] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart, “Fusion of imu and vision for absolute scale estimation in monocular SLAM,” *Journal of intelligent & robotic systems*, vol. 61, no. 1-4, pp. 287–299, 2011.

[79] A. Martinelli, “Vision and imu data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination,” *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 44–60, 2012.

[80] “Open source computer vision library,” OpenCV, 2018. [Online]. Available: https://opencv.org/ [Accessed: 2018-07-19].

[81] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” *International journal of computer vision*, vol. 1, no. 4, pp. 321–331, 1988.

# Appendix A: Detailed installation of cameras, IMU and GPS antennas

In Figure A1, a detailed installation of the stereo camera setup is presented, including their base line distance as well as their inclination (pitch) angle with respect to the world in static conditions. No roll angle was present at the time of mounting the setup (i.e. 0°), and the rotation around the world Z-axis (yaw) with respect to the ship was not measured.

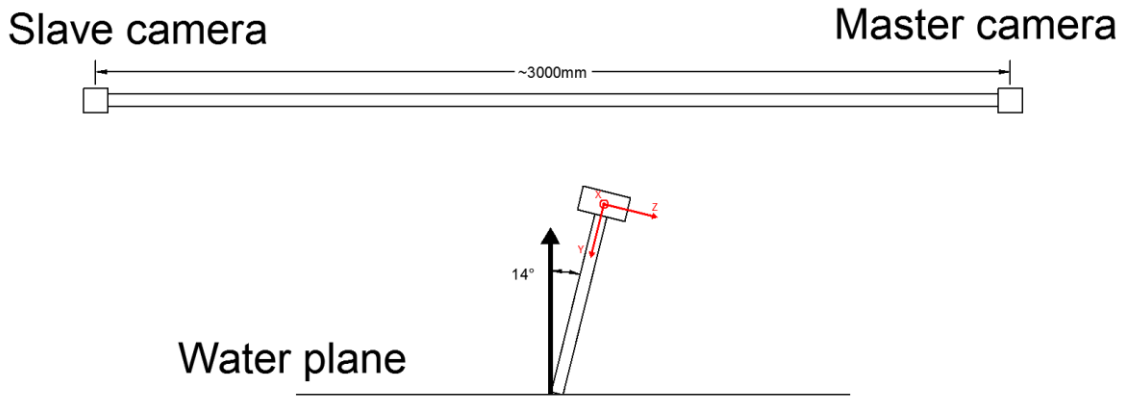


Figure A1. Baseline distance for the stereo camera setup (top) and inclination angle (i.e. pitch angle) with respect to the world Z-axis (bottom image). Roll and yaw angles are assumed to be 0°.

In Figure A2, the mounting position of the IMU inside the crow's nest is presented. In absolute coordinates (i.e. world coordinates), the IMU presents a roll of 6.8 degrees around the X-axis, and a pitch of -184 degrees around the Y-axis. Once again, the rotation around the Z-axis (yaw) is assumed to be 0°.


Figure A2. Mounting position of the IMU inside the crow's nest.

Lastly, Figure A3 presents the reader with the mounting position of the GPS1, which is located at the back of the crow's nest and roughly in the middle of the metallic plate.

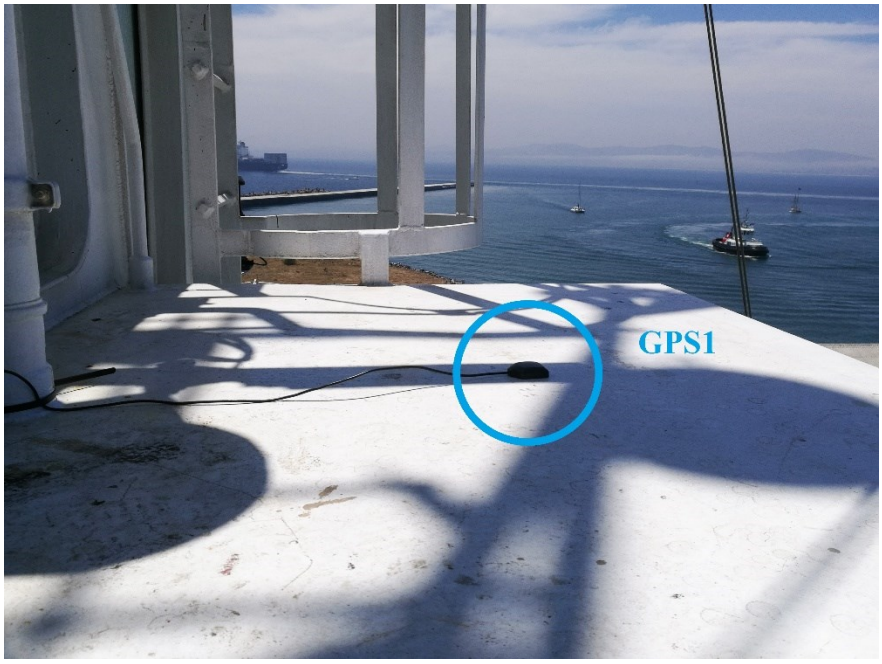


Figure A3. Mounting position of the GPS1 antenna on the S.A. Agulhas II ship.

Figure A4 presents the mounting position for the GPS2 antenna, on the side rail along the crow's nest. The exact position for both antennas may be extracted with the help of the ship's technical drawings.

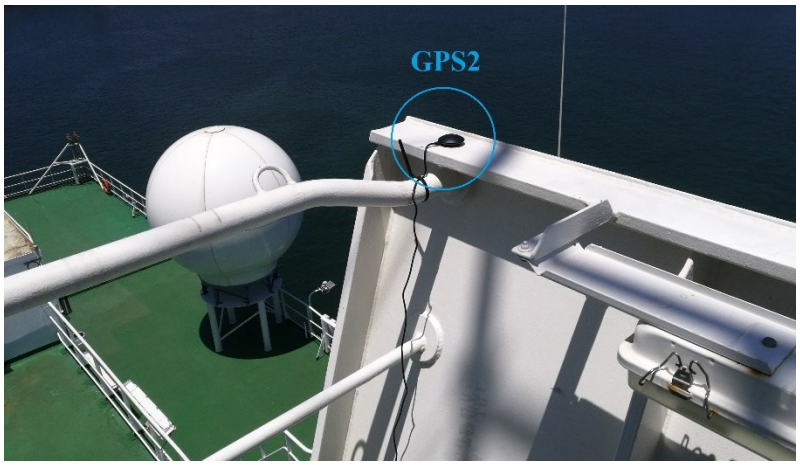


Figure A4. Mounting position of the GPS2 antenna on the S.A. Agulhas II ship.

# Appendix B: Excerpt from [60]

TABLE 1. Representative all-wave solar albedos of surface types in the East Antarctic sea ice zone in spring and summer. Values in bold are derived from measurements; all others were interpolated or extrapolated using Fig. 5. The seasons are indicated by SON and DJF.

| Ice type | Ice thickness (cm) | No snow | | Thin snow (<3 cm) | | | | Thick snow (>3 cm) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Clear | | Cloudy | | Clear | | Cloudy | |
| | | Clear | Cloudy | SON | DJF | SON | DJF | SON | DJF | SON | DJF |
| Open water | 0 | **0.07** | **0.07** | — | — | — | — | — | — | — | — |
| Grease | <1 | **0.09** | **0.09** | — | — | — | — | — | — | — | — |
| Nilas | <10 | **0.14** | **0.16** | **0.42** | **0.39** | **0.45** | **0.42** | — | — | — | — |
| Young grey ice | 10–15 | **0.25** | **0.27** | 0.55 | 0.51 | 0.59 | 0.56 | 0.72 | 0.67 | 0.76 | 0.72 |
| Young grey-white ice | 15–30 | 0.32 | 0.34 | 0.64 | 0.59 | 0.68 | 0.64 | 0.76 | 0.70 | 0.81 | 0.76 |
| First-year ice <0.7 m | 30–70 | 0.41 | 0.45 | **0.74** | **0.69** | **0.79** | **0.74** | 0.81 | 0.75 | 0.87 | 0.82 |
| First-year ice >0.7 m | >70 | **0.49** | **0.54** | **0.81** | **0.75** | **0.87** | **0.82** | **0.81** | **0.75** | **0.87** | **0.82** |

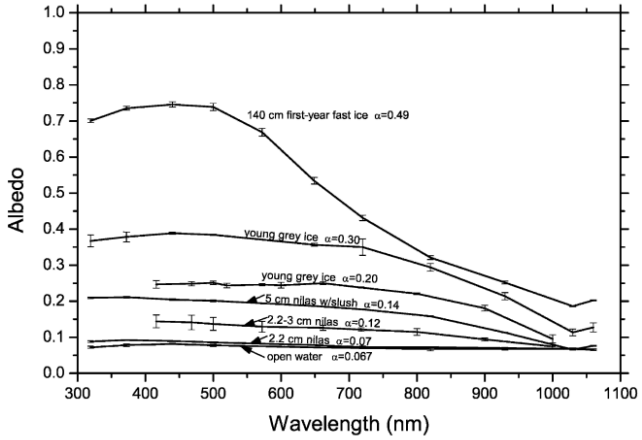


FIG. 1. Spectral albedos of snow-free ice and open water. Measurements from the 1996 voyage begin at 320 nm; those from the 1988 voyage begin at 420 nm. Broadband solar albedo $\alpha$ is also given. Ice thickness $Z_i$ is given, except for two ice types that were observed only from a helicopter. The curve for 2.2–3-cm nilas was included in Fig. 11 of ABW but was mislabeled there as "3–4 cm."

# Appendix C: Additional results from Experiment C

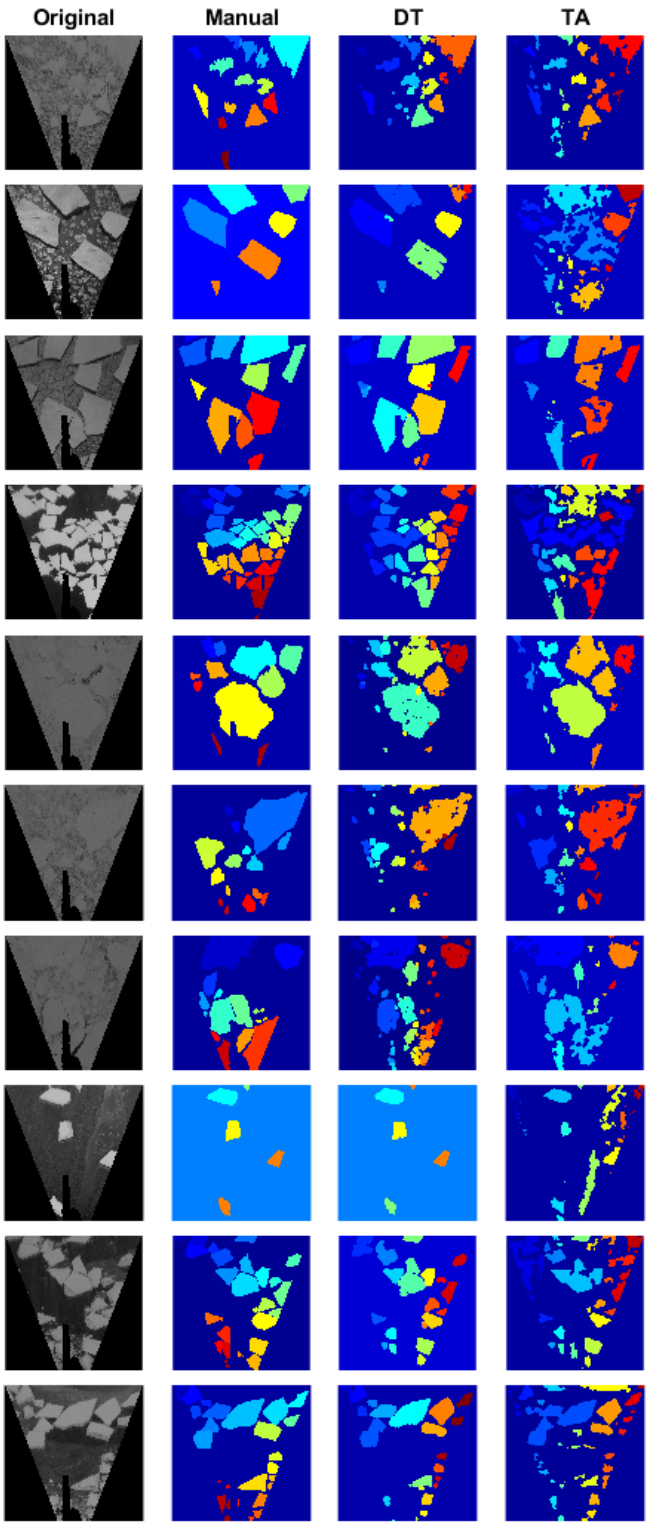


Figure D1. Colour comparison of identified sea-ice floes through three methods. Floe and backgrounds colours are random and serve the mere purpose of identifying two or more separate floes. Each row corresponds to the index number of the same image through the experiment.

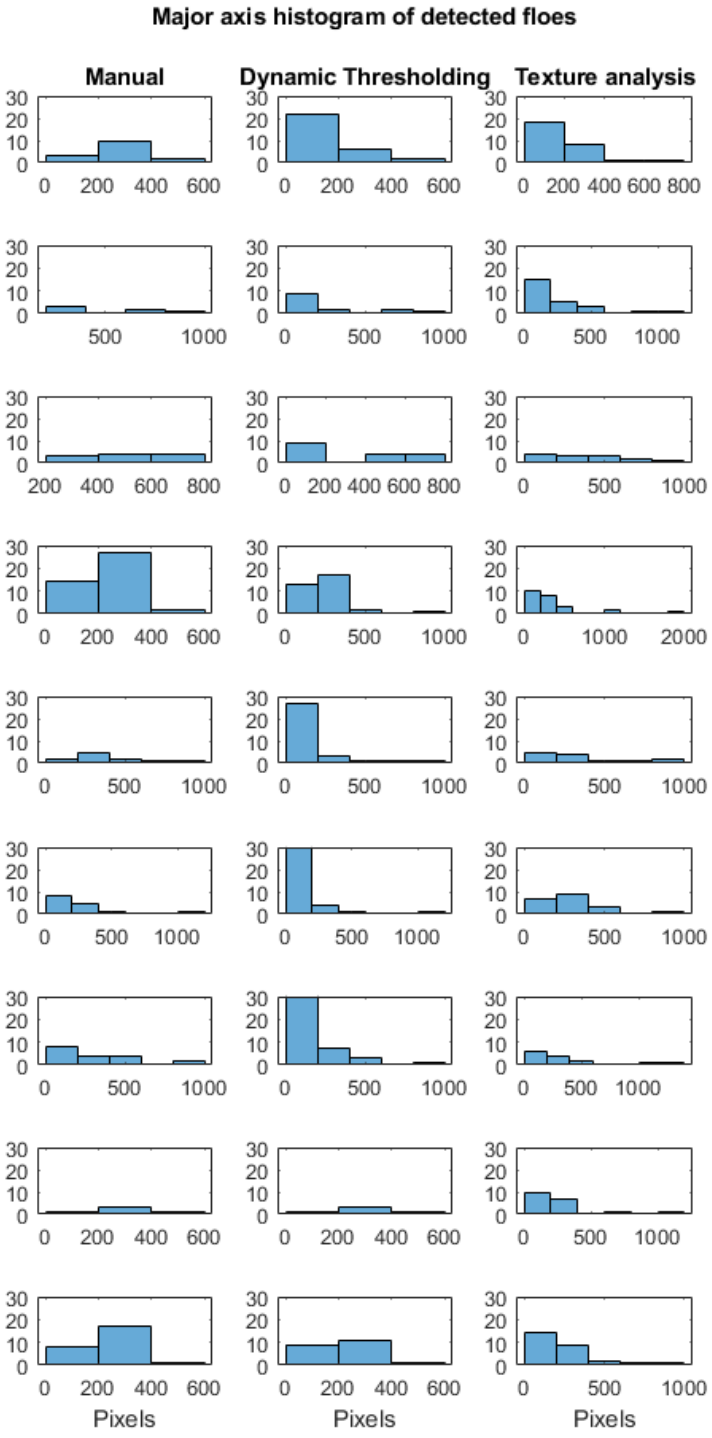


Figure D2. Histograms of the major axes of the detected floes in the complete test set of images, using a bin width of 200 pixels. Each row corresponds to the index number of the same image through the experiment.

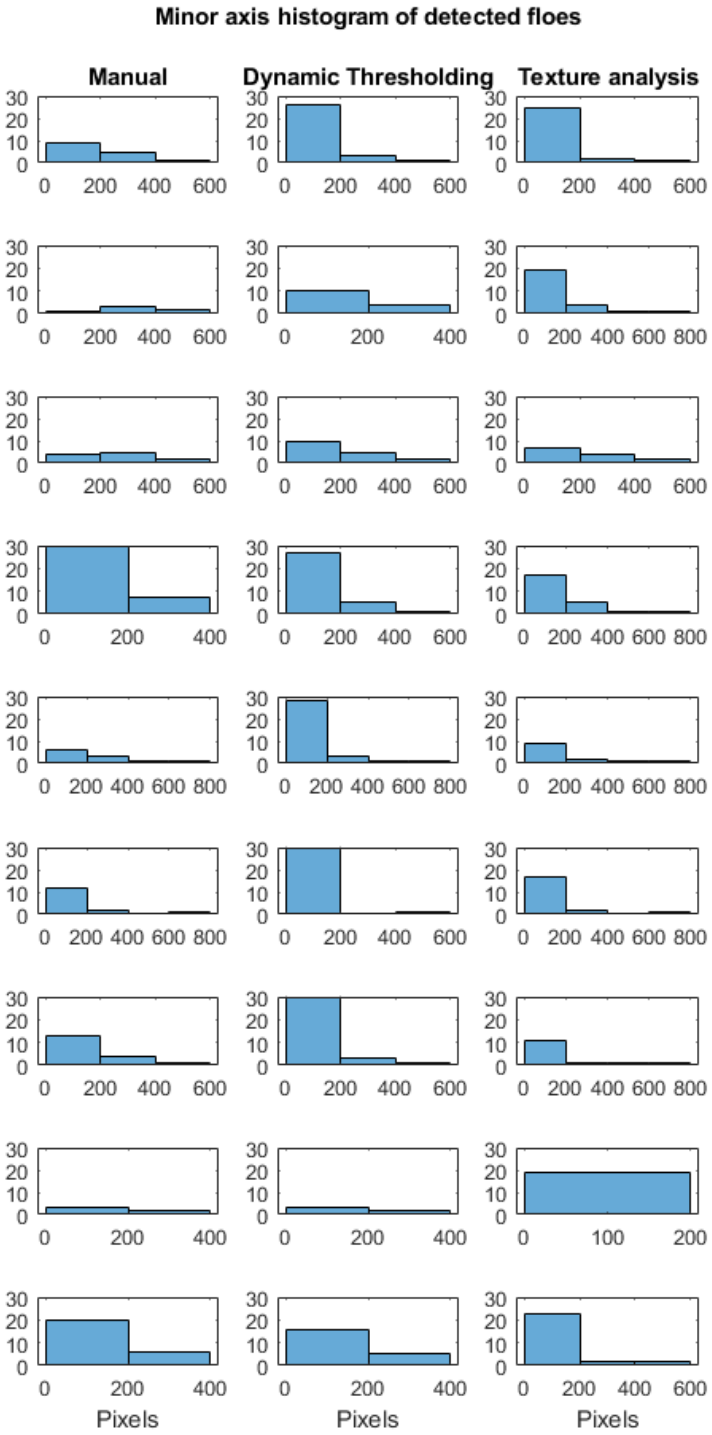


Figure D3. Histograms of the minor axes of the detected floes in the complete test set of images, using a bin width of 200 pixels. Each row corresponds to the index number of the same image through the experiment.

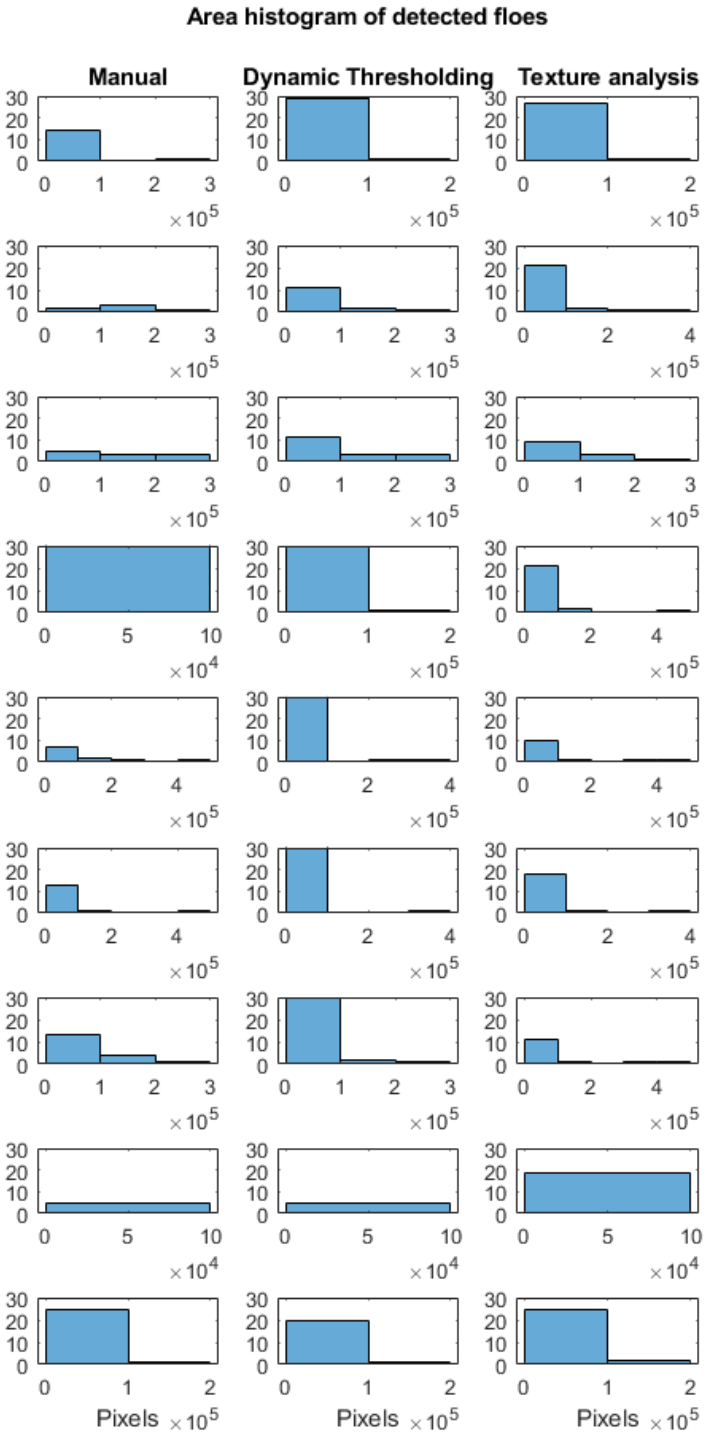


Figure D4. Histograms of the areas of the detected floes in the complete test set of images, using a bin width of 100000 pixels. Each row corresponds to the index number of the same image through the experiment.