Stereo System Design

For Ship-Based Acquisition in Antarctica

Presented by:
Kelsey Kaplan

Prepared for:
A. Mishra
Dept. of Electrical and Electronics Engineering
University of Cape Town

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfillment of the academic requirements for a Bachelor of Science degree in Electrical and Computer Engineering

February 4, 2020

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Signature: .....
Kelsey Kaplan

Date: 11 October 2019 .....

Acknowledgments

I would firstly like to thank my supervisor, Amit Mishra, for his guidance, support and availability for the duration of this project. His consistent enthusiasm and interest in the project's development were invaluable.

I would also like to thank Robyn Verrinder for the detailed feedback from the July expedition and for helping with the installation of the system on the ship, as well as the time and advice given to this project, approaching none of my questions as too trivial.

I am also grateful to the Oceanography Department, and in particular Marcello Vichi, for allowing me to take on this project and accompany them on the expedition in October.

And finally, I would like to thank my family, without whom, undertaking this degree would have been impossible. And a particular thank you to my brother, for revising my report, as well as the years of unwavering support, successful attempts at giving advice on subjects far from his own field and for helping me retain a good sense of (in)sanity in the final months.

This research has received funding from the DSI/NRF through the South African National Antarctic Programme.

Abstract

Multiple expeditions to Antarctica are undertaken each year to study the dynamic nature of sea ice. A stereo acquisition system for recording sea ice data, consisting of two cameras and an IMU, was deployed on an expedition in July 2019, but required major improvements as a result of system failures. The challenges of designing a ship-based stereo acquisition system for use on research vessels include multi-sensor calibration and synchronisation as well as the continuous acquisition of data for extended periods. Antarctic conditions introduce their own set of problems, requiring that equipment be capable of withstanding sub-zero temperatures, strong winds and ship vibrations. Accounting for the movement of the ship in images acquired is another important consideration. Use of an IMU for image correction poses potential problems as its accuracy at the poles is questionable due to its reliance on the compass and it may undergo significant error drift as it is unable to be recalibrated on the ship. This provided an opportunity to develop an alternative method of image correction, using the horizon in each image to estimate the ship's relative orientation.

An acquisition system was developed and thoroughly tested, and was ultimately successful. However, it has not yet been tested in ship-based or Antarctic conditions. Its performance in this environment therefore, remains to be seen during the October expedition. The horizon detection method has shown promising results and can be implemented as an alternative method of image correction and to assess IMU accuracy on polar expeditions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background to the Study

Every year, the UCT Oceanography Department embarks on multiple expeditions to Antarctica aboard the S.A. Agulhas II, where they collect data and perform numerous studies. Included in their research are the fields of sea ice and wave observation, which require an integrated camera and inertial measurement unit (IMU) system to be set up to record wave and ice data, as well as the ship's motion, during daylight hours.

Figure 1.1: An image of the S.A. Agulhas II [1]

Stereo vision systems, in which multiple images of the same scene are captured to enable 3D reconstructions of the subject matter, are incredibly useful in these studies — particularly for the observation of the dynamic relationship between sea ice and waves.

The acquisition of multiple images allows for accurate depth and elevation information to be derived, without the use of more expensive types of sensors, such as infrared scanners.

A stereo acquisition system was designed and deployed for the expedition in July 2019. During the expedition however, numerous problems were identified. This provided an opportunity to redevelop the system, to address identified problems and ensure that it would be fully functional for the next expedition in October.

1.2 Objectives of the Study

1.2.1 Problems to be Investigated

The complexities of designing a ship-based stereo acquisition system for use on research vessels in Antarctica centre on the problems associated with multi-sensor calibration and synchronisation as well as the continuous acquisition of data for extended periods of time. Accounting for the movement of the ship in images acquired is likewise an important consideration. The post-analysis of the image data often requires that images be corrected and aligned to a global coordinate system, in order to retrieve accurate depth and elevation. Additionally, Antarctic conditions introduce their own set of challenges, requiring that the equipment be capable of withstanding sub-zero temperatures, strong winds, turbulence and ship vibrations.

1.2.2 Purpose of the Study

The primary objective of this research project is to design and develop a fully operational and robust stereo acquisition system that can be tested in October and used on future expeditions.

Arguably the most expensive and time-consuming aspect of research in the Antarctic is the collection of data because of Antarctica's remoteness. This remoteness also means that there are often long periods between expeditions. Because it is unfeasible to make any major changes to the system's operation, both in terms of hardware and software, once the expedition has begun, it means that everything has to be prepared and well tested beforehand. Therefore, the design and particularly testing phases of this project are incredibly important in order to identify and solve any of the problems that may arise

on the ship.

1.3 Scope and Limitations

In the months prior to the start of this research project, I worked with the Oceanography Department to help set up the initial stereo system for the expedition undertaken in July. As a result of my prior work developing the system, it is important to clearly define the starting point of this project, and what work is considered out of scope as it was completed before the official project start date.

For the original system, I assisted with setting up the hardware components and developed the Matlab acquisition code. The equipment and assembly on the ship will primarily stay the same and is therefore considered to be out of scope. However, addressing a number of hardware issues identified in July falls within scope since they are associated with this additional phase of development. These specific issues are explicitly detailed in later sections. With regards to software, it was completely redeveloped during the project — as acquisition was migrated from Windows to Linux and written in Python. Additionally, no major testing of the acquisition system was done prior to the start of this project.

The field of stereo vision includes a specific focus on the processing steps that take place after acquisition, in which the 3D reconstructions are developed. In the early stages of the project, there was an intention to develop stereo algorithms for images of ice. However, due to the numerous difficulties experienced during the July expedition the project focus shifted back towards improving the system itself. The focal point of this project is therefore the acquisition step of stereo vision systems. This includes motion compensation techniques for image correction, which is also a precursor to the post-processing phase of stereo vision.

One of the limitations of the project is that there was no access to the equipment or the acquired data prior to the return of the July expedition, which was a month after the official start date of the project. Additionally, the research vessel departed for a subsequent expedition in September, only to return a few days before the project deadline, which prevented any opportunities for conducting onboard tests prior to the project's submission.

1.4 Plan of Development

The following report documents the design and development of the stereo acquisition system. The first section provides a literature review that covers relevant topics, and will contextualise the project within the greater field of study. Following this, the methodology chapter justifies the project definition and requirements as well as details the planning, development and analysis of the system The results chapter describes the testing protocols for the system, and includes a discussion on the results that were obtained. Subsequent to this, the conclusions chapter revisits the initial project requirements, assesses whether the system has met these criteria and provides some concluding remarks on the overall project. The final chapter, recommendations, concludes the project by providing final thoughts for future development and further work.

Chapter 2

Literature Review

The following chapter provides an overview of literature relevant to this research project. This includes some background on sea ice research in Antarctica, as well as the use of stereo vision systems within this context. An in-depth study on the post-acquisition phase of stereo vision was initially conducted, as in the early stages of the project stereo algorithms were intended to be the focus of research. However, due to the change of scope, this has been condensed in the following chapter and a more comprehensive review can be found in Appendix A. The chapter concludes with an overview of motion compensation techniques for images, specifically for ship-based acquisition.

2.1 Sea Ice Research in the Antarctic

2.1.1 Introduction

Found primarily in the polar regions, sea ice covers approximately 7% of the world's ocean surface and refers to any forms of ice that result from sea water freezing [2]. Research on sea ice is important in the fields of climatology, meteorology, oceanography, physics, maritime navigation and marine biology and serves as a strong indicator of the effects of climate change [2].

In recent years, research interest into sea ice has been on the rise, expanding the need for the development of methods for measuring sea ice dynamics [3]. Satellite data is available and can be used in a number of ways for sea ice observation, however, this

Figure 2.1: Images of sea ice from the July Expedition.

type of data has limited application because it lacks the ability to record the onsite dynamics and interaction of waves and ice, providing only information on sea ice extent and concentration. In contrast to this, ship-based observations contribute an extra wealth of information on sea ice dynamics that can only be ascertained from closer-range observations [4].

2.1.2 Observation Methods

UCT researchers embark on expeditions to Antarctica every year to study sea ice. They use a combination of visual observation and remote sensor data for analysing the sea ice environment. Human observations follow the standardized Antarctic Sea Ice Processes and Climate (ASPeCt) protocols, in which observers record hourly estimations on parameters such as ice floe concentration, size, distribution and type, based on purely visual cues [4].

This method of observation is unreliable as the results are vulnerable to subjectivity, ambiguity and uncertainties, especially in the case of unexperienced observers or bad weather conditions that make it impossible to decipher depth [4]. The use of post-processed digital images of sea ice as a supplementary means for observation has, therefore, been used extensively to obtain a more comprehensive and accurate estimation of the environment. Furthermore, human observations are limited in their ability to quantitatively read the full dynamics of the sea ice environment, situating the added value of image processing in quantifying parameters that are indiscernible to the human eye, such as the complex physical processes of surface wave propagation through sea ice [5].

Some sea ice characteristics, such as ice floe size, are strongly dictated by surface wave parameters like wavelength and amplitude. And the reverse is also true, that surface

wave kinematic behaviour is determined by the surface ice characteristics [5]. Access to information on these dynamic properties is, therefore, essential to any meaningful study of sea ice.

2.1.3 Motivation for Stereo Vision Systems

Observing the dynamics of the relationship between sea ice and surface waves is impossible without the aid of remote sensors. Wave buoys are frequently employed, but are limited to a certain threshold of high frequency waves and tend to underestimate maximum wave displacement [5]. In recent years, stereo vision has become a popular alternative for measuring these characteristics [5]. Aside from being relatively affordable when compared with other remote sensing equipment, another advantage of utilising stereo vision systems for observation is that they are non-intrusive on the environment [6].

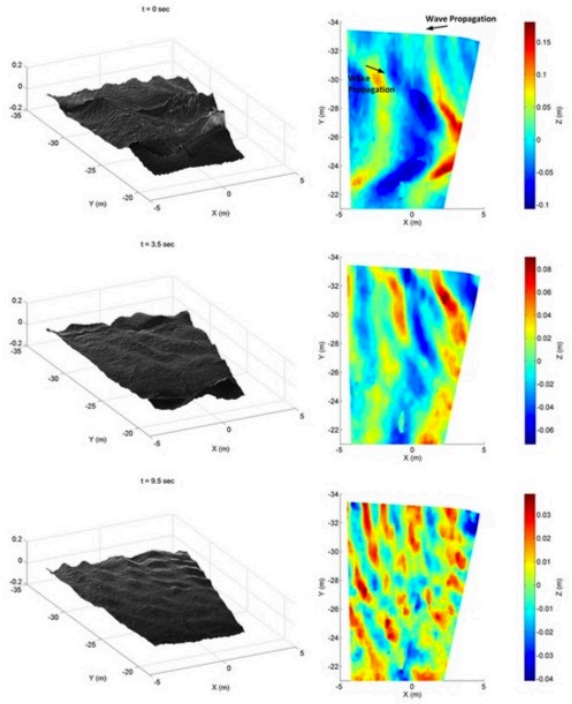


Figure 2.2: An image showing the result of reconstruction of the spatio-temporal wave field and the elevation information that can be derived using stereo vision [5].

The first experiments with ship-based stereo vision were done in 1939 by Schumacher and in 1960 Cote was the first to use stereo vision to measure sea topography for long ocean waves [6]. Stereo vision only became a widely available tool in the field of Oceanography however, with the improvements in hardware in the 1980's. Today, advances in modern digital image processing have further increased the potential for its application in sea ice research [7].

Stereo vision has the ability to provide accurate depth information of the captured scene and is able to reconstruct the spatio-temporal wave field with high accuracy (Fig. 2.2) [8]. Additionally by providing surface height, volumetric measurements of sea ice can be obtained [9].

The implementation itself is considered a complex task and therefore, in [8] they present an open-source stereo processing pipeline, WASS, for the 3D reconstruction of waves. This system automates the processes associated with stereo vision to produce dense point clouds. An example of elevation extracted from a stereo pair by the WASS pipeline can be seen in Fig. 2.3.

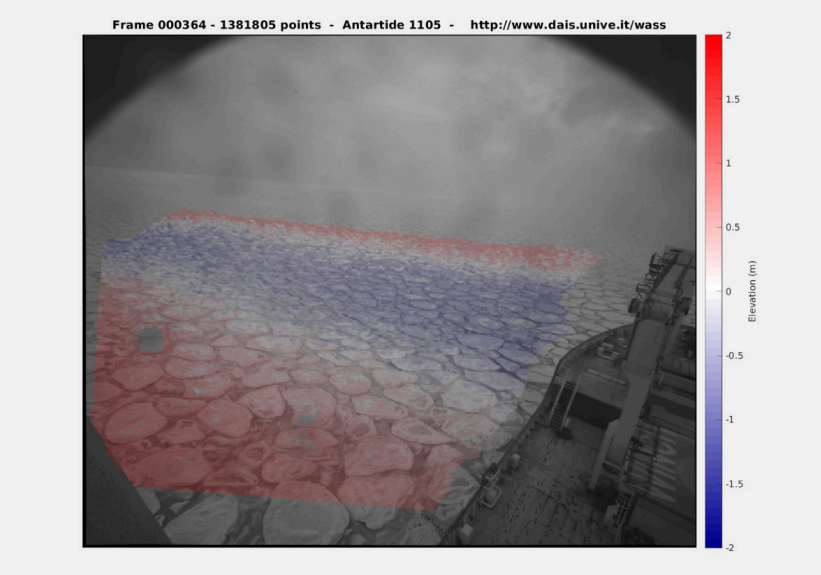


Figure 2.3: An image of sea ice with elevation information, processed by the WASS pipeline [8].

Previous studies have most commonly used stereo systems mounted on static platforms situated in the ocean, but this limits research as it constrains the range of conditions under which 3D wave characteristics can be ascertained [8]. This has resulted in a drive to employ the use of stereo systems on moving vessels, however, this complicates the system as the motion parameters of the vessel need to be recorded in order to map the data to a common reference system that is aligned with the direction of gravity [8]. There are a number of methods that can be used to compensate for the motion of the vessel, which include using other instruments, such as IMUs, the horizon line or the 3D sea surface field [8].

2.2 Stereo Vision

2.2.1 Introduction

Stereo vision is currently one of the most active research areas in Computer Vision [10] as a result of its value in creating machines with a visual ability matched to that of humans, whose visual system is stereoscopic [11]. The main objective of stereo vision is to recover 3D information, namely depth, from multiple 2D views of the same scene [11]. It has a wide range of applications which include tracking systems, navigation for autonomous vehicles, industrial automation, augmented reality and facial recognition [11].

Prior research has shown that human vision becomes monocular beyond six metres, which has led to disputes as to the applicability of stereo vision applications for long range depth estimates. However, controlled experiments have shown that in an artificial stereo system, accurate depth information can be obtained up to a range of 300 metres, provided that the system's baseline — the distance separating the two cameras — is sufficiently wide [12].

Figure 2.4: A diagram showing the stages of a stereo vision system.

Computationally, there are three problems that need to be addressed in stereo vision. These are rectification, correspondence and reconstruction [13]. Rectification involves calibrating the system, to find the stereo parameters that allow us to align the image planes [13]. Correspondence is the process of searching for pixels in the two images that represent the same physical point in 3D space [13]. While reconstruction solves the problem of how to associate each pixel pair with its corresponding depth in 3D space.

The calibration process of rectification will be discussed below, as it is the only stage relevant to this research project because of its implementation occurring prior to acquisition.

2.2.2 Calibration

The first step in rectification is to find the parameters of the stereo system through calibration. Calibration can be further subdivided into finding the intrinsic and extrinsic parameters. From these values we can compute a rectifying transformation that will align the image planes to the baseline [13], [14].

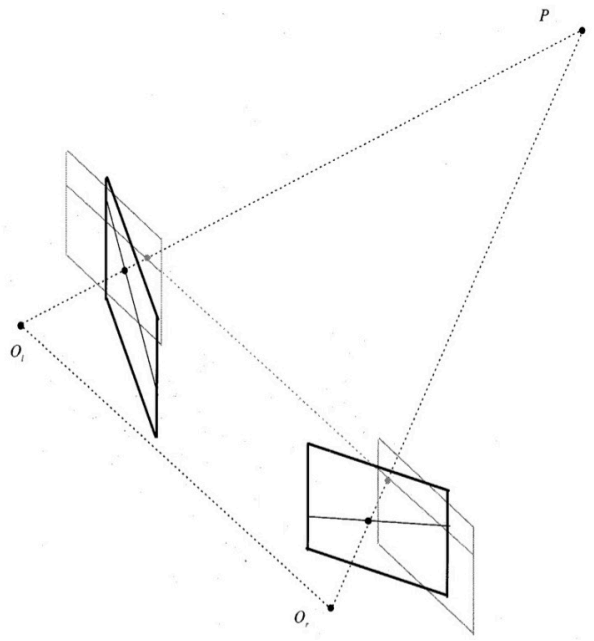


Figure 2.5: An image explaining the rectification step of stereo vision, which is to align the left and right image planes to the baseline [13].

Intrinsic parameters

The intrinsic parameters of a pinhole camera are defined by the camera's characteristics and include the distortion imposed by the lens, the focal length (f_x, f_y) and its optical centres (c_x, c_y).

Pinhole cameras have two types of distortion, tangential, a result of the misalignment between the lens and image plane; and radial, which is due to the curvature of the lens resulting in straight lines appearing curved [15]. These two types of distortion are accounted for by five parameters, known as the distortion coefficients.

$$(k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3) \tag{2.1}$$

OpenCV provides distortion correction functions that use the following formula for tangential distortion correction

$$x^{'} = x + [2p_1xy + p_2(r^2 + x^2)] \tag{2.2}$$
$$y^{'} = y + [p_1(r^2 + 2y^2) + 2p_2xy] \tag{2.3}$$

and for radial distortion correction

$$x^{'} = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{2.4}$$
$$y^{'} = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \tag{2.5}$$

where $(x^{'}, y^{'})$ are the corrected image pixels, (x, y) the original image pixels, $r = \sqrt{x^2 + y^2}$ and k_1, k_2, p_1, p_2 and k_3 the above-mentioned distortion coefficients [15].

The focal length and optical centres define the transformation relationship of a point between pixel and camera coordinates [13]. These parameters are stored in the camera matrix, K.

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

There are existing libraries, such as OpenCV's Camera Calibration Toolbox, for finding reliable estimates of distortion coefficients, focal lengths and optical centres. They use multiple images from different viewpoints of a simple pattern, typically a checkerboard, detect the corners and then calculate the parameters by averaging the results [15].

Extrinsic parameters

Extrinsic parameters describe the relative position and orientation of a camera reference frame with respect to some known world reference frame [13]. They are important for relating the positions of the two cameras in the stereo system, so that their image planes can be aligned [15].

The transformation from the image plane to the world frame is described by matrix G,

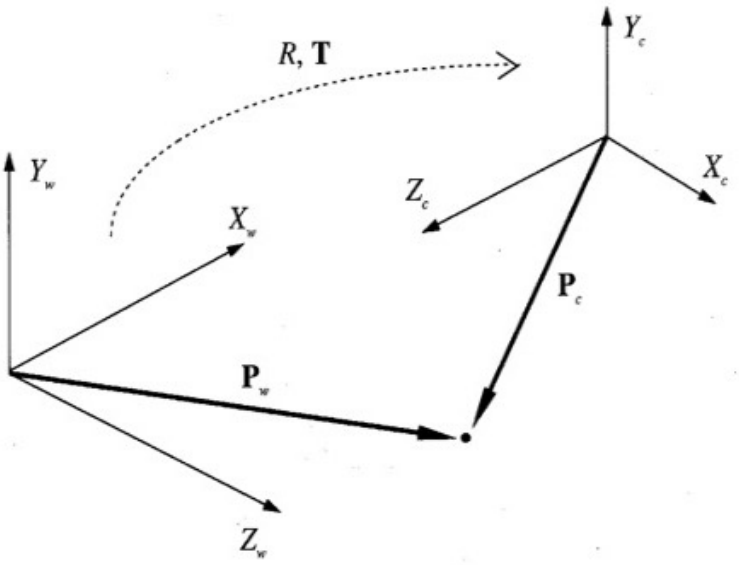


Figure 2.6: An image showing the transformation between the camera and world reference frame [13].

where R is a $3x3$ rotation matrix and t the translation

$$\mathbf{G} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{2.7}$$

In some cases, these parameters are unknown, and solving for them becomes integrated into the post-processing stage, using feature extraction methods to identify corresponding points which can then be used to find the transformation between camera frames [13].

The Camera Model

With the above parameters in mind, the transformation from a point on the image plane, $\mathbf{m} = [u, v]^T$, to its corresponding point in the world frame, $\mathbf{w} = [x, y, z]^T$, is done by the linear transformation matrix known as the perspective projection matrix $\mathbf{P}$, where

$$\mathbf{P} = \mathbf{K}(I|0)\mathbf{G} \tag{2.8}$$

Therefore the matrix $\mathbf{P}$, also known as the fundamental matrix in other literature [13], can be resolved from the known stereo parameters for a calibrated stereo set-up.

Taking all of the above into account, the transformation from a point in the image plane to the world frame, considering the intrinsic and extrinsic camera parameters, can be described as

$$\tilde{m} = \mathbf{P}\tilde{w} \tag{2.9}$$

$$\begin{bmatrix} u \\ v \\ s \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.10}$$

Note that the use of the tilde above m and w indicate that an extra point has been added to the vector. This additional point is included to create homogeneous coordinates, which essentially makes the point invariant to scaling [16]. More on homography is discussed in Section 2.3.3.

2.3 Orientation Estimation and Correction

2.3.1 Introduction

In order to accurately determine elevation from a series of stereo images, it is essential that depth is ascertained relative to a static world frame that is aligned with the gravity field [8]. However, as a result of the movement of the ship, the captured perspective of each image frame is not consistent. It is therefore necessary to apply a rectifying transform based on the ship's relative orientation.

In monocular image processing of sea ice, it is very important to correct for the perspective distortion that arises from the angle of the cameras with respect to the horizontal sea surface, so that accurate relative ice floe size and concentration can be determined [2]. This process of distortion correction is known as orthorectification, in which a geometric transformation is applied to recover the true shape of the ice floes. For stereo imaging, we are not necessarily interested in this type of rectification, but similar transformation principles to correct images for ship motion are applied.

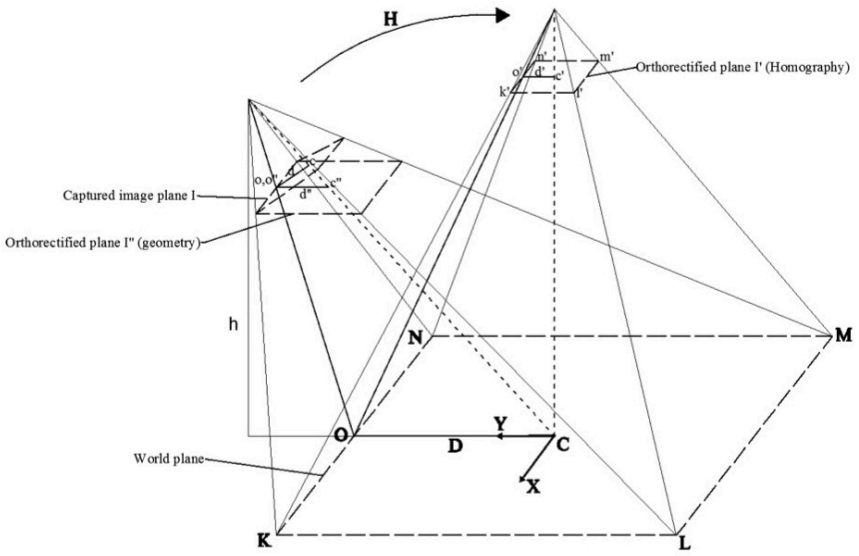


Figure 2.7: An image expressing the homographic transform applied to an image frame to perform orthorectification [2].

2.3.2 Estimation

Orientation estimation, also referred to as 'ego-motion', is a common task in any robotic navigation system, and can be carried out using a multitude of techniques [17]. Because of the equipment available for the purposes of this project, only orientation estimations using (1) IMUs and (2) visual odometry will be reviewed.

IMUs

An IMU is capable of providing both position and orientation information, and is therefore extremely useful in onboard navigation systems [18]. IMUs generally give triaxial measurements from a gyroscope, accelerometer and magnetometer [18]. By integrating the gyroscope readings, giving angular velocity, one can obtain orientation. Double integration of the accelerometer measurement, after first subtracting gravity, provides position information. And the magnetometer measurements indicate the direction of the magnetic field [18].

IMUs generally have a high sampling rate and their measurements are initially very precise, however, their accuracy deteriorates over time [18]. This is problematic on a ship, where recalibrating the IMU is impossible due to constant wave motion. This, coupled with the fact that IMUs may be inaccurate at the poles due to their strong reliance on the compass for data acquisition [19], raises concern over using IMUs for accurate image correction.

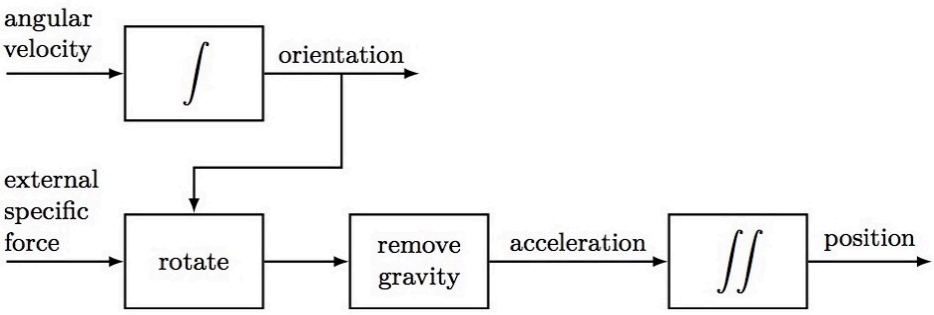


Figure 2.8: A block diagram for an IMU [19].

Visual Odometry

An alternative method for orientation estimation is to use visual odometry techniques. The roll and pitch of the ship relative to the world frame can be determined using the position of the horizon in the image, since the camera coordinate system is fixed relative to the ship. An orientation estimation algorithm would consist of two parts, firstly detecting the horizon and secondly computing the roll and pitch [17].

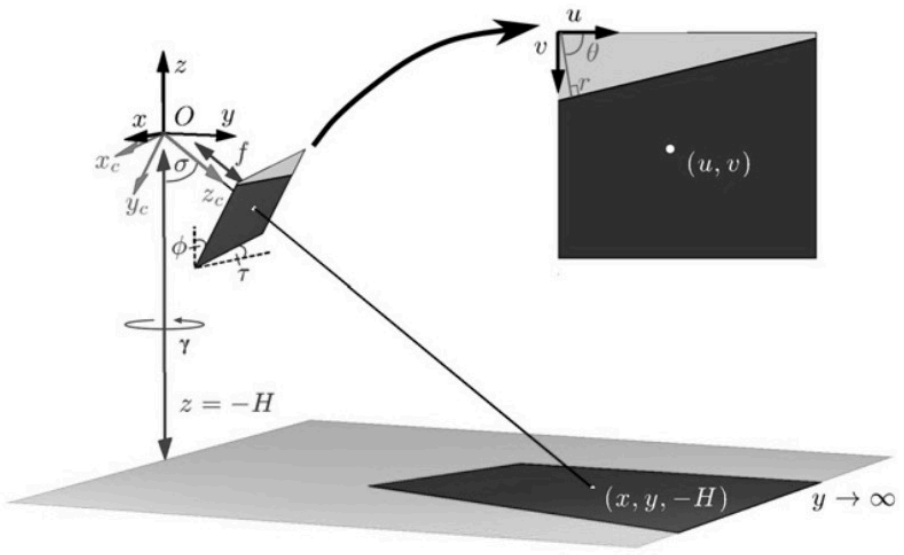


Figure 2.9: A diagram showing the process and calculations of horizon detection for camera orientation estimations [20]. The horizon line is parameterised by polar coordinates.

Horizon detection algorithms can be broadly divided into segmentation and classification, 2D edge detection with Hough Transform and 1D edge detection with least squares algorithms [17]. In [20], they found that first filtering the image using a Canny Edge Detector, which utilises gradient direction information, performed better than other edge filters, to which they then applied the Hough Transform. The advantage of the Hough

Transform is that it parameterizes the found horizon line in polar coordinates (r, θ), which can then be used in the computation of orientation [20].

To compute orientation, or more specifically the pitch (ϕ) and roll (τ), we need to relate the equation of the horizon line to the current pitch and roll of the cameras [17]. In [20], they provide the following equation, which uses the polar coordinates obtained from the Hough Transform, as well the focal lengths (f_x, f_y) and optical centres (c_x, c_y) obtained in the camera calibration procedure.

$$\tau = tan^{-1}[\frac{-f_x}{f_y tan(\theta)}] \tag{2.11}$$

$$\phi = \frac{\pi}{2} - tan^{-1}[\frac{f_x sin(\tau)cos(\theta) - f_y cos(\tau)sin(\theta)}{r - c_x cos(\theta) - c_y sin(\theta)}] \tag{2.12}$$

The benefit of visual odometry based estimation, other than not having to deal with IMU error drift, is twofold. Firstly, acquired IMU and image data does not need to be synchronized, a process prone to errors due to the uncertainties that arise in recorded acquisition times and actual acquisition times of both the IMU and cameras. And secondly, the sensors do not need to be calibrated in terms of position, as the IMU is generally aligned with the heading of the ship and the cameras not always directed along this axis, meaning that IMU readings used to correct for roll and pitch have to be calculated using some linear combination of two axes.

However, the visual odometry method is accompanied by its own challenges. The effects of weather and lighting conditions on image quality, as well as the presence of artifacts such as ships, icebergs and landmasses, may make identifying the horizon line impossible. The presence of waves may also distort the horizon line, making it a non-static landmark or difficult to correctly detect. Furthermore, equations 2.11 and 2.12 used to compute roll and pitch assume that the horizon is infinitely far from the camera, although the distance to the actual horizon is finite [17], which can lead to estimation errors. Fig. 2.10 shows that the uncertainty in the detected polar coordinate value r decreases as the distance to the actual horizon increases.

The distance to the actual horizon can be approximated as

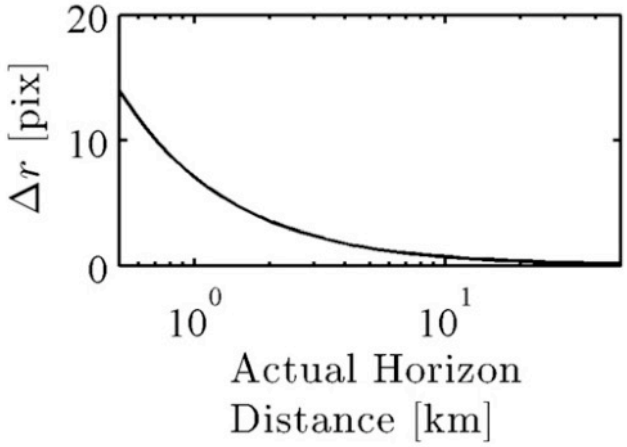


Figure 2.10: A plot showing the decreasing uncertainty in the location of the horizon (in terms of the polar coordinate value r in pixels) for finite horizon distances and a camera height of 10.7m [17].

$$D \approx \sqrt{2RH} \tag{2.13}$$

where R is the earth's radius, and H the height of the camera. As long as the distance to the actual horizon is large enough, the uncertainty can be approximated to be less than one pixel. However if, for example, there is land that obscures the horizon line, making it appear lower in the image, the errors in the estimation of r will increase [17]. This ultimately leads to errors in the computed roll and pitch.

2.3.3 Correction

Whichever method is used to estimate the ship's orientation, the formula applied for correction is the same. There are again multiple transformations one can use to perform image rectification. Homography will be explored for this project, as it draws on the same principles applied for orthorectification in monocular image processing and the rectification step of stereo vision.

A homography is a transformation from one projective plane to another. Projective planes of the same surface, obtained from a pinhole camera, can be related by the homography matrix **H** [16]. The homography matrix encompasses the rotation **R** from one plane to another, as well as the intrinsic parameters **K** of the camera.

$$H = KRK^{-1} \tag{2.14}$$

The $3x3$ rotation matrix $\mathbf{R}$ contains the information obtained about roll and pitch. The rotation about the z-axis, yaw, cannot be determined from the horizon line and is therefore inputted as 0.

$$R = R_z(0)R_y(\phi)R_x(\tau) \tag{2.15}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & -cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & -cos(\phi) \end{bmatrix} \begin{bmatrix} cos(\tau) & -sin(\tau) & 0 \\ sin(\tau) & cos(\tau) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.16}$$

In order to have a system that is calibrated to a particular world frame, we can correct for the image relative to the horizon captured when the ship is static. The statically captured horizon provides the reference values for roll and pitch. The correcting homography matrix $\mathbf{H}$ then becomes

$$H = (KR_{ref})(KR)^{-1} \tag{2.17}$$

where $\boldsymbol{R_{ref}}$ is the rotation matrix associated to the statically captured horizon in the reference image, and $\mathbf{R}$ is the rotation matrix of the target image. This way, all images will be corrected so that their horizon matches the horizon of the reference image.

Chapter 3

Methodology

3.1 Project Plan

The following section gives a clear definition of the problem, as well as the project requirements derived from the problem. The constraints and design approach are also described in detail.

3.1.1 Problem Definition

The objective of this project is to design and develop a stereo system for ship-based acquisition in the Antarctic. Whilst existing equipment will be used, the current system requires improvements and optimisation due to problems that were encountered on the July expedition.

The stereo system needs to be able to operate consistently for up to 24 hours at a time, and take into account the complexities associated with ship-based image acquisition. This requires that orientation correction be performed on images in order to compensate for the motion of the ship. Furthermore, Antarctic conditions necessitate that the cameras be fitted with heaters to allow them to operate in sub-zero temperatures and their mounting be stable enough to withstand strong winds, turbulence and ship vibrations. The system also needs to consider the basic problems associated with stereo acquisition, which include accurate sensor synchronisation and calibration.

3.1.2 Project Requirements

The project requirements were refined over the initial phase of research, to provide a feasible scope to the project and define in specific terms what it aims to achieve. An open ended requirement was also included in order to deal with any problems identified upon the return of the ship in July.

The project requires a stereo system for ship-based acquisition in the Antarctic. The system should:

- run on Linux and be coded in Python and use open source software — for accessibility and speed;
- be simple to operate and usable by non-engineers, as researchers from the Oceanography Department will be using the system;
- be capable of withstanding the turbulent weather conditions of the Antarctic;
- be capable of running continuously for up to 24 hours at a time without lag in acquisition or synchronisation issues;
- acquire synchronised image and IMU data for every second of acquisition;
- use a software trigger to synchronise image acquisition as the hardware trigger is incompatible with Linux;
- be able to account for the motion of the ship and correct images accordingly; and
- deal with any of the problems that occurred on the July expedition.

3.1.3 Analysis of Constraints

Aside from the time constraints placed on the project, the following were also identified:

- Access to the equipment and data was only possible a month into the project start date due to the ship being on an expedition in July.
- The project requirements were altered to prioritise the problems that arose in July. This meant that the project scope and definition could only be clarified a month after the start of the project, which imposed further time constraints.

- Testing of the stereo system could not be performed in realistic Antarctic or ship-based conditions.
- A mobile polar lab was available for system testing, but as the laptop cannot withstand the -20° degree temperature inside and the door had to be properly sealed it was impossible to connect the laptop to the system for testing. This means that the cold lab testing was limited to the camera heaters.
- On the expedition the IMU stopped functioning. Another IMU was assembled and used to collect data, however, it was uncalibrated and as a result no reliable orientation data was available from the July expedition.
- The design of the system was constrained by the need to use pre-existing equipment owned by the Oceanography department. This limited the methods with which the system could be implemented.

3.1.4 Design Approach

A V-Diagram approach was undertaken for this project as it is well suited for use with a clear problem definition. The V-Diagram approach also results in a thoroughly tested design, as although a clear distinction is made between the design and validation phases, they are inherently connected, as validation requirements are developed during the design phase.

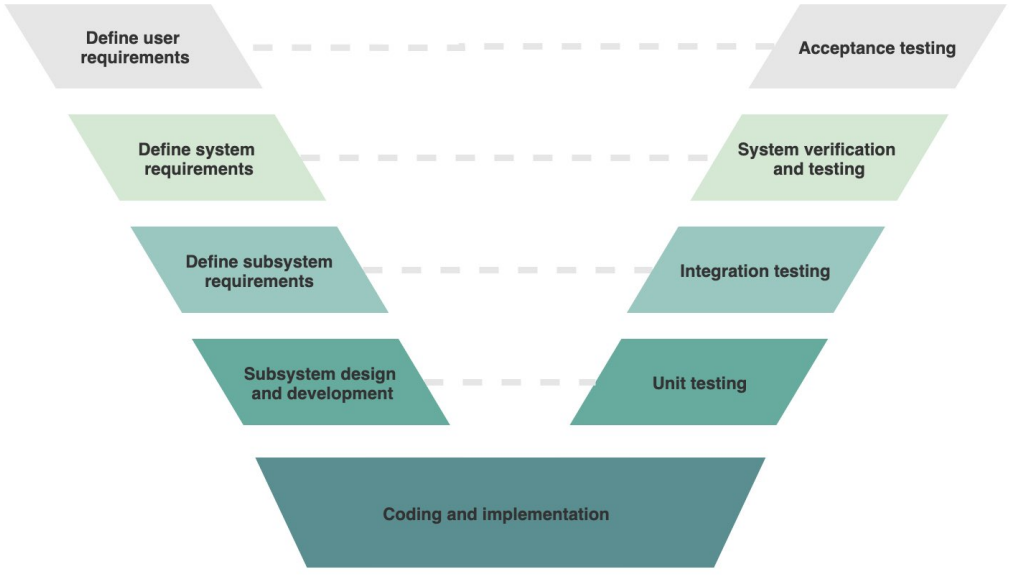


Figure 3.1: A V-Diagram detailing the project's design approach.

The initial design phase involved analysis of the project's primary objectives and context, which allowed the user requirements to be clearly defined. The system was then divided into subsystems, which provided an opportunity to define specific and focused technical specifications and testing procedures for each component of the system based on these requirements. The subsequent phase involved developing and testing each of the subsystems individually to ensure that they met their own objectives. Finally, once adequate performance was achieved, the subsystems were integrated in stages, which supported further integration and validation testing.

A work breakdown structure was drawn up to detail the work required for each of the system requirements (Fig 3.2).

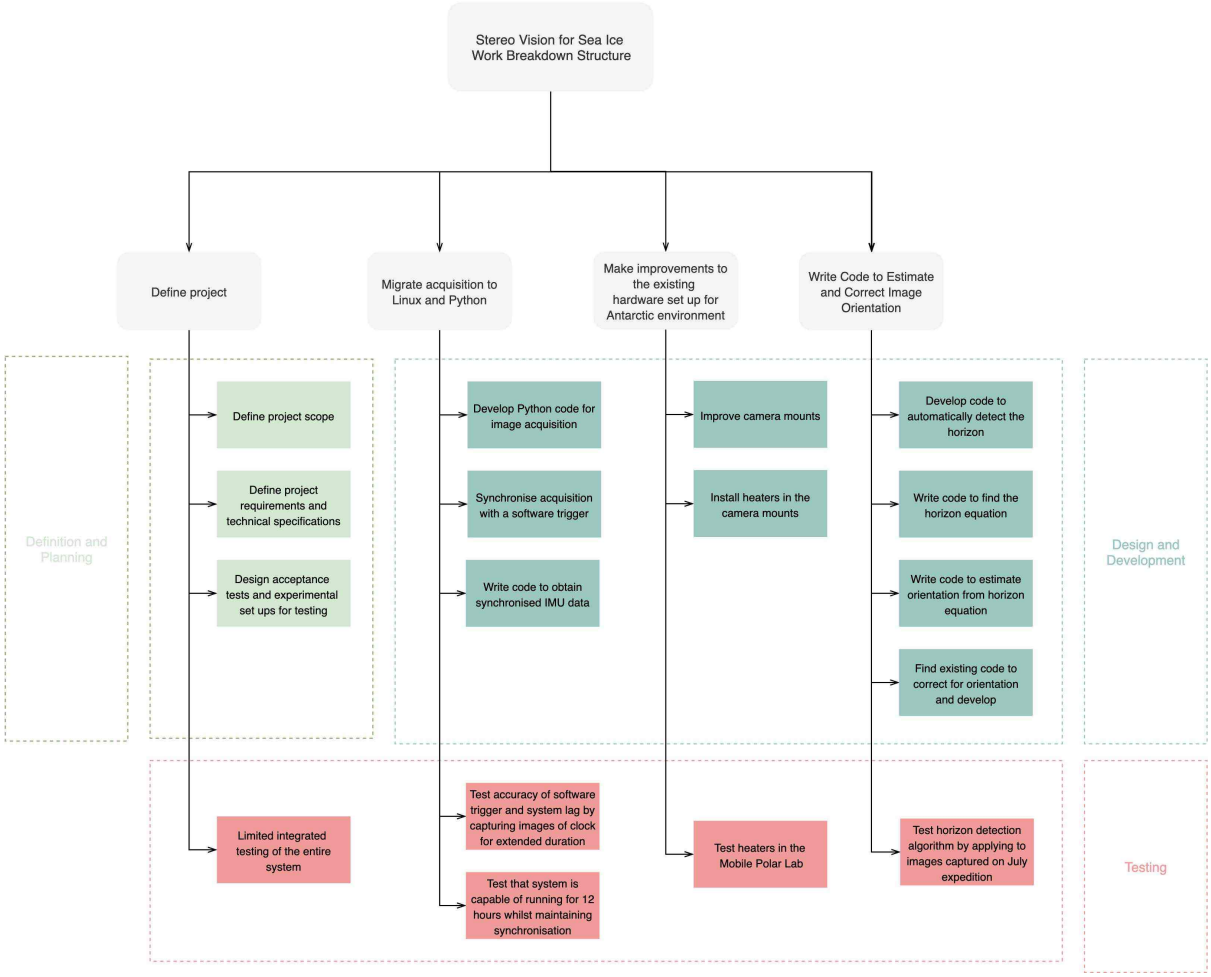


Figure 3.2: A work breakdown structure diagram detailing the tasks within the project requirements.

3.2 System Analysis

Before delving into the functional project requirements, it is important to have an understanding of the existing system and issues encountered during the July expedition in order to provide context for the requirements and technical specifications. The following section therefore gives an overview of both the current system and the problems that were encountered before detailing the technical specifications and testing protocols.

3.2.1 The Current System

3.2.1.1 Equipment and Set Up

Cameras

The characteristics and product description of the cameras are detailed in Table. 3.1 and the required camera accessories are shown in Table. 3.2. Accessories include a watertight and weatherproof housing, as well as camera mounts, heaters installed inside the housing and an external hardware trigger.

Model	Monochrome Industrial Camera
Shutter Type	2/3" CMOS Global Shutter
Resolution	2448 x 2048 (5 MP)
Pixel Bit Depth	8/12 bits
Pixel Size	3,45 um
Lens	Goyo 2/3" 5 mm F 1,8 Manual Iris, C-Mount 10 MP
Maximum Frame Rate	38 fps

Table 3.1: The camera characteristics.

The mounts were connected to custom built platforms that are tightly attached to the ship's railing on the Monkey bridge, as seen in Figs. 3.3 and 3.4.

Synchronizsation	Adept hardware trigger, Ethernet connection via laptop
Housing	Weatherproof HEB housing, 320 mm with sunshield
Mounting	WFWCA Parapet Bracket
Heaters	HEB heating element MNVKHEBH_0521, 230 VAC, 40 W

Table 3.2: The camera accessories.

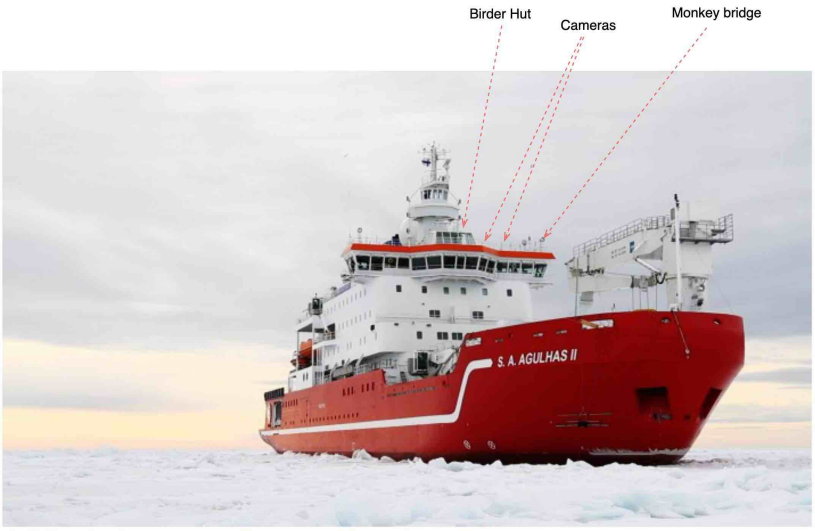


Figure 3.3: An image showing the location of the Birder hut and Monkey bridge on the ship, as well as where the cameras were located. Original image from [21].

Figure 3.4: An image showing the way in which the cameras were mounted on the ship.

IMU

The IMU is a 3-Space Sensor from Yost Labs. It is fitted with triaxial gyroscope, accelerometer and compass sensors, along with on-board filtering algorithms and output formats available as raw data, normalised data, filtered quaternions, Euler angles, rotation matrices and axis angles [19].

Part Number	TSS-USB-WT-S (Watertight Screw-down Sensor Unit)
Supply voltage	+5 v for USB, +3.3 v for RS232
Communication interfaces	USB 2.0, RS232 Asynchronous Serial
Serial baud rates	1 200 - 921 600 selectable, default: 115 200
Orientation outputs	absolute and relative quaternion, Euler angles, axis angle, rotation matrix, two vector
Other outputs	raw sensor data, normalized sensor data, calibrated sensor data, temperature
Temperature range	-40°C to 85°C

Table 3.3: The IMU characteristics.

Figure 3.5: An image of the Yost IMU [19].

Laptop

A Dell PC running Microsoft Windows 10 was used for data acquisition and logging. The cameras were connected by a 30 m cable to the laptop positioned in the Birder hut

(Fig. 3.3). Using a Matlab script, the system was set up to capture image pairs at a rate of 1 fps, triggered by the external hardware trigger connected to the laptop. Data was labelled by an acquisition timestamp and saved to an external hard drive.

Processor	8 x Intel Core i5
Speed	1,6 GHz
RAM	7,5 GB
Operating System	Partitioned with 64-bit Windows and Linux

Table 3.4: The laptop characteristics.

3.2.1.2 Problems Identified on the July Expedition

Heaters

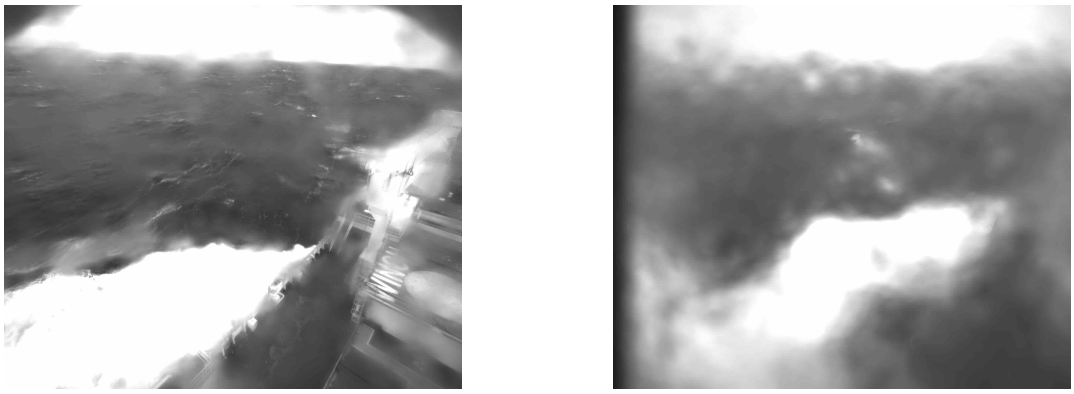


Figure 3.6: Images obtained in July showing the effect of a frosted housing window.

The purpose of the heaters is to prevent the housing windows from frosting over (Fig. 3.6). The majority of images captured did not exhibit this problem, however, a defrosting spray was also manually employed when frosting was noticed during acquisition. There is, therefore, uncertainty as to whether the heaters were ever operational. Experiments to assess the heaters therefore need to be performed prior to the next expedition.

Camera Mounts

The camera mounts (Fig. 3.4) extend the cameras approximately 20 cm above the ship's railing and have a rotating part that eventually loosened as a result of the constant

vibration of the ship. This, in combination with strong winds, caused one of the cameras to spin excessively at one point during the expedition. The camera was stabilized with cable ties for the remainder of the trip. To ensure this does not occur on future expeditions, an alternative mounting system is required.

Image Acquisition

With regards to acquisition, the system experienced severe lag, which resulted in images becoming unsynchronized and caused Matlab to crash occasionally. It was concluded that the Windows operating system combined with the Matlab application was incapable of running consistently, for extended periods of time, while processing large quantities of data. Acquisition will therefore be migrated to Linux and the acquisition code rewritten in Python, in order to extend the running time of the system and prevent the lag and synchronisation issues.

IMU

The Yost IMU was not operational on the ship for unknown reasons, however, potential sources of failure are the cable or the IMU itself. The IMU was not inspected on the expedition because this may have caused irreparable damage to its waterproof casing. The cause of failure is an important concern that needs to be investigated and addressed.

Aside from this, the Yost IMU's manual states that its accuracy in the polar regions is inaccurate because of its strong reliance on the compass [19]. Additionally, IMU errors accumulate over time, indicating that the inability to recalibrate the IMU when operating it for extended periods may lead to further errors. This provides motivation to develop an alternative method for motion compensation using visual odometry, which will also provide a way to assess IMU accuracy on the October expedition.

3.2.2 Subsystem Analysis

With a better understanding of the system's operation on the ship, we can now define the various subsystems that make up ship-based stereo systems. Fig. 3.7 shows each subsystem's place and output in the overall system. The last two subsystems, rectification and stereo correspondence, are included for completeness but are not within the scope of this project.

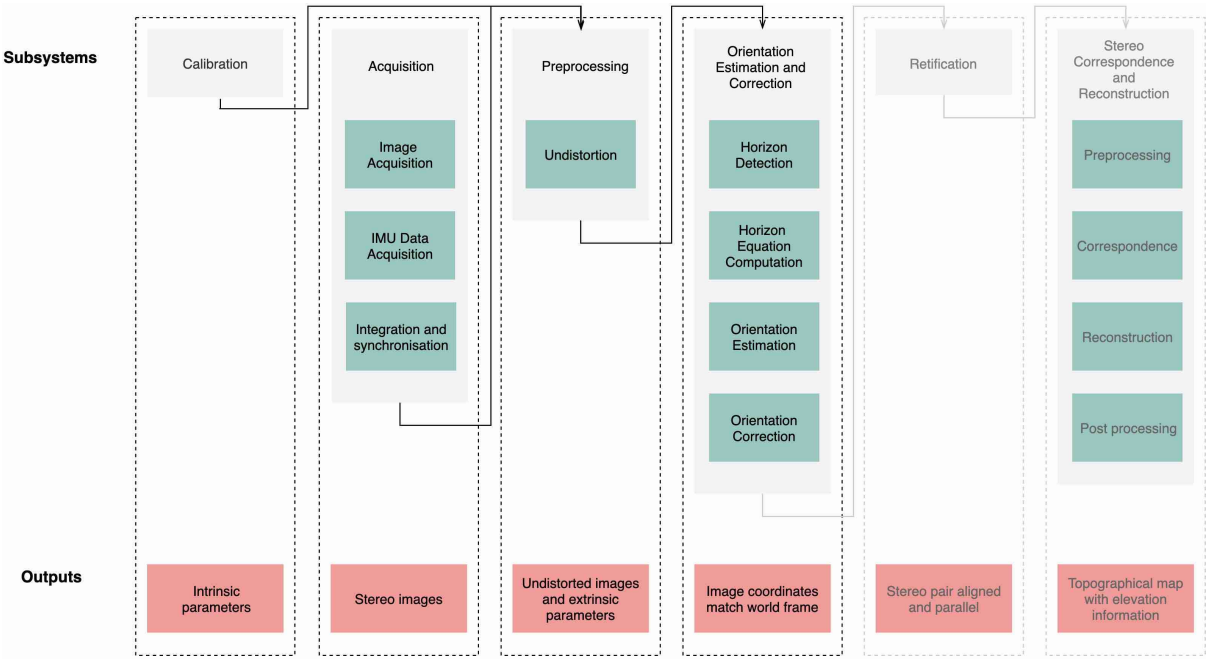


Figure 3.7: A diagram showing the subsystems and corresponding outputs of ship-based stereo vision systems.

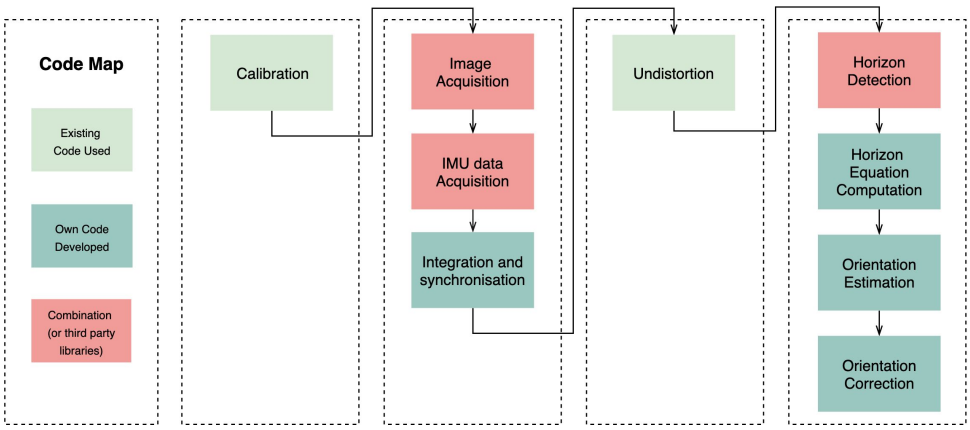


Figure 3.8: A diagram describing the required code for each subsystem and detailing whether it will use existing available code, newly developed code or a combination.

A combination of existing and developed code will be employed in the system and the breakdown of the code is shown in Fig. 3.8. This gives an indication of how much time is required for each of the various subsystems.

3.2.3 Functional Requirements

Based on an analysis of the user requirements and the problems identified on the July expedition, the following functional requirements were identified:

General System Requirements

RG-1: The system should run in Linux.

RG-2: The system should be simple to use.

RG-3: All code should be written in Python and use open source software.

Hardware Requirements

RH-1: A working IMU should be available for the October expedition.

RH-2: The cameras should be fitted with heaters that turn on when temperatures are less than 15°C in order to withstand the Antarctic cold.

RH-3: The camera mounts should be improved to withstand the turbulent weather conditions and ship vibrations.

Calibration Requirements

RC-1: The cameras should be calibrated prior to mounting them on the ship, to obtain the intrinsic parameters with a reprojection error less than 1 pixel.

RC-2: The IMU should be calibrated once aboard the ship when it is still docked, and be aligned with the ships heading for the duration of the expedition.

Acquisition Requirements

RA-1: The system should be able to run continuously for up to 24 hours.

RA-2: The system should have an average error rate of less than 0,01% over a 24-hour period (for every 100 s there is only one image pair or IMU data point not acquired).

RA-3: The cameras should be synchronised to take image pairs with a time difference of less than 1 ms.

RA-4: IMU data collected should be synchronised to each image pair, either during acquisition or in a post-processing step, with a time difference less than 100 ms.

RA-5: Image pairs and IMU data should be acquired every second, with an average interval error rate of less than 1 ms. This means there should be a time drift of less than 3.6 s every hour.

RA-6: Images and IMU data should be stored to an external hard drive and organised into subfolders detailing the acquisition start time. Corresponding image pairs and IMU data points should be labelled so that correspondence is simple.

Orientation Estimation and Correction Requirements

RO-1: The code should automatically detect the horizon of a series of images with an accuracy greater than 90%.

RO-2: The code should be able to use the detected horizon to estimate the ship's relative orientation.

RO-3: The code should correct images so that their horizon accurately lines up with the horizon of the reference image.

3.2.4 Technical Specifications and Acceptance Test Protocols

The technical specifications and corresponding test procedures detailed below were derived from the functional requirements. These exclude the general system requirements as they do not necessitate a more technical description or testing method.

	Specification	**Testing Protocol**
RH-1	A working IMU should be available for the October expedition	The IMU is able to be calibrated and data is consistently obtained when running acquisition code.
RH-2	Heaters should turn on in temperatures $< 15°C$	Test the heaters in the UCT Mobile Polar Lab for 24 hours, checking regularly that they are on.
RH-3	Camera mounts should be resistant to vibrations and stable in turbulent conditions and strong winds	This cannot be directly tested and so a combination of visual inspection and logic will be used to determine if the new mounts are satisfactory.

Table 3.5: The hardware specifications and testing protocols.

	Specification	**Testing Protocol**
RC-1	Cameras should be calibrated with a reprojection error < 1 pixel	The results of the calibration process should produce results that specify an error < 1 pixel
RC-2	The IMU should be calibrated once aboard the ship and be aligned to the heading of the ship for the duration of the journey	This cannot be fully tested, but the calibration accuracy and error drift over time will be assessed by running the IMU for 24 hours after first aligning it to a specific axis, moving it around during acquisition and placing it back in the same position. The data obtained should indicate equivalent orientations at the beginning and end of the experiment.

Table 3.6: The calibration specifications and testing protocols.

	Specification	**Testing Protocol**
RA-1	The system should be able to run continuously for up to 24 hours	Run the system overnight and check if acquisition stops at any point.
RA-2	The acquisition error rate should be $< 0,01\%$	Run the cameras and IMU for 24 hours. Using the counter that labels data, assess the number of data points that were not acquired.
RA-3	The camera synchronisation should be < 1 ms	Run the cameras for 24 hours, taking images of a digital clock to compare and analyse the exact acquisition time of corresponding pairs.
RA-4	The camera and IMU synchronisation should be within < 100 ms	Run the cameras and IMU for 24 hours and compare their timestamps to assess synchronisation precision.
RA-5	Image and IMU data should be acquired every 1 s with a time interval error < 1 ms	Run the cameras and IMU for 24 hours and assess the average time interval and overall time error drift using the timestamps of the acquired data.
RA-6	The data should be stored to an external hard drive and labelled so that correspondence is easy	Determine whether this criteria is met based on the organisation of data obtained in the 24 hour test.

Table 3.7: The acquisition specifications and testing protocols.

	Specification	**Testing Protocol**
RO-1	The code should automatically detect the horizon of a series of images with an accuracy $> 90\%$	Test the detection algorithm on three large sample datasets obtained in varying environmental conditions.
RO-2	The detected horizon should be able to estimate the ship's relative orientation	This can only be fully tested in October on the ship. For the purpose of this project, plotting the values obtained for orientation from a sequence of images and comparing left and right cameras to assess the accuracy of the estimation will be used as a preliminary test procedure.
RO-3	The corrected image's horizon should be identically positioned to the reference image horizon	Compare a set of corrected images to the reference image and find the average number of incorrect pixels (along the detected horizon line) and compare the newly computed orientation values.

Table 3.8: The orientation estimation and correction specifications and testing protocols.

3.3 Subsystem Development

The development of each of the subsystems, as dictated by the requirements, is detailed in the following section. Firstly, the more minor hardware issues that were identified in July are dealt with, following which details are provided on the data acquisition and sensor synchronisation development. Finally, a method for horizon detection and image correction is investigated.

3.3.1 Hardware Problems and Improvements

3.3.1.1 IMU

As previously mentioned, the IMU was not functional on the ship. The initial step taken to find the source of failure of the IMU was to obtain a new cable from Yost Labs. The IMU operated without issue with the replacement cable, which shows that the original cable was responsible for the IMU's failure. This was confirmed by testing the IMU using the 3-Space Sensor Suite Application from Yost Labs, which shows the IMU movement in real time and is also used for IMU calibration.

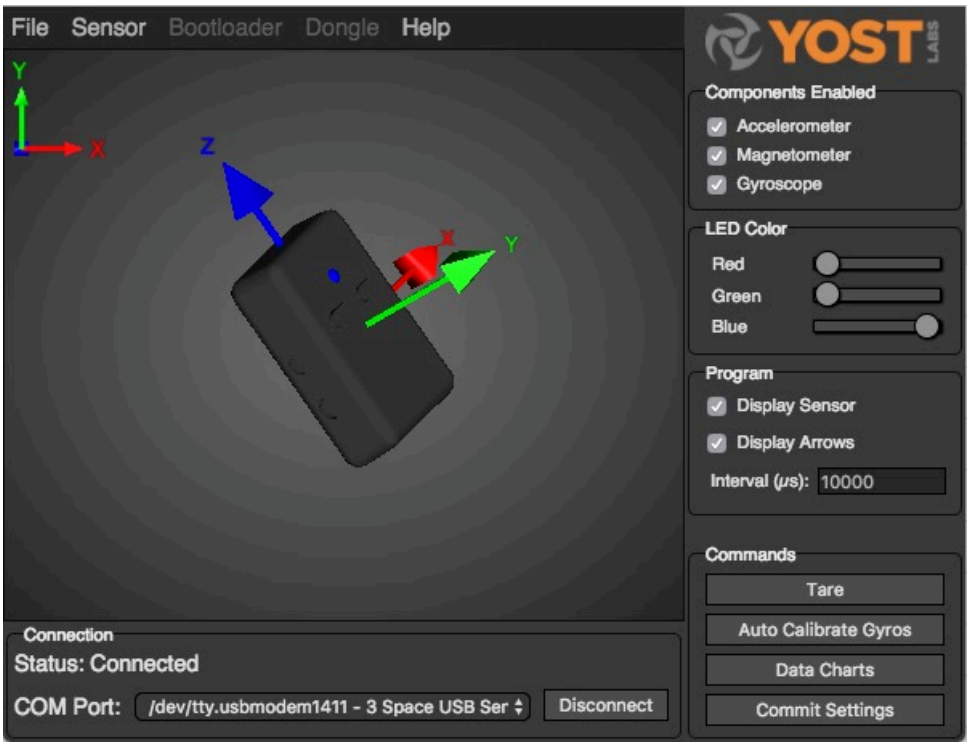


Figure 3.9: An image showing the 3-Space Sensor Suite Application.

Further development and testing of the IMU is detailed in Section. 3.3.3.

3.3.1.2 Camera Mounts

The mounts were redesigned and built with the help of the Mech-Eng Workshop. To lower the height at which the cameras were extended to, the circular centre part of the mounts was cut shorter, and then reconnected. All the nuts connecting the mounts to the wooden platforms were also refitted with nyloc nuts, a type of lock nut that has

extra resistance to turning. This is to ensure that the ship vibrations do not loosen the connections for the duration of the expedition.

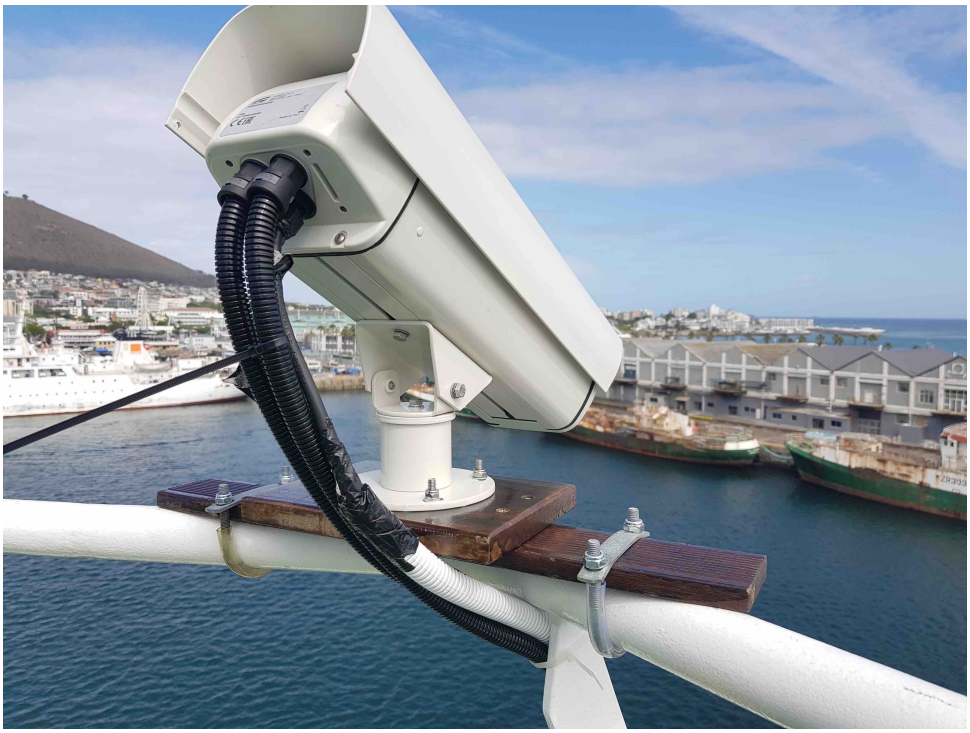


Figure 3.10: An image showing the new camera mounts. They are no longer extended as high, and all nuts were replaced with nyloc nuts.

3.3.1.3 Heaters

The heater components were procured from a company overseas and included only an instruction manual for assembly. No details were provided on the circuit itself. However, it is relatively simple and Fig. 3.11 shows the different elements of the circuit ascertained through some testing and investigation. The circuit is powered by 230 VAC and consists of a switching thermostat that should turn the heating element on when the temperature reaches approximately 15°C or less. The circuit is installed inside the camera housing, and heating element attached under the metal slide that the camera is mounted on.

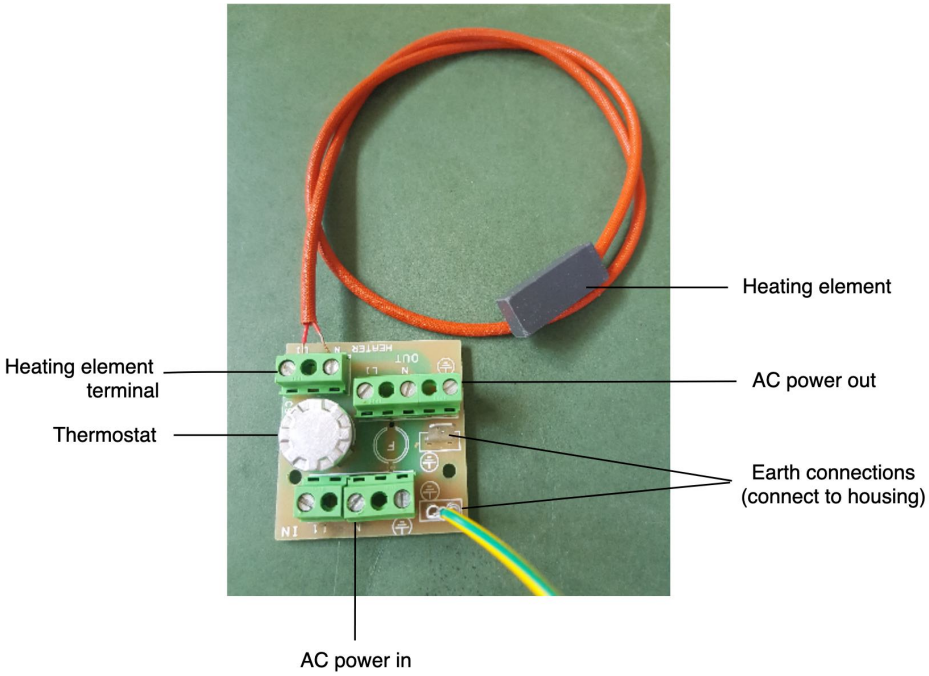


Figure 3.11: An image explaining the components of the heating circuit.

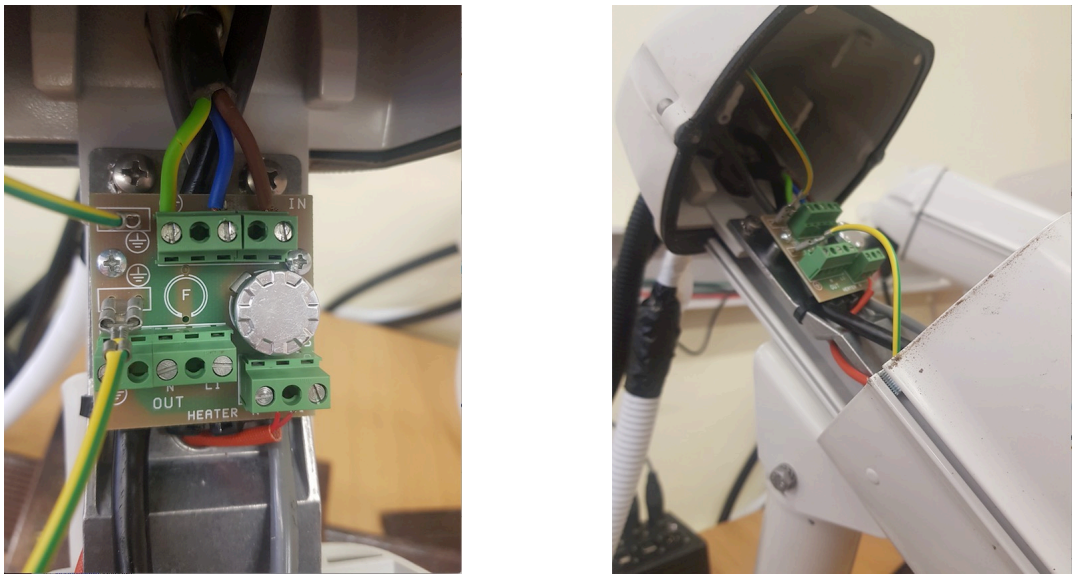


Figure 3.12: Images showing the heater component's placement in the camera housing.

After ensuring that the circuit was successfully receiving power, the heaters, installed in the camera housings, were first tested in the Mobile Polar Lab overnight. The results were inconclusive, as the researcher in the lab at the time reported that the camera mounts felt slightly warmer than the -20°C atmosphere, but not significantly so. The heaters, however, do not make close contact with the outer part of the housing, and so it was realised that simply feeling the housing to determine whether the heaters are on may not be sufficient.

The heaters were then separated from the housing and cameras, and taken into the lab individually for about 5 minutes to check for continuity between live and the heating element terminal using a multimeter. By this time, the lab's temperature had risen to approximately 3°C. There was no continuity detected, but the thermostat did not feel particularly cold.

Finally, the circuit was disconnected and put into a freezer for around 10 minutes. Upon removal, the thermostat felt icy and a continuity test showed that the thermostat was finally responding to the cold temperature by providing a connection from live to the heating element terminal. It was quickly reconnected, and as expected, the heating element very rapidly warmed up. It can therefore be concluded that the heaters are functioning. The lack of response in the cold lab, however, brought the threshold at which the heaters turn on into question.

In order to estimate the threshold at which the heaters turn on and provide evidence of their operation, a thermal imaging camera was used to assess the changing temperature of the thermostat and the heating element. Fig. 3.13 shows thermal images of the heating element's increasing temperature after removing the circuit from the freezer. As can be seen, it heats up very rapidly, increasing to 119°C in approximately five minutes.

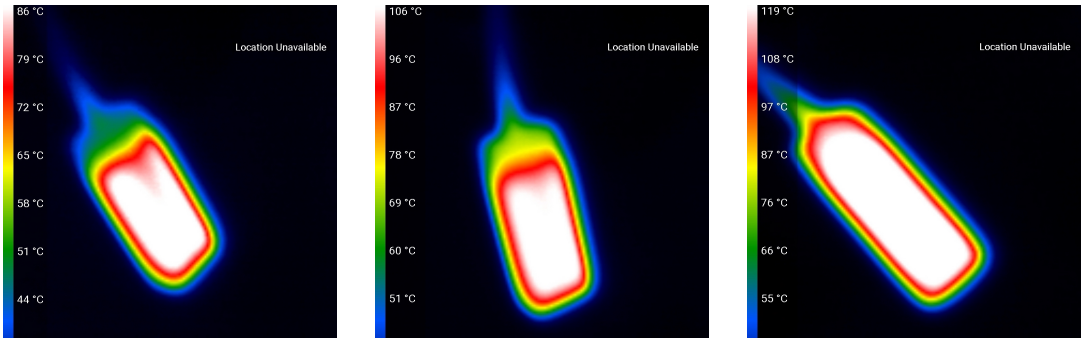


Figure 3.13: Images showing the temperature increase of the heating element as measured by the thermal camera. The heating element immediately after removal from the freezer (left), two minutes after removal from the freezer (centre) and five minutes after removal from the freezer (right).

Fig. 3.14 shows the thermal images captured of the circuit after removal from the freezer. It was found that the thermostat needs to be covered with tape to get accurate temperature readings, due to metals emitting infrared radiation inefficiently [23]. The results show that the circuit switched off when its temperature was approximately 21°C, which is much higher than the required 15°C, however, it is assumed that this method of testing is accompanied by some uncertainties and therefore not a completely accurate indication of the thermostat's threshold. Despite this, it can be concluded that the threshold at which the heaters turn on is sufficiently high and within the required 15°C

threshold.

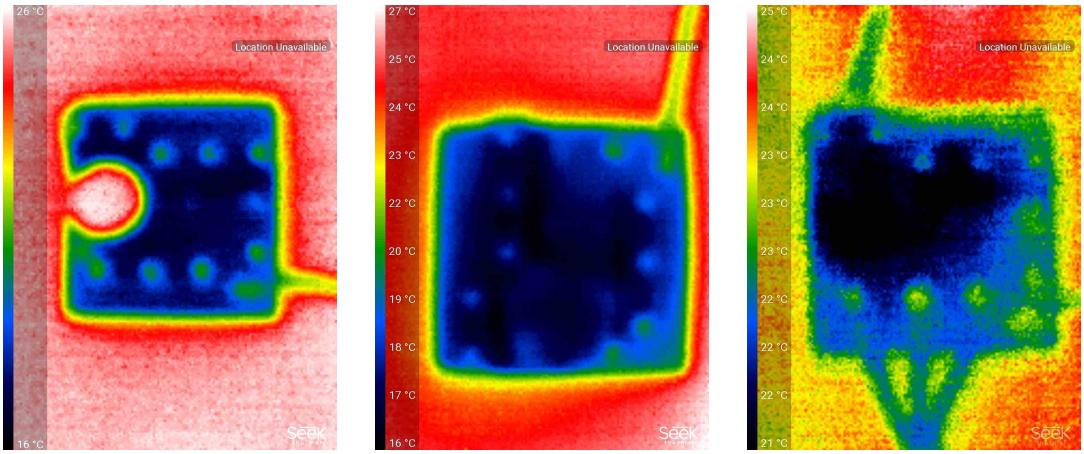


Figure 3.14: Images showing the temperature of the heater circuit as measured by the thermal camera. The circuit without tape covering the thermostat, resulting in an inaccurate temperature reading of the thermostat of 26°C (left), the circuit after removal from the freezer showing a temperature of approximately 17°C (centre) and the circuit moments after the heating element was turned off, showing a temperature of roughly 21°C (right).

3.3.2 Calibration

3.3.2.1 Cameras

The cameras need to be calibrated to obtain the intrinsic parameters used for undistortion, correction and stereo rectification. The simplest method of doing this is to use one of the many existing toolboxes, such as the Matlab Camera Calibration Toolbox [22].

Calibration was performed by taking a series of photos of a checkerboard pattern and manually selecting the four corners of the board in each image, after which the software provides an estimate concerning the location of the rest of the grid points. An optimisation step is then performed, and grid points are reprojected onto the image using the found parameters to assess their degree of accuracy. The reprojection error is quantified as the difference between the estimated grid point locations as ascertained by the parameters and the locations found using feature detection methods.

The results obtained from camera calibration are discussed in the Results chapter.

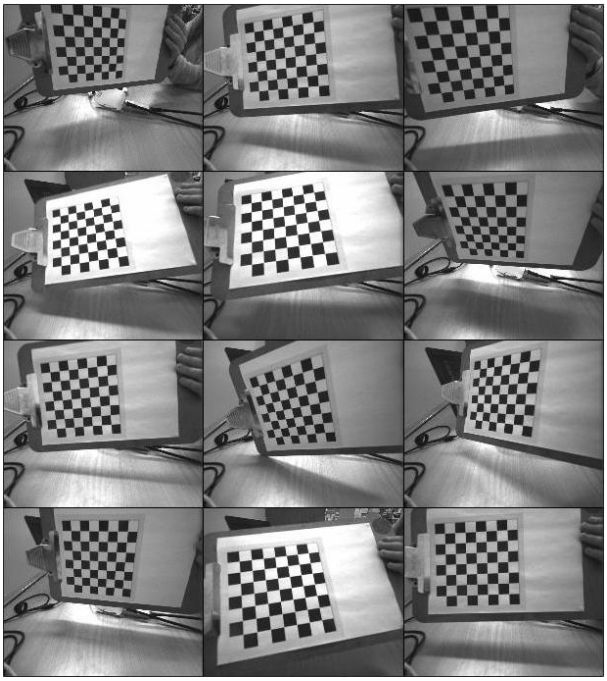


Figure 3.15: Images taken with one of the cameras of a checkerboard pattern from various angles, to use for calibration to find the distortion coefficients and intrinsic camera parameters.

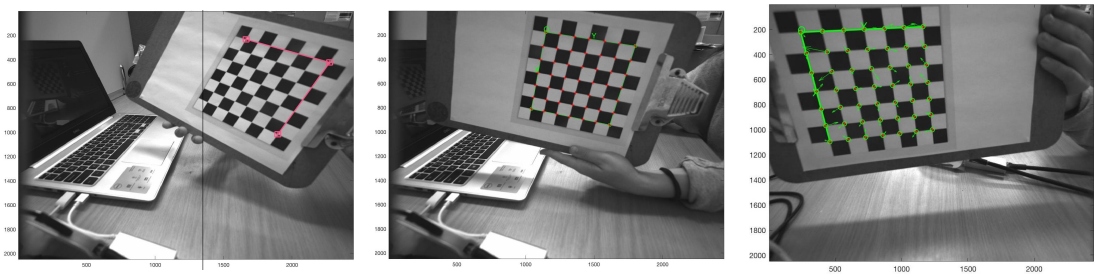


Figure 3.16: Images showing the stages of calibration which include manually selecting corners (left), the automatic extraction of grid points (centre) and the comparison of extracted and reprojected grid points using the estimated intrinsic parameters (right).

3.3.2.2 IMU

Yost Labs fortunately provides an open source calibration software, which removes the complexities generally experienced with IMU calibration. As the IMU will be aligned with the ship's coordinate system, its calibration can only be performed once aboard the vessel.

The calibration accuracy of the IMU for extended periods of time was, however, preliminarily assessed by recording a reading every second from a static IMU for just over an hour. As can be seen in the results (Figs. 3.17 and 3.18), there is a substantial amount of sensor noise and a bias present. The device subtracts gravity from the accelerometer readings and therefore, they should all be centred around zero. The gyroscope error appears to

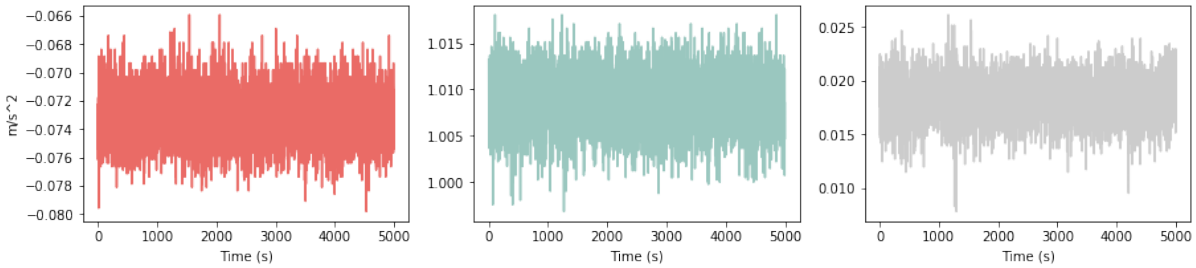


Figure 3.17: Plots showing the gyroscope readings from a static IMU for just over an hour. X-axis (left), y-axis (centre) and z-axis (right).

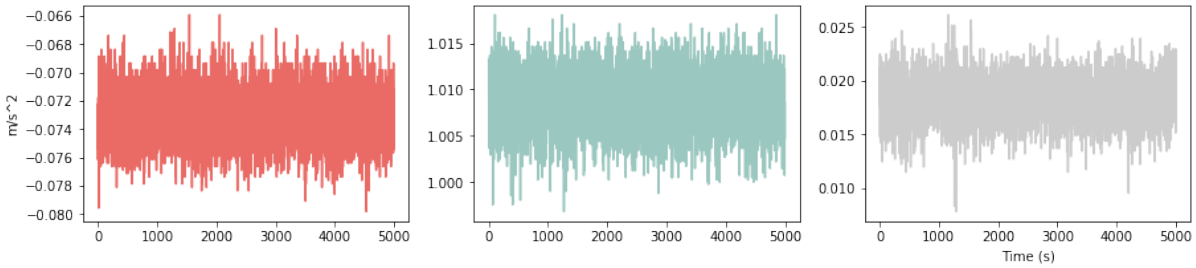


Figure 3.18: Plots showing the accelerometer readings from a static IMU for just over an hour. X-axis (left), y-axis (centre) and z-axis (right).

drift within an interval of approximately $0,005$ dps, and the accelerometer around $0,0075$ m/s^2. Their offset adds an additional uncertainty that is different for each of the axes. Histogram plots of the results (Figs. 3.19 and 3.20) show that the measurement noise appears to have a Gaussian distribution.

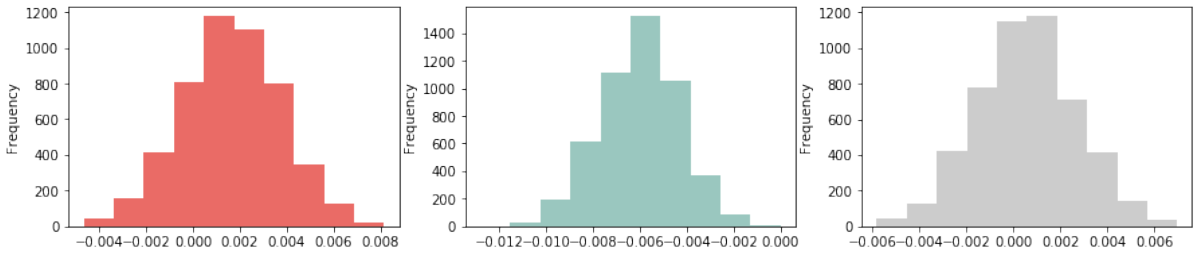


Figure 3.19: Histograms showing the gyroscope readings from a static IMU for just over an hour. X-axis (left), y-axis (centre) and z-axis (right).

Further IMU testing over a longer period of time was conducted and is discussed in the Results chapter.

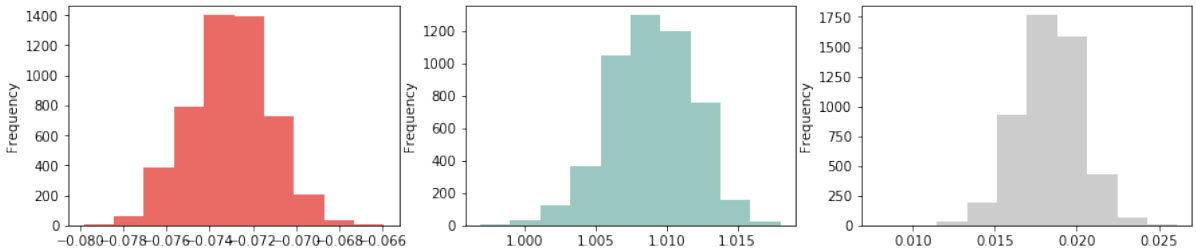


Figure 3.20: Histograms showing the accelerometer readings from a static IMU for just over an hour. X-axis (left), y-axis (centre) and z-axis (right).

3.3.3 Acquisition

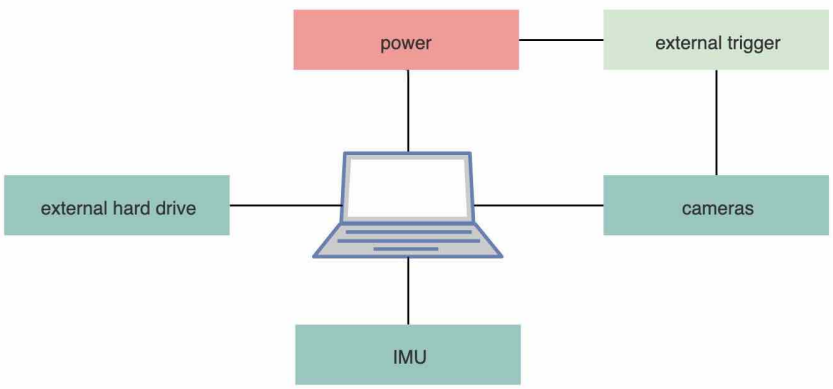


Figure 3.21: A diagram of the experimental system set up. The external trigger is used only to provide power to the cameras.

Data acquisition on the ship generally occurs for the duration of daylight hours. In the Summer, this can mean acquisition is taking place for 24 hours. The system therefore needs to operate continuously for up to 24 hours at a time without lag or synchronisation issues occurring between each of the sensors. The system was set up for testing as shown in Fig. 3.21. The cameras and IMU were first tested individually before integrating the entire system.

3.3.3.1 IMU Acquisition

The Yost IMU is powered through its USB connection to the laptop and uses the *Threespace API* for communication between the laptop and device. An IMU class was set up, and acquisition testing was performed. The class and code structure are seen in Fig. 3.22. The *dataReceived* property is used for error checking, and the *counter* is used to track the data acquisition number.

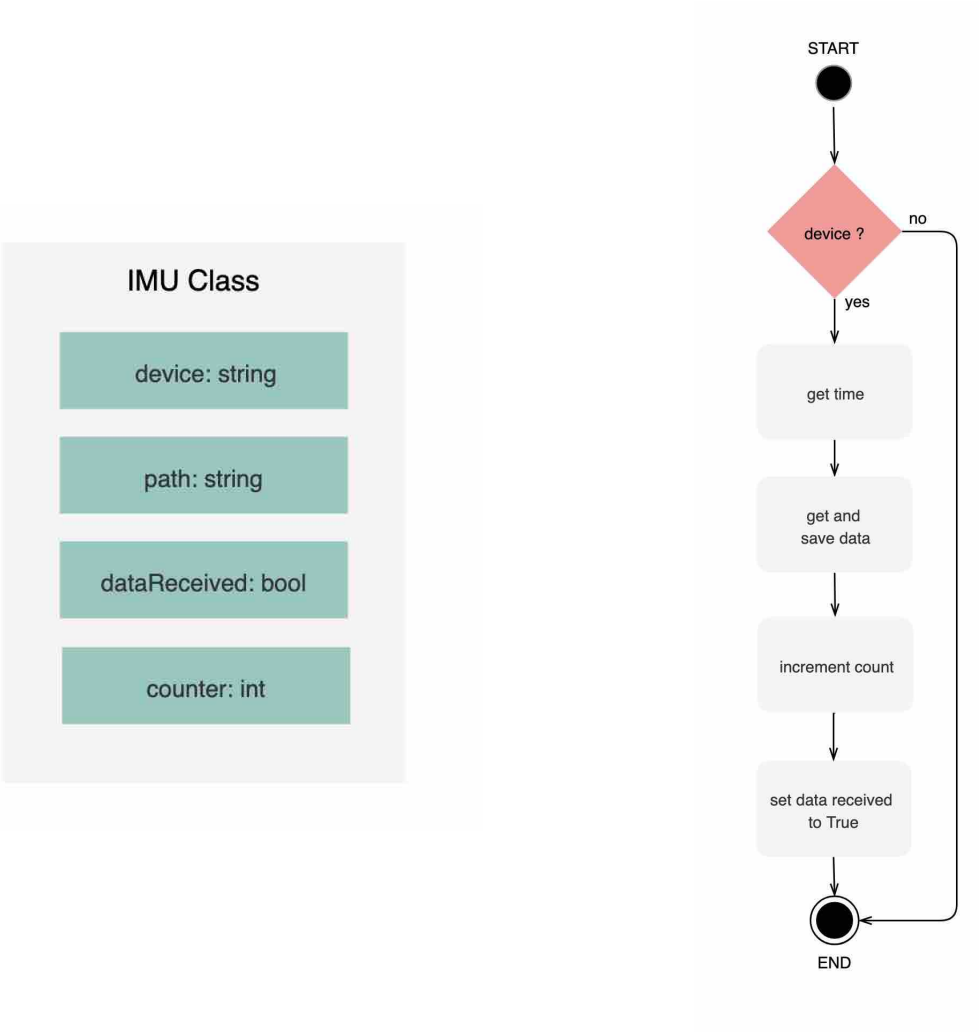


Figure 3.22: Diagrams detailing the IMU class structure (left) and IMU acquisition activity diagram (right).

As the IMU has a very high data rate, but is only required to retrieve data at one second intervals, there were no problems with lag in acquisition. As previously mentioned, the IMU offers data acquisition in multiple formats and Fig. 3.23 shows the acquisition times for these various formats. The times were recorded using Python's *time* library, which provides a time in seconds with microsecond precision, and found by averaging the results of acquisition times for 100 data points. This assessment allows us to evaluate whether there will be a limitation to acquiring multiple formats in one second intervals.

Surprisingly, obtaining raw data had the highest maximum acquisition time. The average acquisition times for all formats however, were well below 4 ms, indicating that there should be no problem whatsoever with obtaining as many formats as desired in the main system loop.

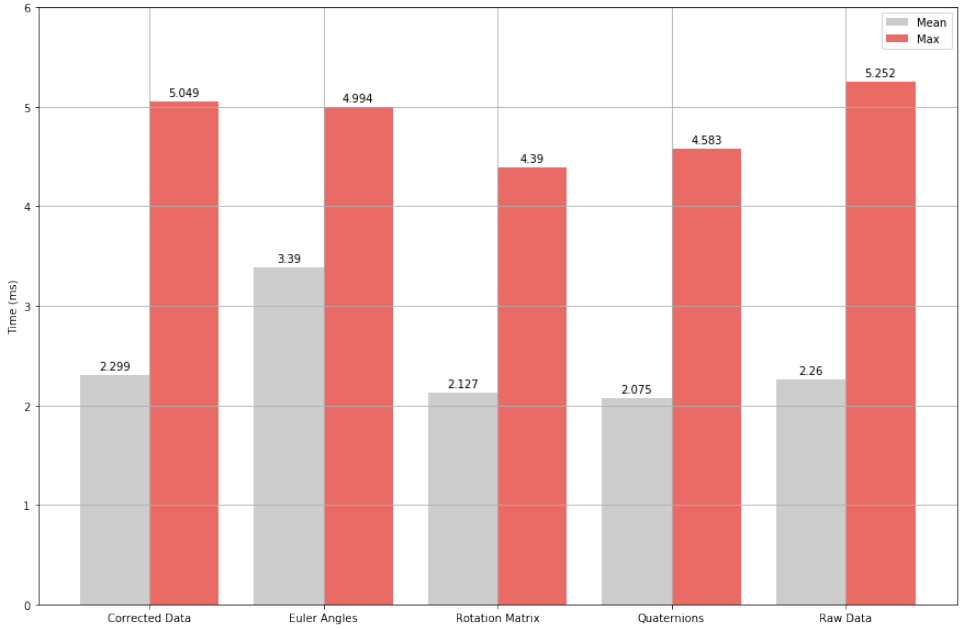


Figure 3.23: A bar graph showing average and maximum acquisition time for various formats of IMU data.

3.3.3.2 Image Acquisition

The cameras are powered using the external trigger box and wired to the laptop via ethernet. A fixed IP address was set up and a Linux SDK for interfacing with GigE cameras through Python code, *Tiscamera*, was utilised in the Python script. The script also depends on numerous other libraries, as shown in Fig. 3.24. As a result of the incompatibility between Linux and the hardware trigger application, a software trigger was tested instead.

The code operates by sending a software trigger to the camera at one second intervals, which exposes a single image frame to the laptop for saving. The camera class is attached to a callback function, which is described in Fig 3.25. The advantage of using a callback is that it implements a non-blocking wait for image acquisition, allowing the rest of the main code loop to continue running without having to continuously poll for a new image. The *busy* property ensures that the callback is not called again before completing an acquisition, the *imageReceived* property allows the main code loop to track the accuracy of acquisition and the *counter* property tracks the image number.

Once the script was set up, the first experiments were conducted to assess the acquisition time of various file formats, in order to identify which would be most efficient. In the previous system, image frames were saved as TIFFs, as this resulted in the least amount of time lag, despite TIFF being the largest image format. It is suspected that the

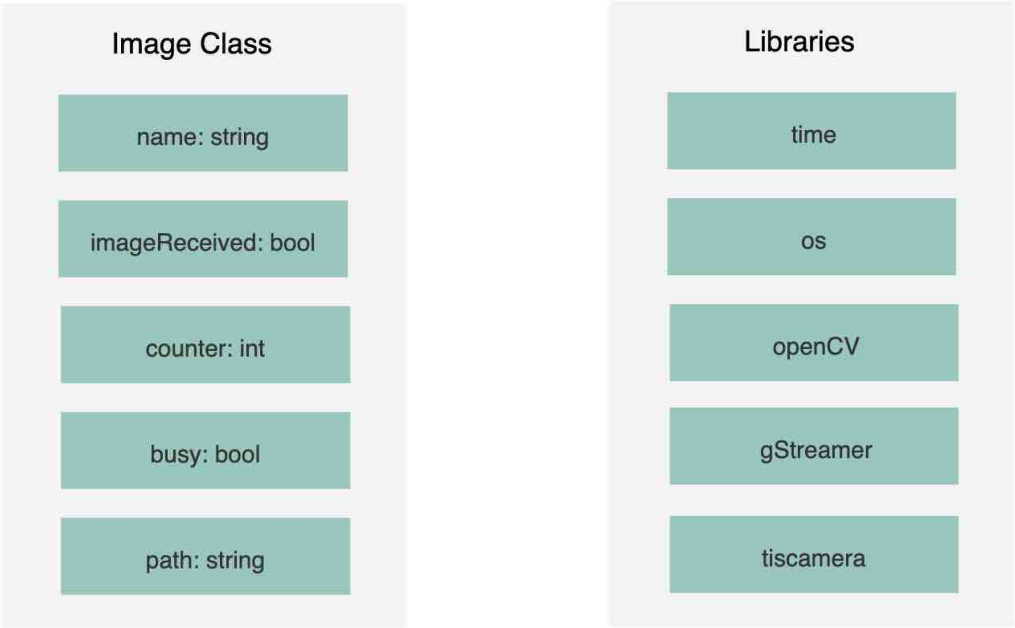


Figure 3.24: Diagrams showing the image class structure (left) and libraries required for the acquisition code (right).

compression step used for jpg and png took longer in Matlab than the time taken to actually save the image. In the experiments run on Linux, the results were more in line with what we would expect, with TIFF found to have the worst performance, and bmp the best. Tests for each format were run for one hour.

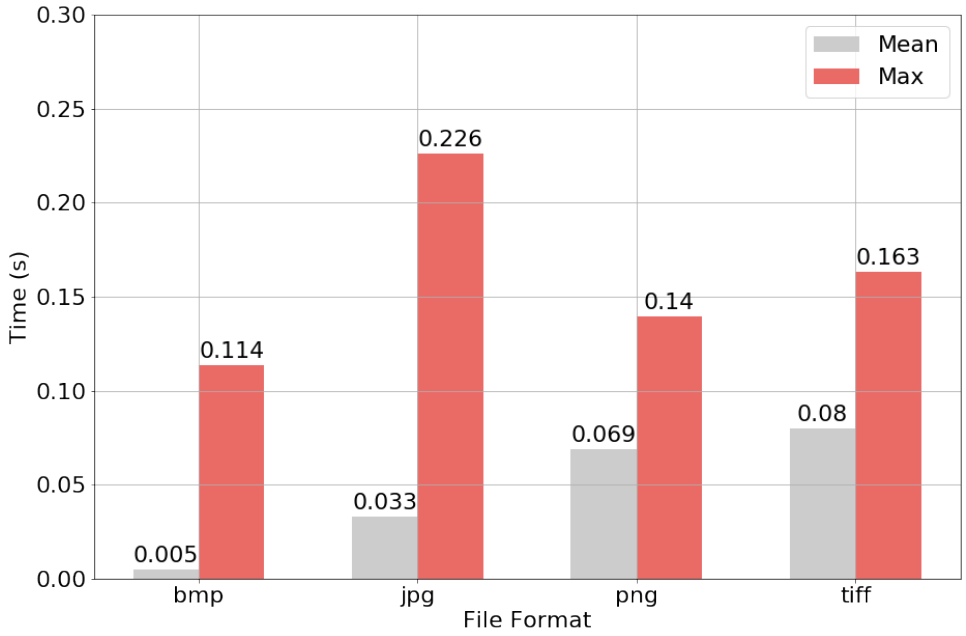


Figure 3.26: A bar graph showing average and maximum acquisition times for various image file formats.

Once the bmp file format had been determined as the most efficient, experiments were

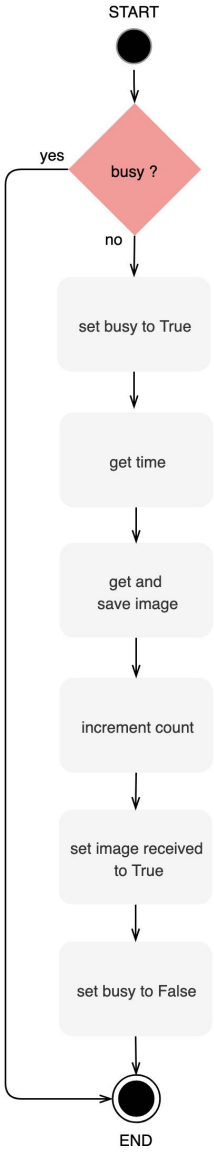


Figure 3.25: An activity diagram for the image callback function.

run using a single camera to assess if there was any lag in acquisition. The system was run multiple times for extended periods and was found to have no lag or missing images for every iteration of acquisition. This led to an acceptable condition to proceed to the sensor integration and synchronisation stage of development.

3.3.3.3 Integration and Synchronisation

The performance metrics to be optimised included (1) the error rate, or number of missed data points, (2) the sensor synchronisation and (3) the time interval between subsequent acquisitions. The aim was to obtain less than one missing data point out of every 100,

for the difference in acquisition time to be less than 1 ms for the cameras and 100 ms between the cameras and IMU, and for the time interval between acquired data points to be within 1 ms of the specified 1 s interval for every second of acquisition.

Numerous experiments were run in order to assess the performance metrics, and various parameters, along with the structure of the code, were altered until optimal performance was acquired.

Error Rate

There was an initial rise in the error rate when introducing both cameras and the IMU to acquisition, which led to numerous iterations of code restructuring. The final implementation found to be successful is seen in Fig. 3.27.

The primary alterations made to the original code structure included:

- adjusting the specified camera parameters when initially connecting to the cameras, namely the frame rate, which seemingly was not an indication of the camera's actual frame rate but rather how many frames the camera was being instructed to capture.
- performing IMU acquisition inside the main loop rather than through a function call, as this took more time to execute.
- restructuring the way the loop implemented the time delay and dealt with detected errors. Error checks were initially being performed before the image callbacks had finished executing, resulting in many false errors and misleading data counters.
- reducing the amount of work done in the image callbacks to optimise execution time.

The error rate was recorded for the majority of experiments conducted during testing, and the graphs below illustrate the changing error rate as iterations progressed. The longer preliminary tests, of 6, 18 and 24 hours, are highlighted on the plots. The results displayed show that the error rate became progressively lower and remained at an acceptable value after the 35th iteration of testing. The preliminary longer duration tests, specifically the 18-hour test, in which there was only a single missing image, led to a satisfactory condition to proceed to the assessment of the next metric.

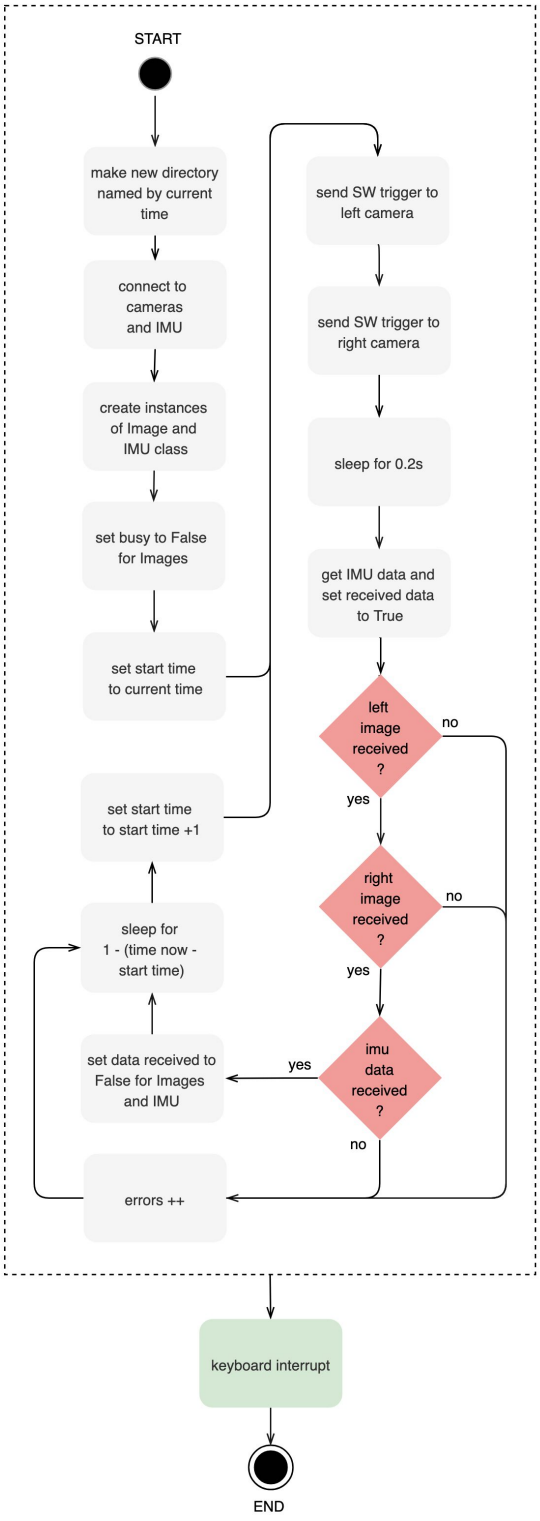


Figure 3.27: An activity diagram for the main acquisition loop.

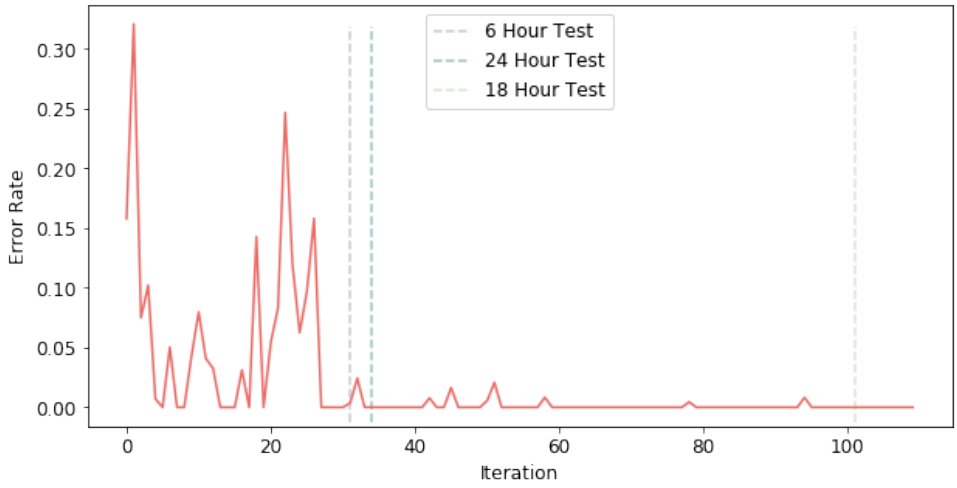


Figure 3.28: A plot showing the decreasing error rate during the numerous tests conducted.

Synchronisation

In order to assess the accuracy of camera synchronisation, the system was set up to take images of a digital stopwatch that included millisecond precision, running on the laptop screen. The images were then cropped to display only the time, and a Python script written to extract the writing from the image in order to automatically build arrays with the timing information.

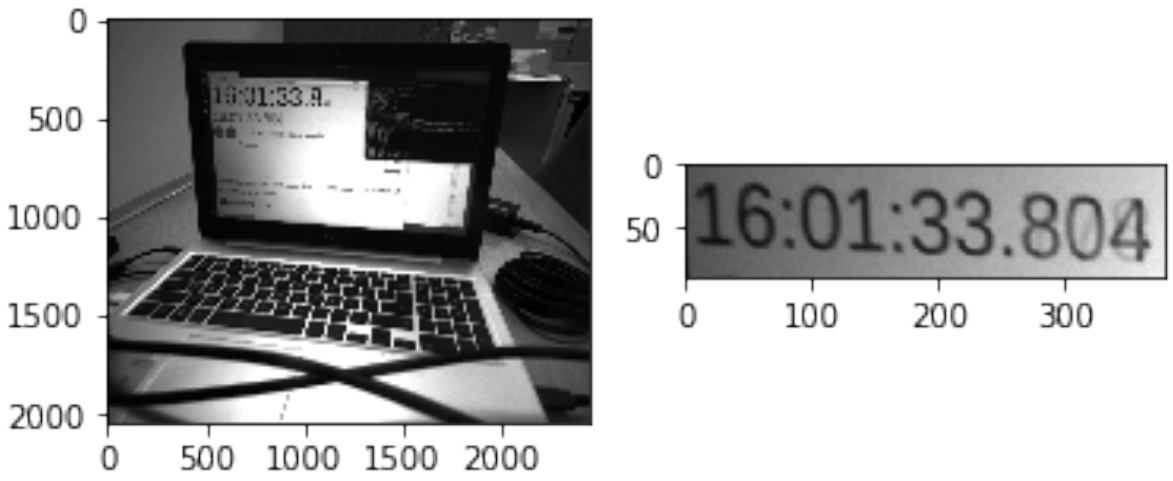


Figure 3.29: An image showing the time extracted from the left camera during the 24-hour test.

The results from the 6-, 18- and 24-hour tests were overwhelmingly positive, with all images that were assessed synchronised to the millisecond. Figs. 3.29 and 3.30 show that the time at which corresponding image frames were captured was exactly the same after 16 hours of acquisition.

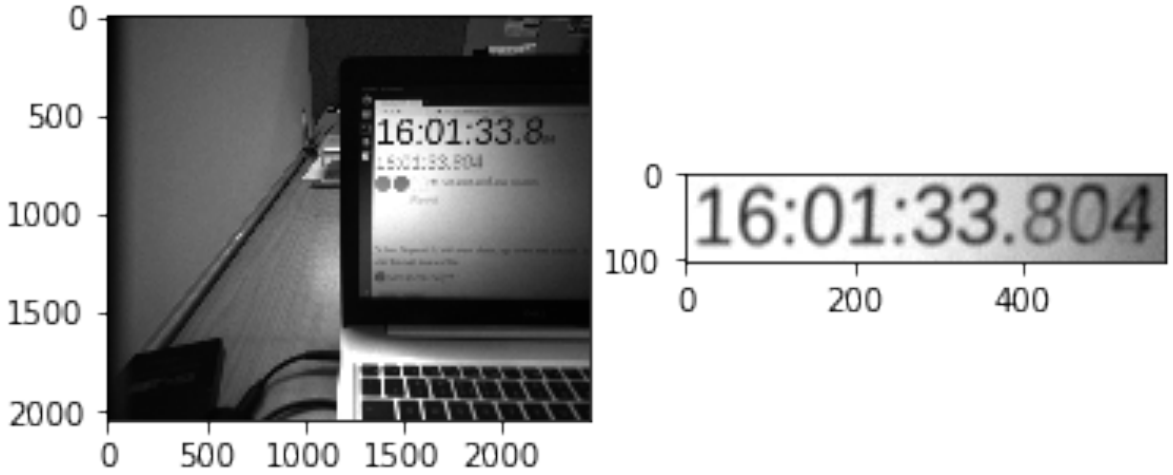


Figure 3.30: An image showing the time extracted from the right camera during the 24-hour test.

A substantial amount of work had to be done to accurately extract the time from images. The changing light in the lab over the 24-hour period demanded that the contrast and brightness be adjusted in an iterative loop until a time could successfully be extracted. Extracted times were then assessed sequentially, and if an outlier was detected the image was processed manually. A second manual assessment was performed if a mismatch was found when comparing the times extracted between image pairs. The results of the preliminary 24-hour test is shown in Table 3.9.

Running time	24 hours
Image pairs captured	86 091
Image pairs analysed	860
Matching image pairs	860

Table 3.9: The preliminary results of camera synchronisation from the 24-hour test.

Because of the large amount of data collected — 86 091 images from each camera during the 24-hour test for example — and the lengths taken to accurately retrieve the time from an image, only every hundredth image pair was compared. However, many hours were spent manually looking through arbitrary sequences of image pairs from different acquisition sets, which confidently led to the conclusion that the successful results of the 100 assessed images can be extended to the entire image set.

To assess the synchronisation of the IMU and cameras, their respective timestamps were compared. Through testing it was found that a delay had to be executed in between the

software trigger and IMU data capture to prevent the IMU data being retrieved much earlier than the images. With experimentation and manual assessment of the timestamps, it was found that a 200 ms delay resulted in more accurate timestamp synchronisation, the results of which are shown in Table 3.12.

Data points captured	3 650
Average time difference (s)	0,002
Maximum time difference (s)	0,001

Table 3.10: The absolute difference in acquisition time between left camera images and IMU data over one hour using timestamp comparison.

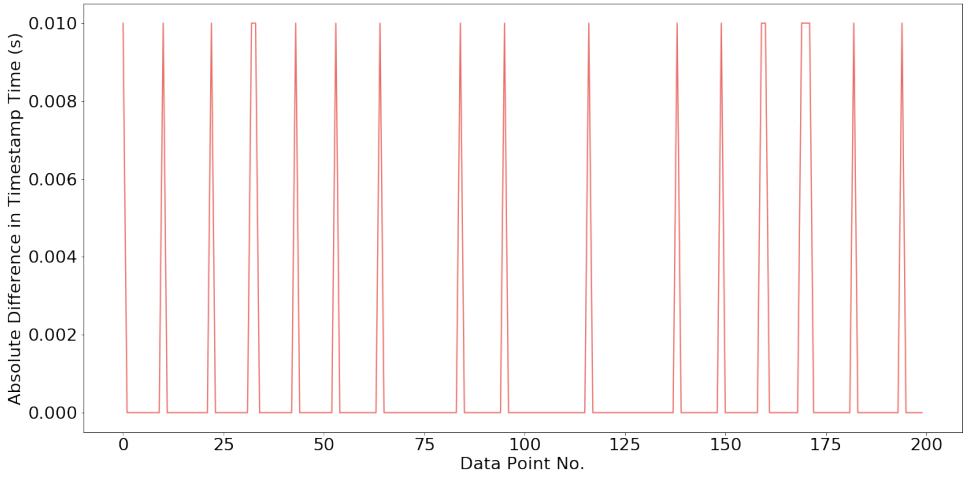


Figure 3.31: A plot showing the absolute difference in acquisition time between left camera images and IMU data for 200 data points.

As seen in the results table and Fig. 3.31, the difference in acquisition time between the cameras and IMU never went above 100 ms, successfully satisfying the second requirement for synchronisation.

The timing diagram for system execution, with average obtained execution times for each stage, is detailed in Fig. 3.32. Note that IMU acquisition executes 200 ms after the image triggers. This indicates that there is uncertainty in the time difference between image capture and saving the image, as the frame could be captured upon trigger execution or later on in the callback. This potential offset between the image timestamp and capture time makes timestamp comparison a less precise method of evaluation for IMU-camera synchronisation.

To mitigate this uncertainty, the digital clock time captured was compared to the image timestamps. Fig. 3.33 displays the successive timestamp and digital clock times obtained

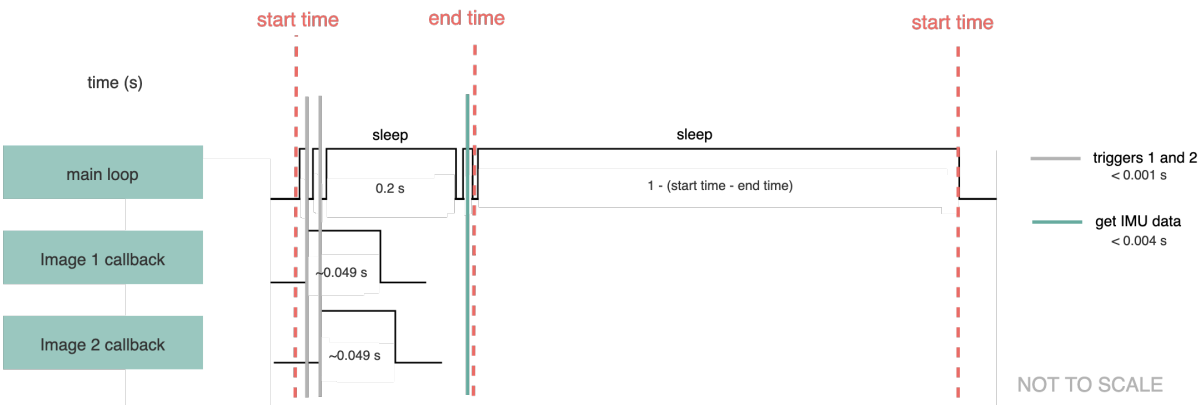


Figure 3.32: A timing diagram for the acquisition system's code execution.

from the 24-hour test. They are plotted in relative rather than absolute values, which allows us to reason that if there is an offset, it is mostly consistent.

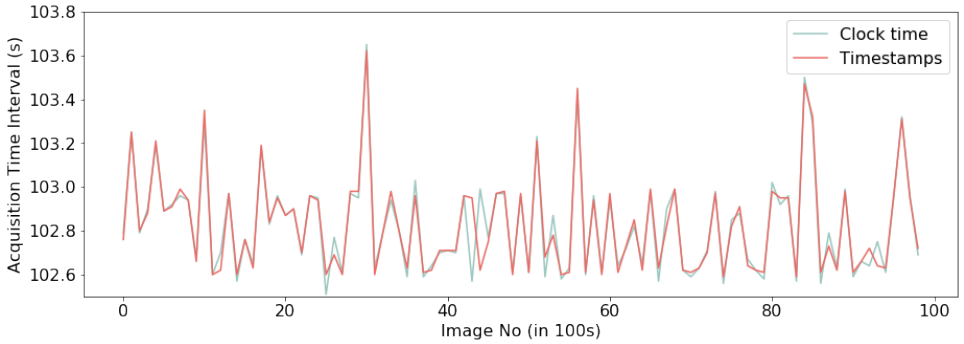


Figure 3.33: A plot showing the timestamp and extracted digital clock time intervals for images from the 24-hour test.

There are also considerations around the time taken to acquire IMU data, as the timestamp is recorded just before acquisition is actually executed. For the purposes of the ship however, where movement is slow and continuous — when compared with IMU applications recording bodily movements for example — it was determined that this level of accuracy comparison is sufficient. This is especially true when considering that the entire process of recording the time and capturing the IMU data executes in less than 4 ms on average for every format (Fig. 3.23).

Based on the timing diagram, the delay in acquisition between IMU and images will be approximately 200 ms in the worst case. This is double our required 100 ms synchronisation time, however, the possibility of this error occurring is determined to be acceptable for now — especially when considering that the IMU parameters obtained are used in relative rather than absolute terms for image correction — until further investigation can be conducted using data from the ship.

Time Interval

The time interval between successive data points is where the disadvantage of not using an external hardware trigger becomes apparent. Through analysis of the data points' timestamps and the difference in the digital clock time between sequential images, it was found that the accuracy of Python's *time.sleep* function was not precise enough and resulted in an accumulating time drift error. For example, in the preliminary 24-hour test, which should have captured 86 400 images, only 86 091 were triggered, indicating a time error drift of 309 s. Whilst this does not seem like a significant error, it can become problematic if acquired data is being used in conjunction with other sensor data obtained on the ship. Figs. 3.34 and 3.35 show the difference between the ideal time and actual time after an hour of acquisition, as well as the non-ideal nature of the time interval.

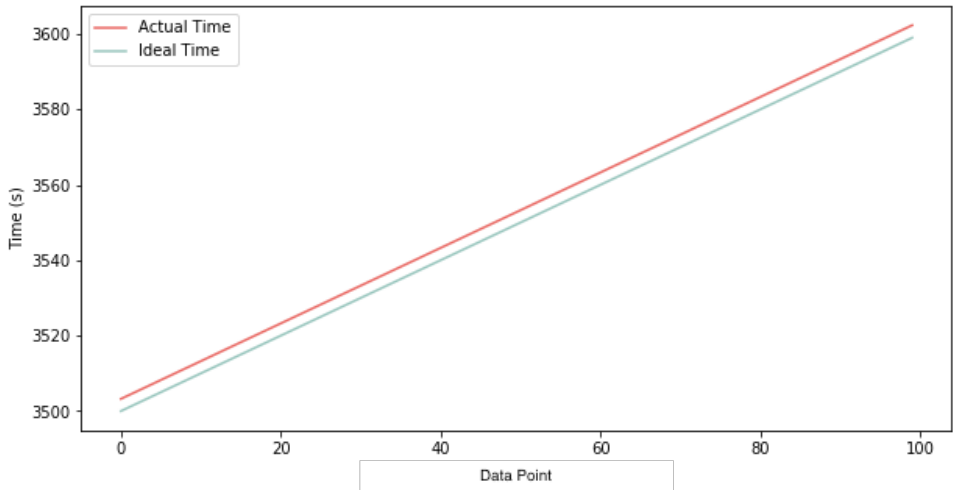


Figure 3.34: A plot showing the difference in the ideal time and actual time for 100 data points after one hour of acquisition.

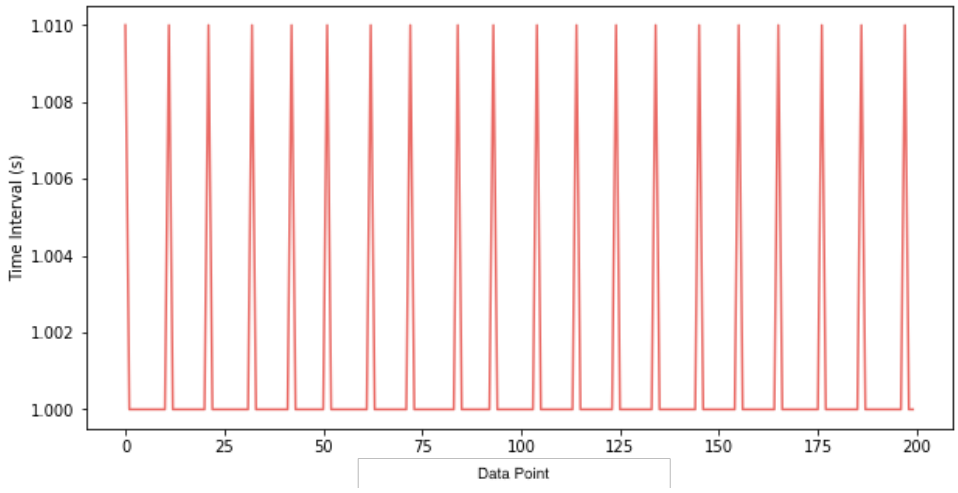


Figure 3.35: A plot showing the intervals between recorded timestamps for 200 data points.

Approximately every ten images, there was an increase in the time interval from one second to 1,01 s, indicating that *time.sleep* actually implements a wait slightly longer

than specified. According to documentation, the *time* library is dependent on the system architecture and implemented slightly differently on every machine resulting in a time delay that may be slightly less or more than requested. Experimentation was therefore conducted to manually find a more accurate time interval.

A time of 0,999052 s was found to produce the most consistent one second time interval for acquisition. The results of a one-hour test run using this new interval are found in Table 3.12. They show that the overall time drift for roughly one hour of acquisition has been reduced from more than 3 s to less than 10 ms for both the cameras and the IMU.

Time interval	**1 s**	**0,999052 s**
Data points	3 637	3 650
Average interval (s)	1,000921	0,999978
Maximum interval (s)	1,01	1,5
Minimum interval (s)	0,99	0,5
Variance (s)	$9,302205 \times 10^{-6}$	0,000138
Errors (intervals $\neq 1 \pm 0.001$ s)	369	30
Overall time drift (s)	3,35	-0,08

Table 3.11: The results from the interval tests assessing left camera timestamps for approximately 1 hour of testing using time intervals of 1 s and 0,999052 s.

While image acquisition intervals clearly improved, IMU acquisition regressed in terms of intervals not equal to 1 s. As the time drift is a more important metric to optimise, and overall time drift was reduced in both cases, we can overlook the increase in variance for IMU acquisition.

Time interval	**1 s**	**0,999052 s**
Data points	3 637	3 650
Average interval (s)	1,000924	0,999997
Maximum interval (s)	1,01	1,5
Minimum interval (s)	0,99	0,49
Variance (s)	$8,664386 \times 10^{-6}$	0,000178
Errors (intervals $\neq 1 \pm 0,001$ s)	346	1 384
Overall time drift (s)	3,36	-0,01

Table 3.12: The results from the interval tests assessing IMU timestamps for approximately 1 hour of testing using time intervals of 1 s and 0,999052 s.

Data Storage

A method of logically saving data that could be implemented in all future systems for continuity was determined, using the ordering and naming conventions laid out in Fig. 3.36. Each image and IMU data file will be named by its count and timestamp, ensuring correct data correspondence in post-processing.

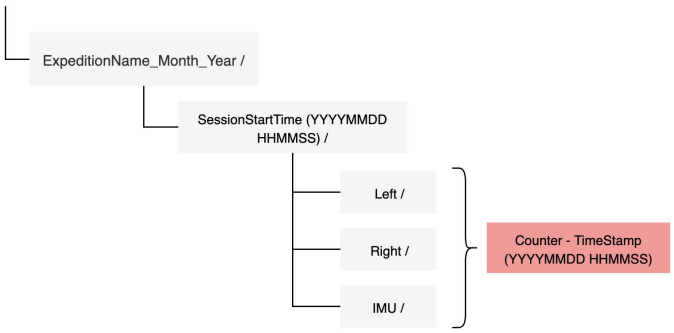


Figure 3.36: A diagram detailing the filing structure determined for the upcoming and future expeditions, in order to create some consistency for the organisation of datasets obtained.

3.3.4 Orientation Estimation and Correction

The horizon in each image is an indication of the roll and pitch of the ship relative to a particular reference orientation. Yaw, the rotation about the z-axis, cannot be determined from the horizon, however, it is not important for the purposes of image correction.

Orientation can be quantified by yaw (θ), pitch (ϕ) and roll (τ), which describe rotational displacement around the z, y and x-axes respectively.

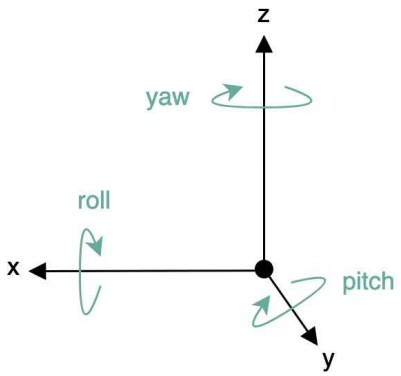


Figure 3.37: Diagrams showing the parameters describing orientation.

Fig. 3.38 shows how the changing location of the horizon in an image, in terms of both a translation and rotation, is a reflection of a ship's movement.

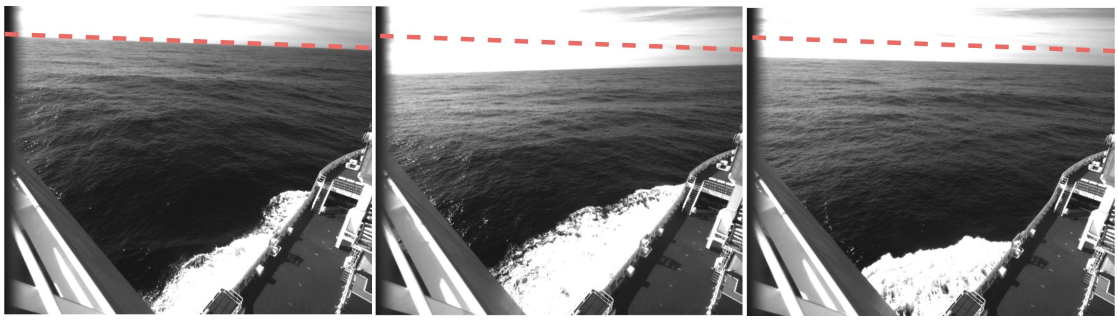


Figure 3.38: A series of images showing how the change in the location of the horizon in the image is an indication of the ship's relative orientation, with the reference horizon from the left image highlighted in red.

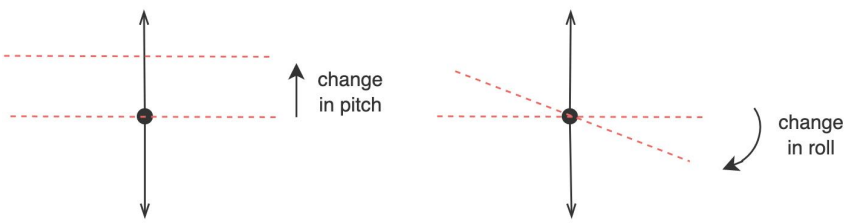


Figure 3.39: Diagrams demonstrating how the changing location of the horizon line (red) relates to a change in the ship's roll and pitch.

Image correction is a three step process (Fig. 3.40). Firstly, it involves developing a method for detecting the horizon to get the horizon equation. Secondly, the equation is used to estimate the ship's relative orientation. Lastly, a method must be developed for performing the correction.

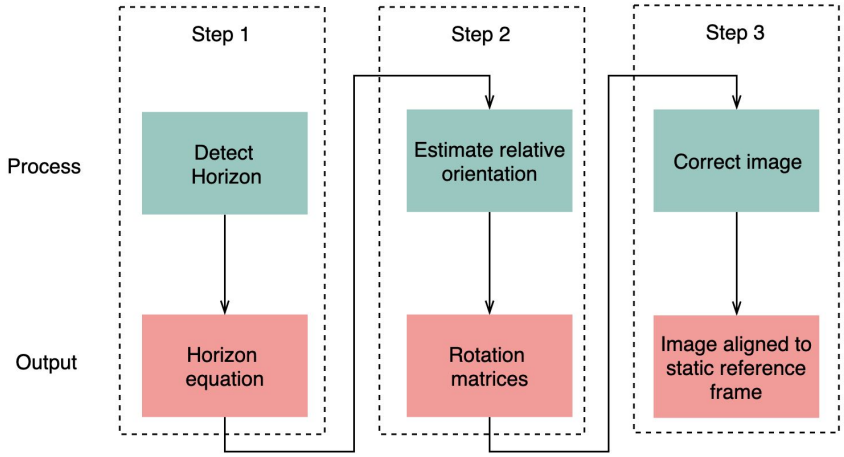


Figure 3.40: A diagram showing the horizon detection and image correction processing stages.

It is important to note that the horizon can only be used to estimate orientation relative to a static reference orientation, rather than an absolute orientation. In this case, the static reference horizon, representing the reference orientation, would be the horizon as captured when the ship is docked and experiencing minimal motion.

3.3.4.1 Coordinate Systems

Before performing orientation correction, the various coordinate systems need to be defined. In this context there are three, the global (world), the local (ship) and the cameras.

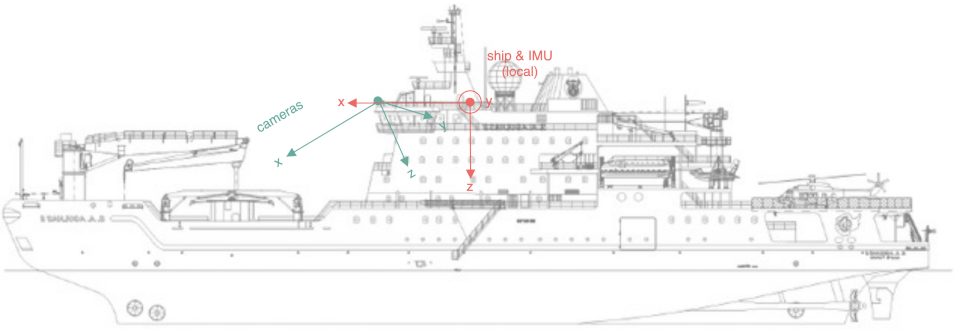


Figure 3.41: A diagram showing the coordinate systems of the cameras and ship on the SA Agulhas II.

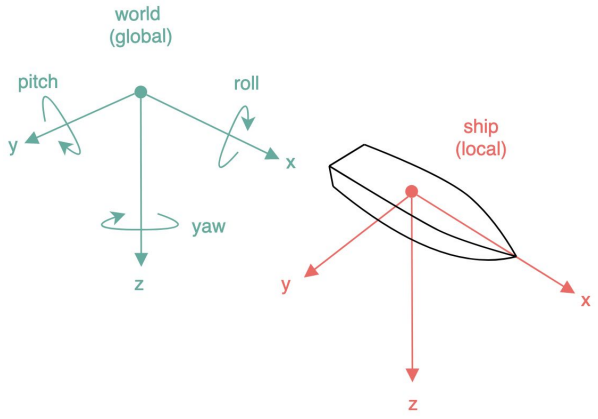


Figure 3.42: A diagram showing the coordinate systems of the ship and world reference frame.

Global: The global frame is a static reference, fixed to the earth and used to determine the relative orientation of the other coordinate systems. This frame is represented by the horizon in each image, and is the coordinate system we want to correct images to.

Local: The local system defines the orientation of the ship, specifically with respect to its roll and pitch in relation to the global frame. Rotation about the z-axis (yaw) is unimportant for image correction, but determined to be aligned with the ship's heading. This local frame's relative orientation is generally recorded by the IMU.

Cameras: The camera frame is fixed within the local frame. Its axes differ as the cameras are angled downwards towards the sea and as a result of their placement, are not aligned with the ship's heading. The origin is located at the optical centre of the camera, with the z-axis pointing along the optical axes. This means that each camera technically has its own coordinate system, but since they are parallel and fixed, their relative orientation to the local system is the same. The cameras' field of view (FOV) overlaps and includes both a section of the ship and the horizon. (Figs. 3.43 and 3.44).

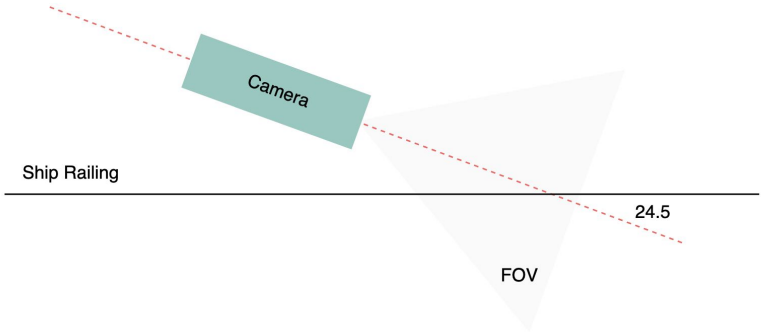


Figure 3.43: A diagram showing the set-up of the cameras in relation to the ship in elevation.

The fact that the camera coordinate system is fixed relative to the ship is what allows us

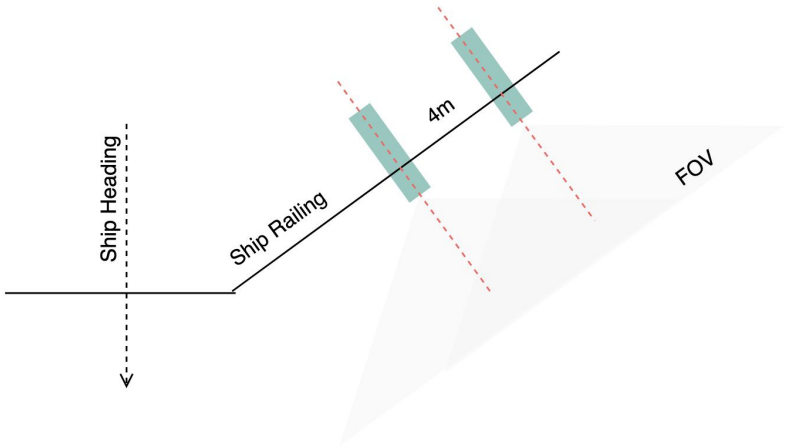


Figure 3.44: A diagram showing the set-up of the cameras in relation to the ship in plan.

to use it to determine an estimate of the ship's orientation. The differences between the camera and ship frames are only relevant when performing orthorectification, meaning that we can view the ship and camera system as one for the correction process.

Each element of orientation is represented by a $3x3$ rotational matrix, which gives the overall rotation $\mathbf{R}$ when multiplied together. For image correction, we consider only angles associated with roll (τ) and pitch (ϕ). Yaw is therefore described by $\theta = 0$.

$$R = R_z(0)R_y(\phi)R_x(\tau) \tag{3.1}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & -cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & -cos(\phi) \end{bmatrix} \begin{bmatrix} cos(\tau) & -sin(\tau) & 0 \\ sin(\tau) & cos(\tau) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

3.3.4.2 Detection

The Hough Transform applied with Canny Edge Detection was used to detect the horizon line. These methods are provided by the *OpenCV* library. The Hough Transform outputs an array of lines in polar coordinates, which are ordered by the number of pixels present in each line equation. Therefore, the longest or most prominent line is the first element in the array. Fig. 3.45 shows the multiple lines identified on an image from the July expedition using this method.

It was found that before applying this method, accuracy improved if images were first put through a Gaussian filter, to smooth the edge lines of any sea ice present. Images were also cropped to prevent the prominent lines of the vessel being the strongest lines detected.

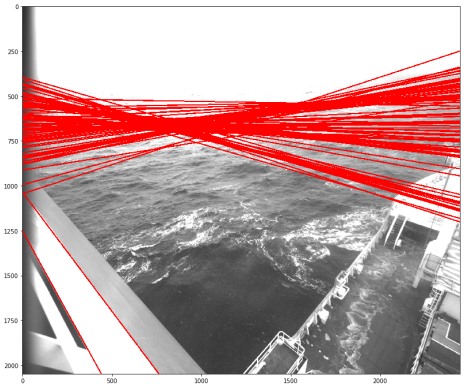
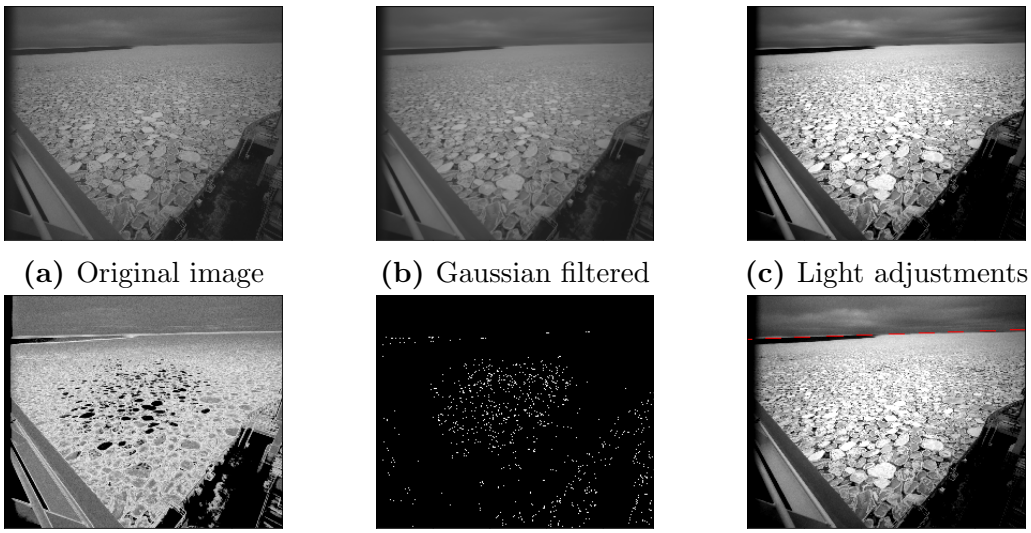


Figure 3.45: An image from July showing the lines detected using Canny Edge Detection and the Hough Transform. Note the prominent lines of the ship that are detected.

Fortunately, the ship could be completely cropped from all images whilst leaving the full horizon visible. Through experimentation with various filters, entropy based filtering was found to be the most effective as a precursor to Canny Edge Detection. Detection also performed better when adjustments were made to the brightness and contrast of the image. These editing stages are shown in Fig. 3.46.



(a) Original image **(b)** Gaussian filtered **(c)** Light adjustments

(d) Entropy filtered **(e)** Canny edge detection **(f)** Detected horizon

Figure 3.46: A series of images showing the outputs of the horizon detection algorithm at various stages in the process.

This processing pipeline was tested on images from the collected data deemed challenging, as a result of weather and lighting conditions, background objects, blurred housing windows and the lack of contrast between sea and sky due to the presence of ice. Through this process, it was found that the detection is iparticularly sensitive to the adjustments made to the image contrast and brightness, with many images requiring different parameters to be carefully selected. In so doing, the detection was generally satisfactory.

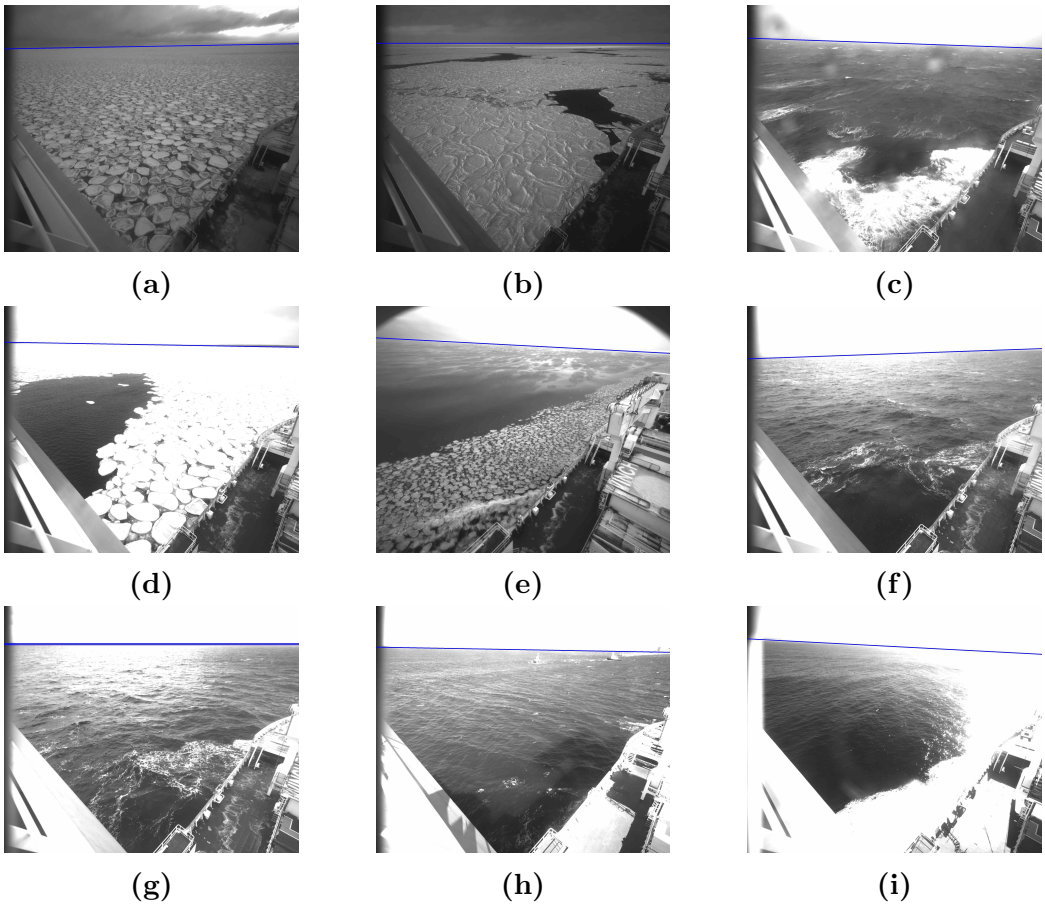


Figure 3.47: A series of challenging images showing successful horizon detection.

In some images however, the horizon was not detected either because it was not visible or because the environment and lighting conditions made it unrecoverable. To account for the images where no horizon was detected, as is seen in Fig. 3.48, where the horizon is impossible to recover, they had to be recorded as unidentified and discarded.

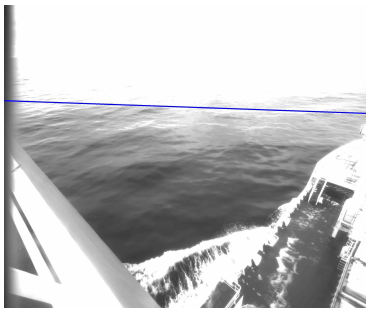


Figure 3.48: A problematic image, where the horizon is undetected due to it not being visible from overexposure.

3.3.4.3 Estimation

Using the equations from Section 2.3.2, the polar coordinates (r, θ) of the detected line were used to calculate roll (τ) and pitch (ϕ).

$$\tau = tan^{-1}[\frac{-f_x}{f_y tan(\theta)}] \tag{3.3}$$

$$\phi = \frac{\pi}{2} - tan^{-1}[\frac{f_x sin(\tau)cos(\theta) - f_y cos(\tau)sin(\theta)}{r - c_x cos(\theta) - c_y sin(\theta)}] \tag{3.4}$$

To assess the reliability of the estimations obtained, the data from left and right cameras were plotted for comparison (Figs. 3.49 and 3.50). As left and right cameras experience the same movement, their relative roll and pitch between sequential images should be identical. The values were compared by first subtracting the roll and pitch of the reference horizon — taken to be the first image in the sequence.

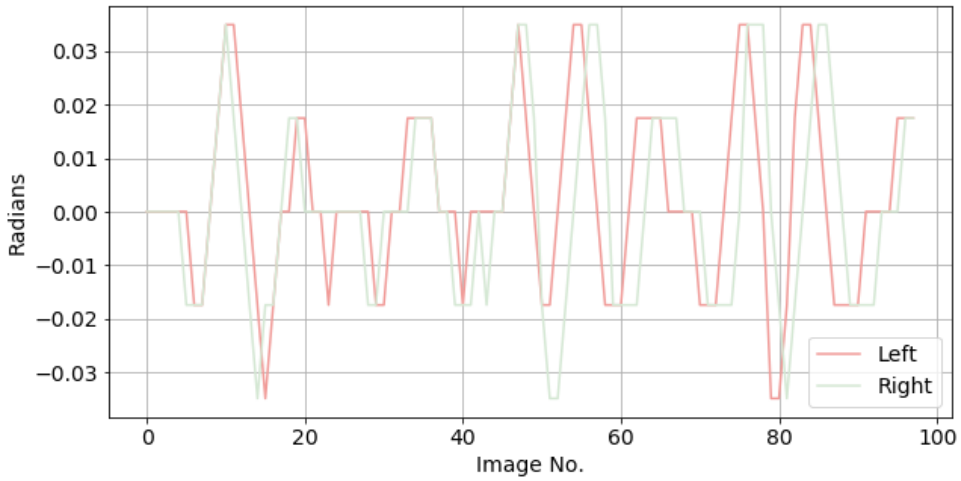


Figure 3.49: A plot of the sequentially obtained values for roll from left and right image sequences.

The inconsistencies between left and right can primarily be accounted for by system lag as a result of problems with acquisition. These problems meant that images were skipped and timestamps were not accurately recorded. This made it impossible to recalibrate corresponding images. However, if we look past the occasional offsets that occur, the path the plots follow are similar enough to conclude that estimation is being accurately obtained.

There are more considerable disparities in the graph showing pitch. This could be due to the fact that errors in the detected horizon's polar coordinate r lead to uncertainties only in pitch, and do not affect the accuracy of estimated roll. This implies that the

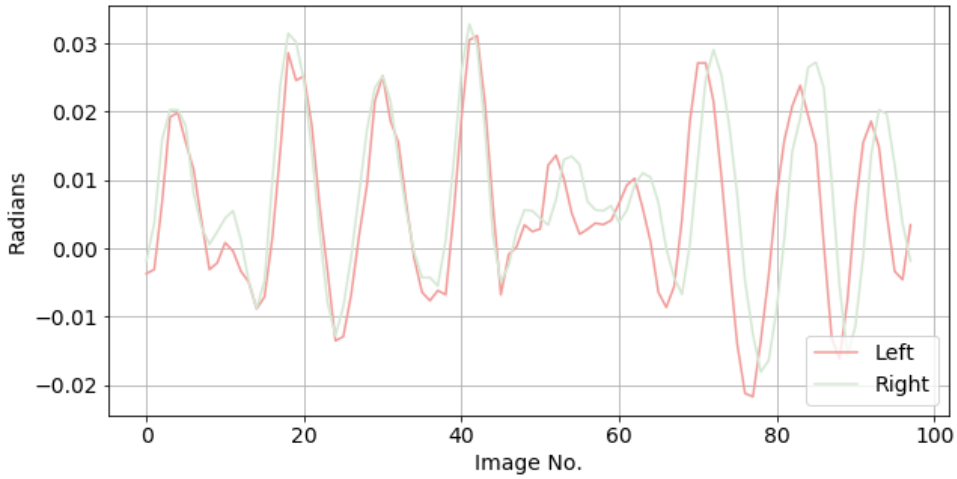


Figure 3.50: A plot of the sequentially obtained values for pitch from left and right image sequences.

computed r value is being estimated with some degree of pixel error.

In order to prevent false estimations when automating the algorithm, a method of accuracy checking was investigated. It compares the orientation obtained from the first element of the Hough Transform array to the estimated orientation of the previous image. Since the movement of the ship is relatively slow and consistent, it is unlikely that it will experience any drastic changes in orientation. Therefore, if there was a change detected which was above some threshold, the next element obtained in the array from The Hough Transform was evaluated. If an appropriate line was not found, the image was noted as having an undetectable horizon.

To quantify a threshold for the maximum change in orientation, a smaller set of sequential images were first manually assessed for correctness, and changes in their orientation then measured. The maximum change in roll and pitch obtained — plus an additional degree of uncertainty — were then used as the threshold value for the larger set from the same sample. An example of maximum change parameters obtained for a subset of 100 images is detailed in Table 3.13. Again, the values for roll between left and right were identical, but there was a mismatch in pitch.

	Left images	**Right Images**
Maximum relative change in roll (rads)	0,035	0,035
Maximum relative change in pitch (rads)	0,016	0,015

Table 3.13: The maximum orientation changes obtained from a correctly detected sample of 100 sequential images.

3.3.4.4 Correction

The correcting transform applied to each image was computed relative to the horizon obtained when the ship was static, using the *OpenCV* function *warpPerspective*. This function takes as input the calculated transform matrix **H**, consisting of the computed image orientation **R**, the reference image orientation $\boldsymbol{R_{ref}}$ and camera matrix K.

$$H = (KR_{ref})(KR)^{-1} \tag{3.5}$$

This method was first tested on a set of artificial data, where an image of a horizon was rotated and shifted in Photoshop (Fig. 3.51), in order to assess the accuracy of correction. The desired result, the target image being rotated and warped so that the horizon is identically located to its placement in the reference image, was successfully achieved (Fig. 3.52).

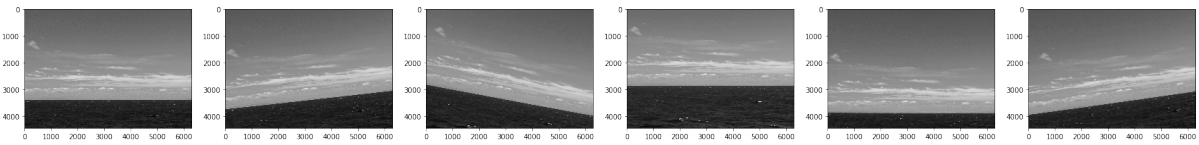


Figure 3.51: An artificially created sample data set (reference image is first on the left).

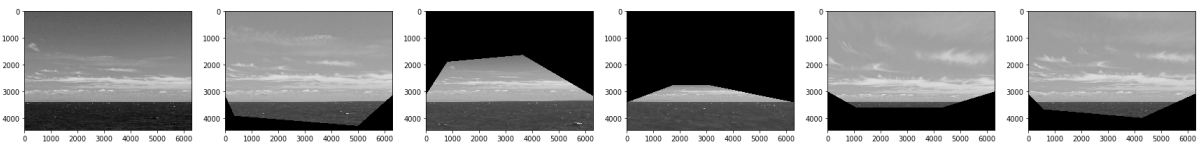


Figure 3.52: A series of images showing the results of correction (reference image is first on the left).

An example of image correction applied to the data collected in July is shown in Fig. 3.53. As the ship's movement is much less pronounced than was simulated in the artificially

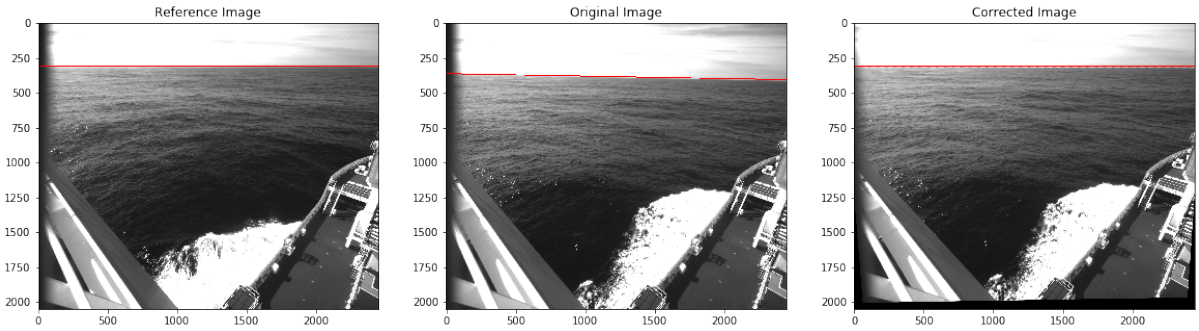


Figure 3.53: Images showing correction applied to data collected in July. The reference image (left), target image with a correctly detected horizon (centre) and the corrected target image (right).

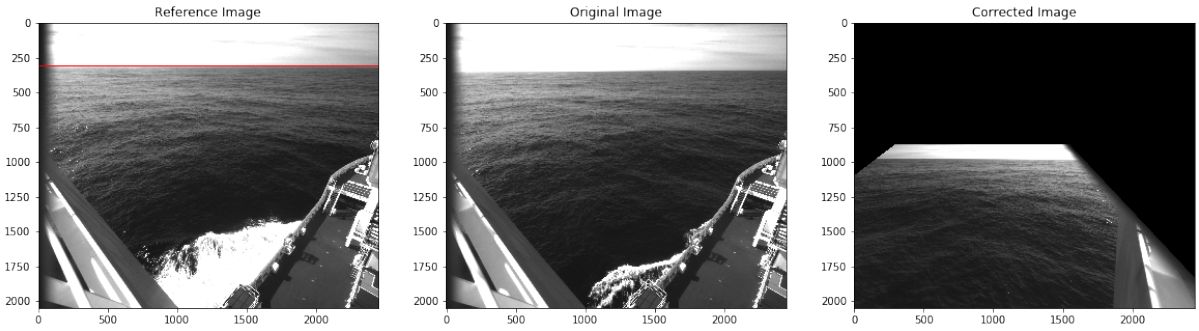


Figure 3.54: Images showing a correction incorrectly applied to data collected in July. The reference image (left), target image with an undetected horizon (centre) and the incorrectly corrected target image (right).

created data, the correction is harder to see, but visible when looking at the edges of the corrected image where it is clear that the image frame has been manipulated. Fig. 3.54 displays an example of what happens when the horizon is incorrectly detected or undetected in an image.

3.3.4.5 Automatic Correction Algorithm

After individually developing and testing detection, estimation and correction, an algorithm integrating all these steps was developed. Some results can be seen in Fig. 3.55, and the flow diagram for the algorithm in Fig. 3.56.

Through testing the algorithm's performance, it was found that applying the constraint for maximum change in orientation ironically resulted in a deterioration in performance, forcing false detections to be made. This is likely due to the fact that the ship's movement is difficult to quantify, as it can vary significantly over days and even hours depending on the conditions. This method of accuracy checking was, therefore, removed from the

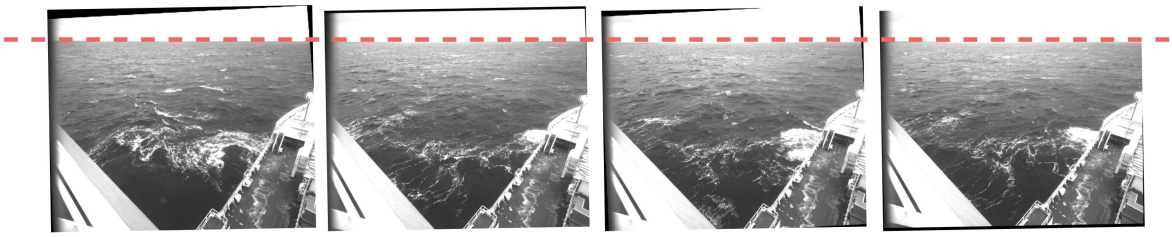


Figure 3.55: A series of images showing a sequence of corrected images with identical placement of the horizon (all images were rotated by the same amount to allow the horizon line to lie parallel to the page).

algorithm, and instead tighter constraints were placed on the location of the identified horizon. This meant restricting the polar coordinates found for the horizon line to be within certain parameters, as this is a more consistent and less sensitive constraint.

Another major problem encountered in testing was the changing light conditions in image sets, due to the sensitivity of the method to the contrast between sky and sea or ice. The changing light required responsive brightness and contrast adjustments to be applied to images for correct horizon detection, and this was therefore included in the algorithm.

The algorithm's performance is assessed and discussed in the Results chapter of this report.

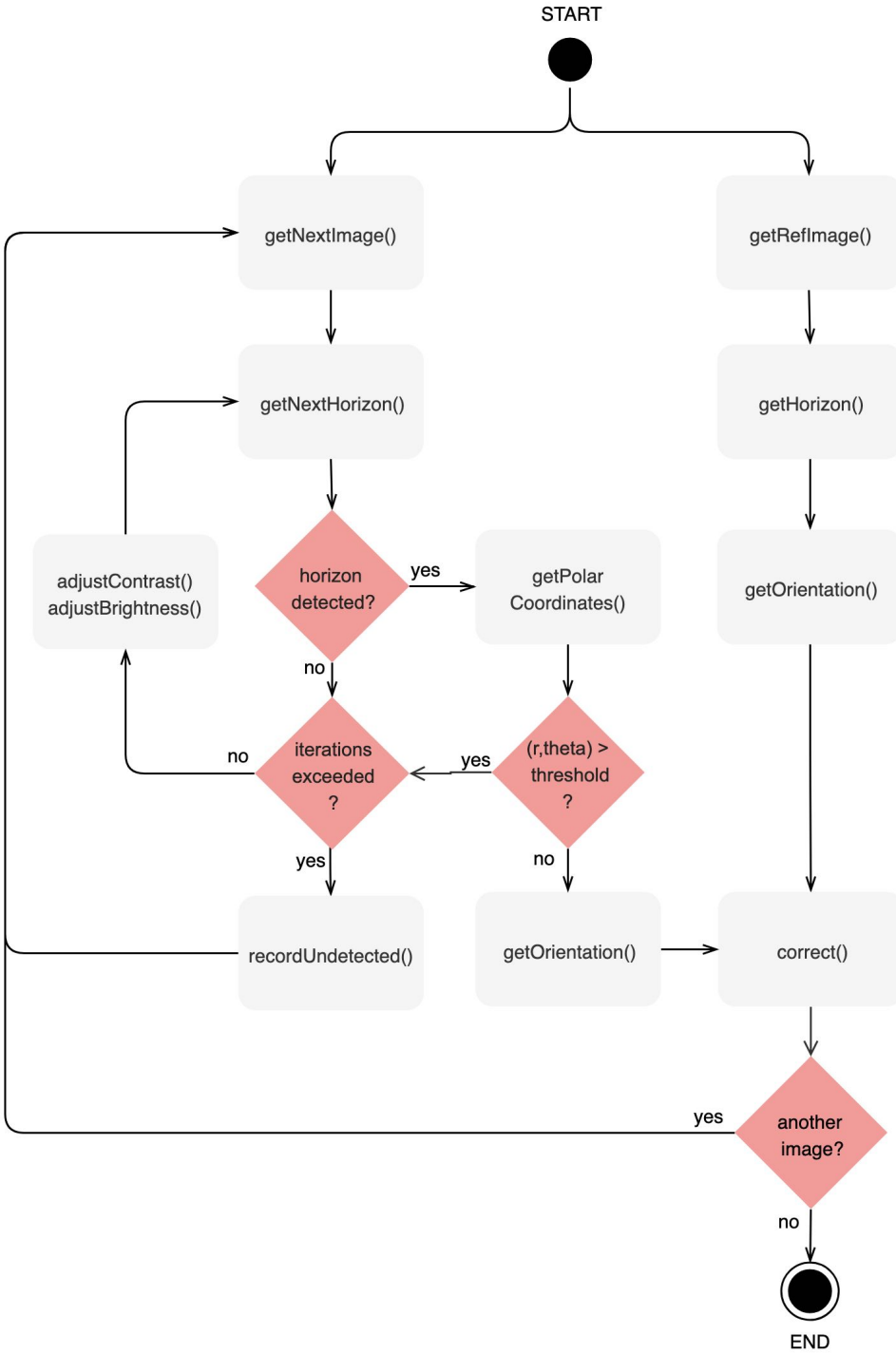


Figure 3.56: An activity diagram for the image correction algorithm.

Chapter 4

Results

The results pertaining to the developments in hardware, specifically the heaters and camera mounts, are not discussed, as they will only undergo proper testing during the October expedition.

4.1 Calibration

4.1.1 Cameras

The parameters obtained from calibration using Matlab's Camera Calibration Toolbox are shown in Table 4.1, and show that a reprojection error of less than 1 pixel was successfully achieved.

Using a script written in Python, these parameters, defining the camera matrix K and distortion coefficients, were used to undistort images obtained on the July cruise. Fig. 4.2 shows the results of undistortion using two different methods. The changes to the images are subtle as the distortion from the camera lenses is not very pronounced, but can be seen by observing the bottom right corners of the image frames.

	Camera 1	**Camera 2**
Focal length $(f_x; f_y)$	(1457,45; 1452,52)	(1468,12; 1466,01)
Optical centre $(c_x; c_y)$	(1230,86; 1030,48)	(1258,08; 1046,18)
Distortion coefficients $(k_1; k_2; p_1; p_2; k_3)$	(0,01244; -0,01056; 0,00073; 0,00206; 0,00)	(0,02797; -0,01591; 0,00506; 0,00337; 0,00)
Reprojection error (x;y)	(0,42142; 0,36073)	(0,44198; 0,36297)

Table 4.1: The intrinsic camera parameters and distortion coefficients obtained through calibration.

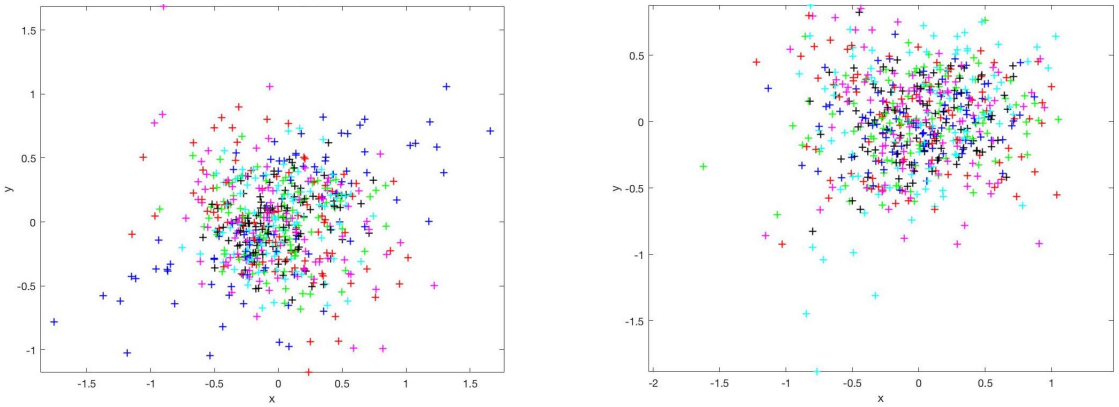


Figure 4.1: A plot showing the reprojection errors in pixels of camera 1 (left) and camera 2 (right). The different colours represent each image's reprojection errors with respect to their estimated grid points.

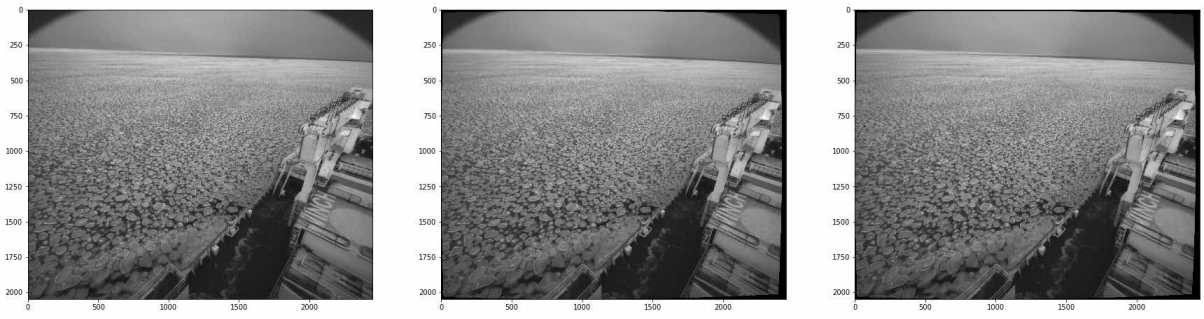


Figure 4.2: Images showing the results of undistortion using two methods. Original image (left), undistorted image using method one (centre) and using method two (right).

Since the distortion is minimal, the method used for removing the distortion is not important as they produce similar results. When comparing the pixel differences between

the two resulting images, it was found that only 0.03% of pixels differed. Fig. 4.3 shows the pixels that were different highlighted in green.

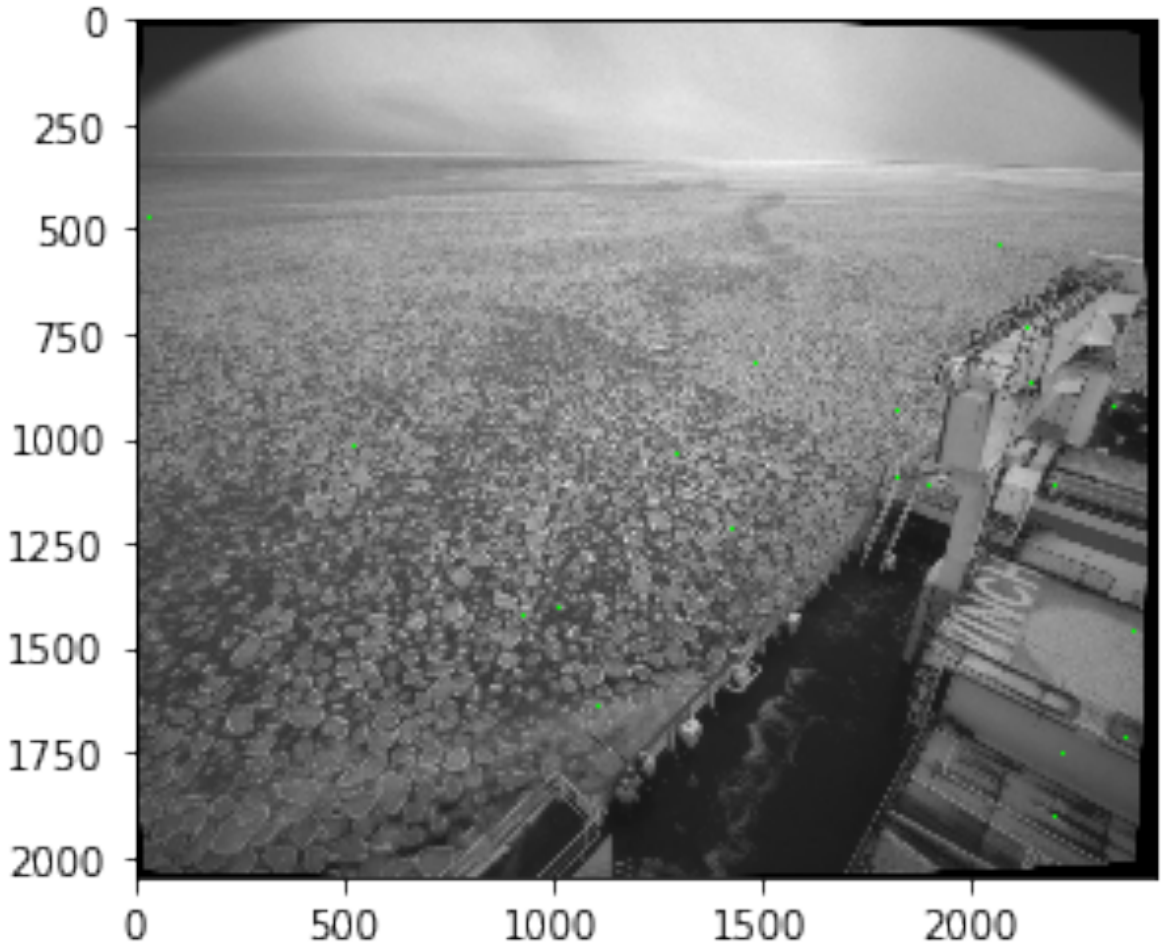


Figure 4.3: Image showing the differences between the results of the two undistortion methods. Pixel differences are highlighted in green.

4.1.2 IMU

The accuracy of IMU calibration and the error drift for extended periods of time was assessed in two ways. Firstly, the IMU was moved around and placed back into the same position, to assess whether it correctly gave readings indicating the same orientation at the beginning and end of acquisition (Fig. 4.4). Secondly, the roll and pitch were recorded from a static IMU for 24 hours, and the first and last 100 data points obtained were plotted and compared (Figs. 4.5 and 4.6).

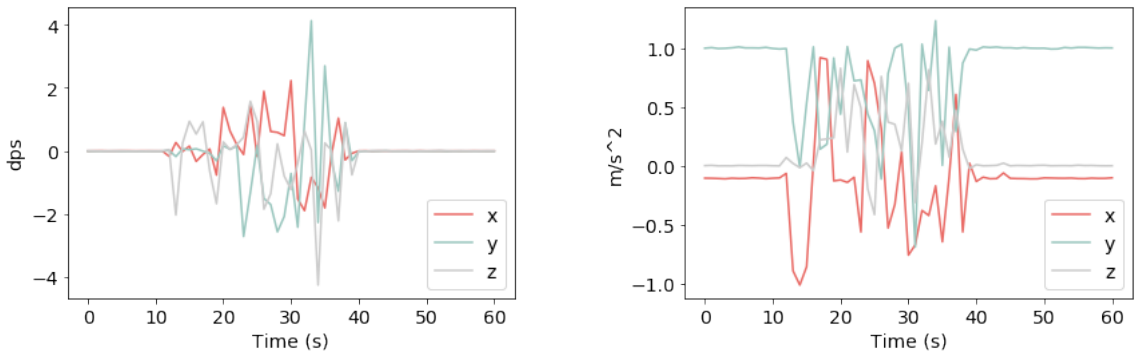


Figure 4.4: Plots showing the results of the movement test for the gyroscope (left) and accelerometer (right).

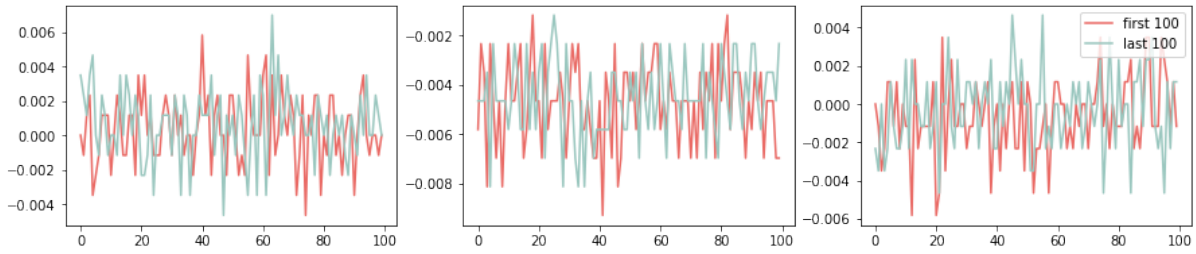


Figure 4.5: Plots showing the first and last 100 data points obtained for roll. X-axis (left), y-axis (centre) and z-axis (right).

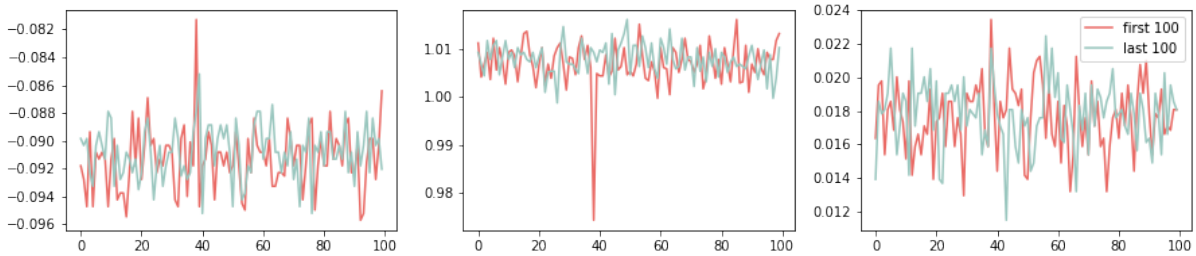


Figure 4.6: Plots showing the first and last 100 data points obtained for pitch. X-axis (left), y-axis (centre) and z-axis (right).

The results of the movement test confirmed that the IMU is successful in accurately reacquiring its position after significant movement. And the error drift over the 24 hour period was also shown to be inconsequential, as although there is sensor noise, the readings from the first and last 100 data points showed similar distributions (Figs. 4.7 and 4.8).

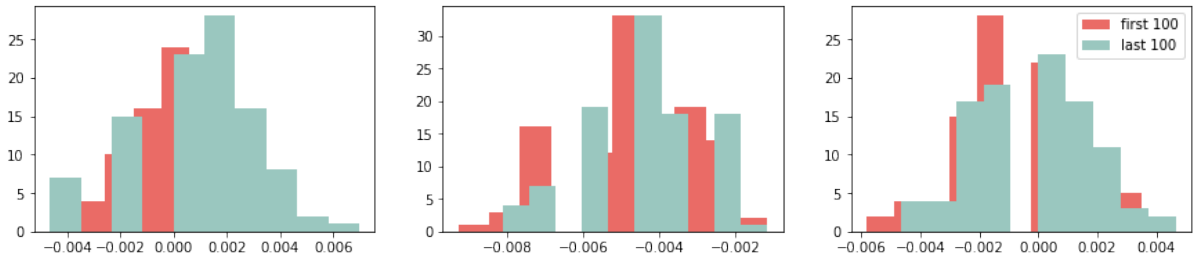


Figure 4.7: Histogram showing the distribution of the first and last 100 data points obtained for roll. X-axis (left), y-axis (centre) and z-axis (right).

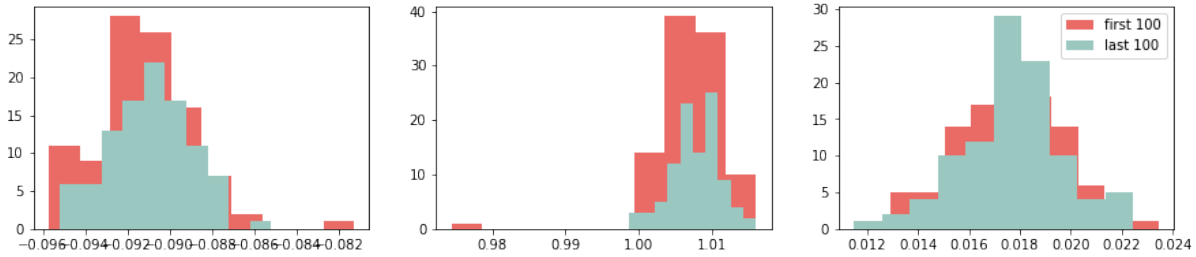


Figure 4.8: Histogram showing the distribution of the first and last 100 data points obtained for pitch. X-axis (left), y-axis (centre) and z-axis (right).

The accuracy of calibration will be further tested on the expedition, by comparing the orientation obtained before the ship departs and after it returns. Its position may be slightly different upon its return, but the roll and pitch should be the same.

4.2 Acquisition

4.2.1 Evaluation Procedure

A final 24-hour test was run in the lab, implementing all the developments detailed in the Methodology chapter. The equipment and experimental set up used is shown in Figs. 4.9 and 4.10. The cameras were again set up to take images of a digital clock running on the laptop.

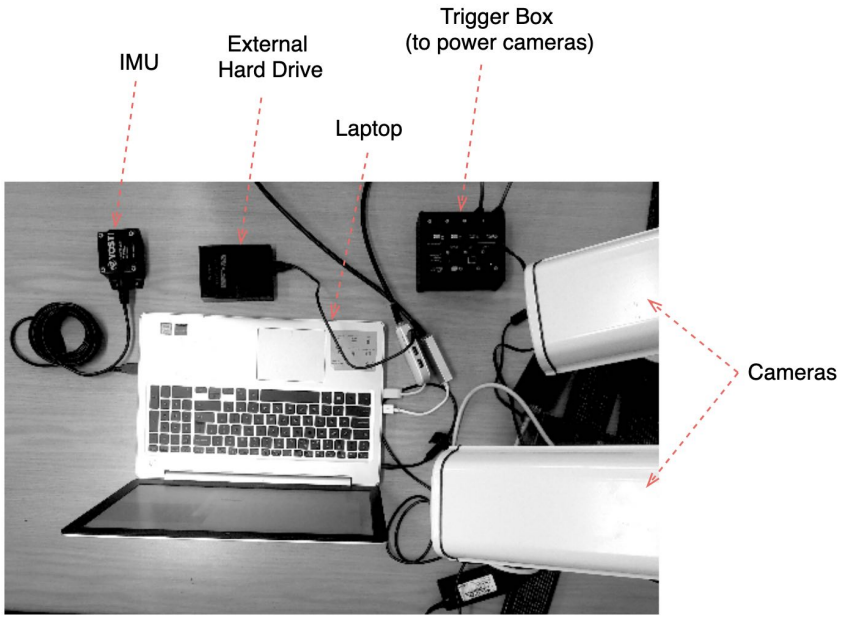


Figure 4.9: Image showing the equipment used in acquisition testing.

Figure 4.10: Images showing the experimental setup for acquisition testing. The cameras are taking images of a digital clock running on the laptop.

4.2.2 Results from Testing

The tables below display the results evaluating the error rate, the synchronisation and the time interval.

IMU data points acquired	86 400
Left camera images acquired	86 400
Right camera images acquired	86 400
Error rate	0%

Table 4.2: The error rate results from the final 24-hour acquisition test.

Image pairs assessed	864
Image pairs with matching clock time	864
Synchronisation error	0%

Table 4.3: The camera synchronisation results from the final 24-hour acquisition test.

Data pairs evaluated (IMU and left images)	86 400
Average time difference (ms)	30
Pairs with a time difference > 100 ms	3
Synchronisation error	0,003%

Table 4.4: The IMU-camera synchronisation results from the final 24-hour acquisition test.

Expected data points acquired	86 400
Actual data points acquired	86 400
Average time interval (s)	1
Interval errors (intervals $\neq 1 \pm 0.001$ s)	63
Time difference between first and last acquired data point (s)	86 399
Time drift (s)	0

Table 4.5: The IMU time interval results from the final 24-hour acquisition test.

Expected data points acquired	86 400
Actual data points acquired	86 400
Average time interval (s)	1
Interval errors (intervals $\neq 1 \pm 0,001$ s)	1926
Time difference between first and last acquired data point (s)	86398,91
Time drift (s)	0,09

Table 4.6: The left camera time interval results from the final 24-hour acquisition test.

4.2.3 Discussion of Results

The metrics to evaluate were the error rate, the synchronisation between sensors and the precision of the 1 s time interval. The first observation to make is that the system successfully obtained a data point for every second of acquisition — 86 400 data points from each sensor for 86 399 s of acquisition — and had an error rate of 0%. This in itself is an indication that the system was successful with regards to many of the required parameters. The synchronisation between cameras was to the millisecond, and between the IMU and cameras well within the required maximum timestamp difference of 100 ms. Finally, the average time interval between data capture, assessed using timestamps, was the required 1 s. Although there were time interval errors, with 1926 successive data points not within the specified interval time for the left camera, this is only a 0,02% error when considering all data points captured. There was a 0 s time drift overall for the IMU and 0,09 s for the cameras.

These results indicate that the system is ready for the expedition. In terms of robustness, it should be noted that although only a single 24-hour test was used for the final analysis, the multiple tests run whilst developing the methodology, providing similar results, indicate that the success of the final test is not a singular event but a repeatable outcome. This allows the reasonable conclusion to be drawn that the system is robust.

This conclusion, however, can only be extended to apply to the conditions in the lab. The question now is whether the system will be able to perform to the same standard in Antarctic conditions. Although the laptop is located in the Birder hut which is heated,

it is still significantly colder than the environment the tests were run in. Unfortunately, acquisition tests were unable to be run in the Mobile Polar Lab on campus, which would have given a good indication of performance on the ship. Therefore, it remains to be seen how temperature will affect the system.

One further uncertainty to be investigated is whether the synchronisation between the cameras and IMU is sufficient. A maximum 100 ms difference in acquisition time was required and achieved, however, the uncertainties around the actual acquisition time and recorded acquisition time of data, as well as whether 100 ms is a small enough interval to use IMU data for accurate image correction, will require further investigation once data is obtained from the ship.

4.3 Orientation Estimation and Correction

4.3.1 Evaluation Procedure

Tests were run with sample sets of 2 000 images. Each of the three samples were taken from different days of the cruise where weather and environmental conditions differed vastly. The first set was from a day in open water with clear skies, the second in a sea of smaller ice floes and darker overcast skies and the third in semi-consolidated ice with cloudy weather.

(a) Open water with clear skies.

(b) Surrounded by smaller ice floes and darker skies.

(c) Consolidated sea ice and overcast weather.

Figure 4.11: Images from the three sample datasets, used to test the performance of the algorithm.

The 2 000 images consisted of 1 000 each from left and right cameras, and account for an acquisition period of approximately 17 minutes. All testing was also performed on images where the horizon was visible, and therefore detectable.

4.3.2 Results from Testing

The overall results are detailed in Table 4.10, and figures below display some of the results from the three samples.

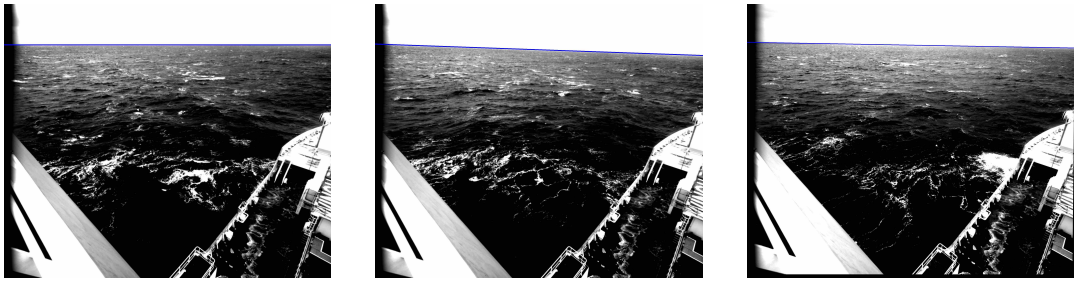


Figure 4.12: Images from sample one showing correctly detected horizons.

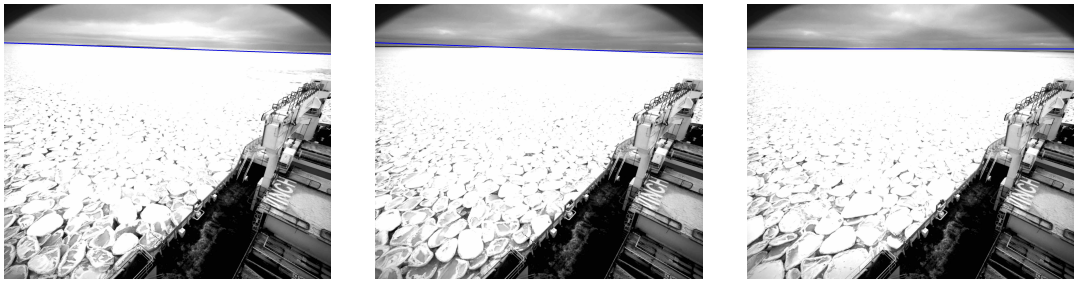


Figure 4.13: Images from sample two showing incorrectly detected horizons. The incorrect detections are due to the presence of open water very close to the actual horizon, which are then recognised as the most prominent line. Fig. 4.17 shows a bigger one of these images so that the badly detected horizon is easier to see.

Figure 4.14: Images from sample three showing correctly detected horizons.

	Sample 1	**Sample 2**	**Sample 3**
Acquisition start time	15h33	13h43	08h41
Weather	Clear skies	Overcast and grey skies	Cloudy with visible patches of sky
Environment	Open water	Small ice floes	Semi-consolidated ice sheets
Images assessed	2 000	2 000	2 000
Incorrectly detected horizons	24	827	320
Undetected horizons	36	0	0
Correctly detected horizons	1 940	1 173	1 680
Horizons correctly identified by responsively adjusting brightness and contrast	25	2	16
Success rate	97%	59%	84%

Table 4.7: The results of the horizon detection algorithm applied to three sample sets of 2 000 images.

As mentioned in the technical specifications, the aim to accurately obtain the ship's relative orientation cannot be properly tested until the system is mounted on board. As a preliminary measure of validation however, Figs. 4.15 and 4.16 show the obtained roll and pitch from the three sample datasets to comparatively display the orientation change in different conditions.

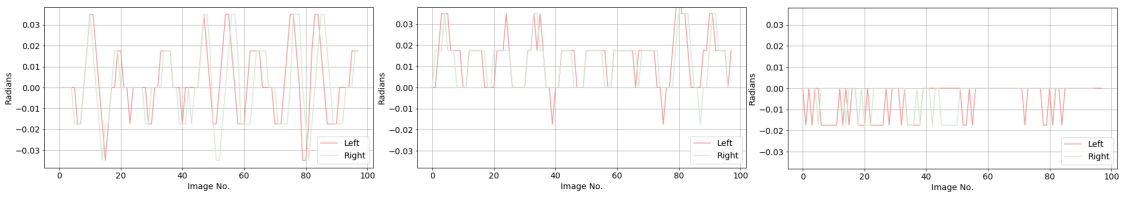


Figure 4.15: Plots showing roll obtained for 100 images from the three sample sets. Sample one (left) in open ocean, sample two (centre) in small ice floes and sample three (right) in semi-consolidated ice.

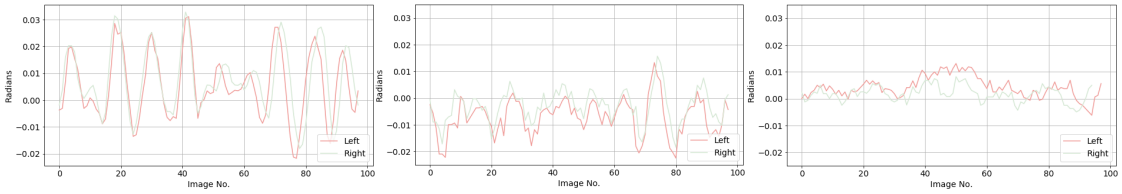


Figure 4.16: Plots showing pitch obtained for 100 images from the three sample sets. Sample one (left) in open ocean, sample two (centre) in small ice floes and sample three (right) in semi-consolidated ice.

To assess the precision of correction, the orientation of corrected images was compared to the orientation of the reference image. Corrected images should be corrected to have the exact same orientation as the reference image. The pixel-wise match of the horizon line drawn on corrected images was also compared to the reference horizon line, as again, the horizon lines should be identically positioned in both reference and target images. The results were as follows:

Images assessed	100
Images accurately corrected in terms of roll	94%
Maximum error in roll	0,017
Images accurately corrected in terms of pitch	94%
Maximum error in pitch	0,174
Average number of horizon line pixels matching the reference horizon line	70%

Table 4.8: The results from assessing the accuracy of correction of 100 corrected images.

4.3.3 Discussion of Results

The specifications for this subsystem were for the algorithm to correctly detect 90% of the horizons in three sample datasets, for the ship's relative orientation to be accurately estimated and for images to be precisely corrected so that their horizon matched the reference horizon.

Whilst the first and third sample sets showed promising results in horizon detection, the

algorithm struggled with the second set due to the strong lines from the presence of dark water amidst white ice by the horizon line (Fig. 4.17). This led to many of the images needing to be discarded due to incorrect detections.

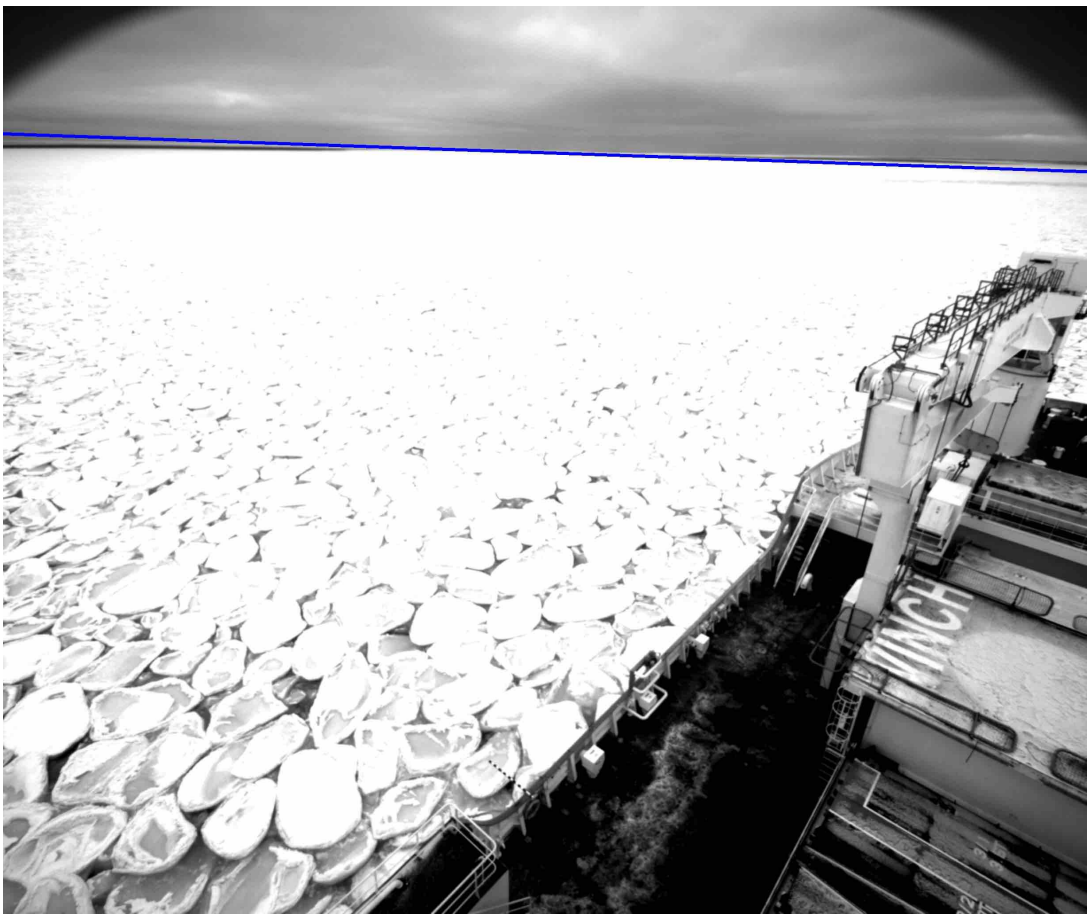


Figure 4.17: An image from sample two showing a badly detected horizon.

The difficulty in this algorithm clearly lies with the changing light and environmental conditions both within image sets and between image sets. Despite introducing a method to responsively alter the contrast and brightness, the initial settings as well as their increments had to be individually selected for every set based on some limited experimentation. This detracts value from the intended automatic nature of the algorithm. The adjustments to brightness and contrast also slow down the algorithm considerably, and from the results we can see that it only aided in the detection of a relatively small number of images.

Another problem is that unless the algorithm becomes more reliable, images will need to be manually inspected for correctness. For the purposes of these expeditions, where the number of images taken is in the hundred thousands, this is not really feasible. Therefore, the detection algorithm requires further development if it is to be used on images with a wider range of lighting and environmental conditions.

In terms of correctly estimating orientation, the roll and pitch plots above show the results we would expect. In open ocean the roll and pitch have the greatest variability, whilst in the consolidated ice field, where wave activity would be dampened by the ice sheet, they have the lowest. Although it is not currently possible to quantify, the plots imply that there is at least some degree of accuracy in the orientation estimations. Until corresponding IMU data is collected, we cannot evaluate this further.

The results analysing the accuracy of corrections presents positive results. The roll and pitch estimated from corrected images is almost always equal to that computed in the reference image, with an accuracy of 94%. And the percentage of matching pixels representing the horizon line displays a good level of precision at 70%. However, since the precision is not 100%, it seems important to understand how inaccuracies in the detected horizon will affect the estimated roll and pitch values.

To assess this, the uncertainty in roll and pitch were computed from different potential uncertainties in the horizon equation's polar coordinates (r, θ). Uncertainties were computed for $r \pm 1$ pixel and $\theta \pm 0,5°$, and for a bigger uncertainty of $r \pm 10$ pixels and $\theta \pm 5°$. Figs. 4.18 and 4.19 plot these uncertainties (red) alongside the correctly detected horizon (blue).

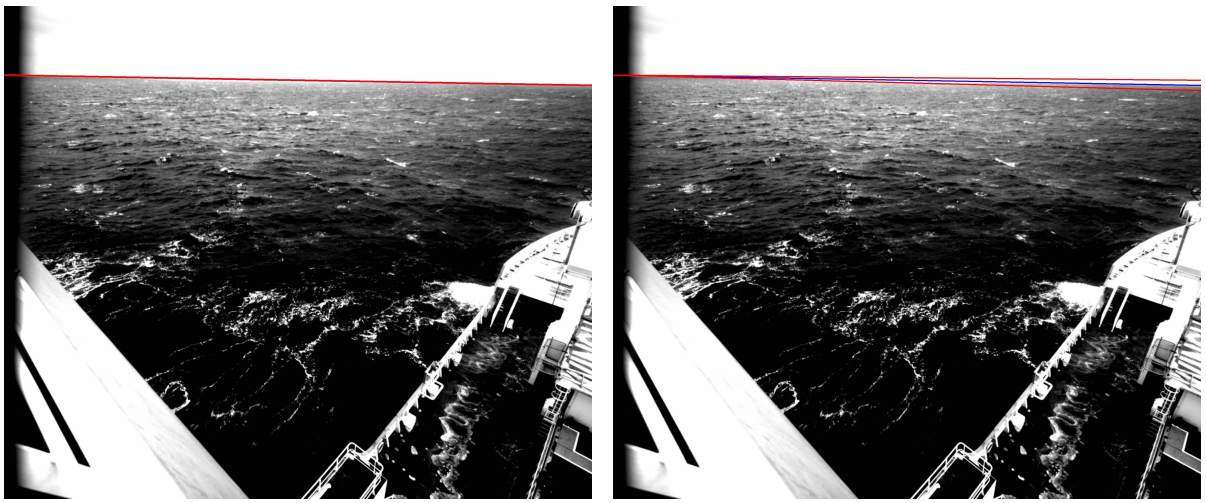


Figure 4.18: Detected horizon (blue) and horizons detected with uncertainties of $r \pm 1$ pixel and $\theta \pm 0.5°$ (red).

The following corresponding uncertainties in orientation were obtained:

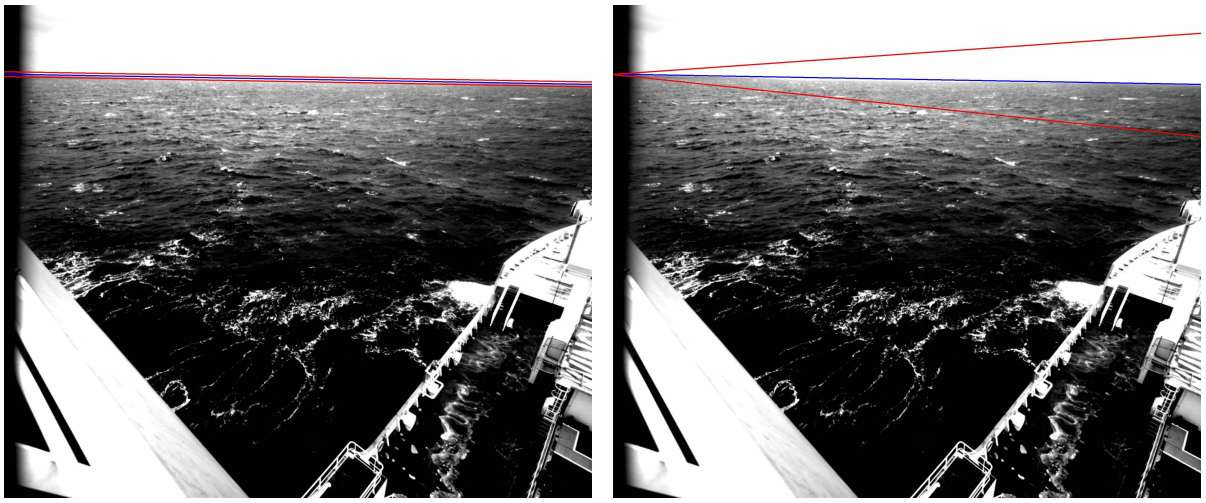


Figure 4.19: Detected horizon (blue) and horizons detected with uncertainties of $r \pm 10$ pixels and $\theta \pm 5°$ (red).

Uncertainty in r (pixels)	**Uncertainty in roll (rads)**	**Uncertainty in pitch (rads)**
± 1 pixel	0	$\pm 6,24 \times 10^{-4}$
± 10 pixel	0	$\pm 6,24 \times 10^{-3}$

Table 4.9: The corresponding uncertainty in orientation for uncertainties in r.

Uncertainty in θ (degrees)	**Uncertainty in roll (rads)**	**Uncertainty in pitch (rads)**
$\pm 0,5°$	$\pm 8,73 \times 10^{-3}$	$\pm 5,75 \times 10^{-3}$
$\pm 5°$	$\pm 8,73 \times 10^{-2}$	$\pm 5,99 \times 10^{-2}$

Table 4.10: The corresponding uncertainty in orientation for uncertainties in θ.

Uncertainties in θ are easy to see, and therefore horizons incorrectly detected in terms of θ will likely be discarded when visually inspected. The error in r however, is much more difficult to observe. Since errors in r do not lead to any errors in the computed roll, it explains why the plots showing left and right pitch always appeared less consistent than those for roll. An error of $r \pm 10$ pixels was experimentally found as the highest uncertainty that may lead to a misclassification of a correctly detected horizon through visual inspection, meaning that it is likely that detected horizons will be calculated with an error in pitch.

The question is now whether horizon detection or the IMU lead to a greater uncertainty in the calculated pitch. Using the distribution plot of pitch obtained for a static IMU (Fig. 4.8), IMU measurement noise, ignoring the offset, leads to a maximum uncertainty in pitch of approximately $\pm 0,009$ rads (along the y-axis), whilst an uncertainty in r of 10 pixels leads to an uncertainty in pitch of $\pm 0,006$ rads. These uncertainties are very similar, but allow us to conclude that if IMU functionality at the poles was not compromised and the IMU time drift error was not significant, using horizon detection would provide, if not better, similar results to the IMU, in terms of accuracy. This justifies having investigated and further developing horizon detection for image correction.

Chapter 5

Conclusions

5.1 End of Project Requirement Verification

General System Requirements

RG-1: **Fully satisfied** - The system should run in Linux.

RG-2: **Fully satisfied** - The system should be simple to use.
Note: The system is run with a single script that just needs to be started in the terminal. Unless the user wants to change the location data is saved to or the amount of time an acquisition session runs for, no changes need to be made to any of the settings or the script in between acquisition sessions. The code is well commented to enable these adjustments to be easily made.

RG-3: **Fully satisfied** - All code should be written in Python and use open source software.
Note: Camera calibration was performed in Matlab, but an alternative method using Python is detailed in Appendix B.

Hardware Requirements

RH-1: **Fully satisfied** - A working IMU must be available for the October expedition.

RH-2: **Fully satisfied** - The cameras need to withstand the Antarctic cold and must therefore be fitted with heaters that turn on when temperatures are less than $15°C$.

RH-3: **Partially satisfied** - The camera mounts need improvement to withstand the turbulent weather conditions and ship vibrations.
Note: This can only be fully tested once on the expedition.

Calibration Requirements

RC-1: **Fully satisfied** - The cameras should be calibrated prior to mounting on the ship, to obtain the intrinsic parameters with a reprojection error less than one pixel.

RC-2: **To be completed** - The IMU must be calibrated once aboard the ship when it is still docked, and be aligned with the ships heading for the duration of the cruise.

Acquisition Requirements

RA-1: **Fully satisfied** - The system should be able to run continuously for up to 24 hours.

RA-2: **Fully satisfied** - The system should have an average error rate of less than 0.01 (for every 100 seconds there is only one image pair or IMU data point not acquired).

RA-3: **Fully satisfied** - The cameras should be synchronised to take image pairs with a time difference of less than one millisecond.

RA-4: **Fully satisfied** - IMU data collected must be synchronised to each image pair, either during acquisition or in a post-processing step, with a time difference less than 100 milliseconds.

RA-5: **Fully satisfied** - Image pairs and IMU data should be acquired approximately every second, with an average interval error rate of less than 1 ms.

RA-6: **Fully satisfied** - Images and IMU data should be stored to an external hard drive and organised into subfolders detailing the acquisition start time. Corresponding

image pairs and IMU data points should be labelled so that correspondence is simple.

Orientation Estimation and Correction Requirements

RO-1: **Partially satisfied** - The code should automatically detect the horizon of a series of images with an accuracy greater than 90 %.
Note: The accuracy of the detection algorithm depends strongly on the environment and lighting conditions of the images. For ideal conditions of clear skies and open water, it fulfils the requirement. However, for cloudy skies and sea ice its performance is inadequate.

RO-2: **Partially satisfied** - The code should be able to use the detected horizon to estimate the ship's relative orientation.
Note: This requirement can only be properly tested once corresponding IMU data is available from the ship.

RO-3: **Partially satisfied** - The code should correct images so that their horizon is identically positioned to the horizon of the reference image.
Note: The accuracy of corrected images is between 70 % - 94 %, depending on the assessment criteria, and therefore leaves room for improvement.

5.2 Conclusions from Testing

The thorough testing that went into the stereo system has shown that it will perform — if not without fault — at a much higher level than demonstrated in July. It successfully fulfils all of the required specifications. The system can comfortably operate for up to 24 hours without lag. Synchronisation between the cameras is precise and has shown that a hardware trigger is not a necessity. The only remaining uncertainty is with regards to the system's operation in colder conditions and the sufficiency of the IMU-camera synchronisation.

The use of horizon detection for motion compensation has potential, but requires further development before it can be reliably implemented. However, its usefulness in assessing the accuracy of the IMU at the poles and for extended periods of time should be well noted. Comparing the manually inspected orientation obtained from even a small number

of images and the corresponding IMU data points will provide an indication of the IMU's accuracy. The horizon detection method can also be used in post-processing to recalibrate the IMU if it experiences severe error drift over the long duration of its operation.

5.3 General Conclusions

The application of engineering systems in the field often uncovers unexpected difficulties that may not be readily apparent through testing undertaken in controlled lab conditions. This was experienced in the first deployment of the system in July. The successful identification of these problems combined with the application of new methods to account for them, supports the prediction that the system will perform far better on the October expedition. However, this experience has demonstrated that challenging conditions can ultimately result in the emergence of new problems. It is expected that new issues will arise, but a greater understanding of the system, in terms of both its software and hardware, will allow the rectification of any potential issues during the expedition without compromising data collection to any significant degree.

Whilst the majority of this project's scope was dictated by what was practically required for the October expedition, horizon detection was investigated in an attempt to ensure that there was an additional theoretical-based element to this project, instead of the purely practical approach necessitated by the development of the stereo system itself.

Although sensor calibration and synchronisation employs theoretical considerations, it often relates to new hardware implementations, such as using FPGAs and other boards. It was decided for this project to limit the system to equipment already obtained, as the existing equipment is very expensive, and it would therefore be wasteful not to thoroughly test its capability before resorting to acquiring new hardware.

There are a number of outstanding issues not covered in this report, such as whether the horizon detection method can accurately estimate the ship's orientation, and whether the IMU-camera synchronisation is sufficient for image correction. This may pre-empt a question as to why these aspects were included in the project scope when it would be impossible to evaluate them until the system was tested on the ship, after the project deadline has elapsed. Incorporating these aspects into the project was important as they are considerations that need to be resolved in order to further the development of the system and excluding them would reduce the potential for further research to build directly on this work.

Detailed instructions for the system's software configuration, installation and operation can be found in Appendix B.

Chapter 6

Recommendations

Calibration

Cameras

The most challenging aspect of calibration is obtaining images of a completely flattened checkerboard pattern because the parameters used for calibration are sensitive to minute distortions. To account for this sensitivity it is advisable to calibrate the system using a checkerboard pattern printed on a rigid medium.

IMU

With regards to the IMU, to assess potential error drift, tests should be undertaken that compare IMU data obtained immediately prior to departure with data obtained immediately upon the ship's return to the dock. This method is applicable because although the ship may be docked in a different position, the orientation with respect to roll and pitch should be the same.

Acquisition

Hardware Trigger

It is recommended that the use of the hardware trigger for acquisition in Linux should undergo further investigation. Although the software trigger has proven sufficient, it would be preferable to utilise a hardware trigger because it could significantly improve overall accuracy of camera synchronisation and the time interval between successive data points. According to communications with the manufacturers, an Arduino board can be used to control the hardware trigger through Linux.

IMU-Camera Synchronisation

It is recommended that an evaluation be conducted on the accuracy required for ship-based acquisition. While this project determined a 100 ms difference in acquisition time as acceptable for the ship-based system, that is an estimate that is assumed to be sufficient for the purpose of the research data. It is expected that the use of horizon detection for orientation estimation will assist in determining whether this assumption is accurate. Furthermore, it is also recommended that if the IMU displays error drift over the duration of the expedition, horizon detection be employed as a method of recalibrating IMU data at certain intervals.

Orientation Estimation and Correction

Detection

In terms of horizon detection, it is recommended that investigations be undertaken on alternative methods to Canny Edge Detection and the Hough Transform. Segmentation, in which sea and sky are separated, is an option that could be implemented. A machine learning approach may also remove the limitations experienced with the algorithm when applied to different environmental and lighting conditions.

Estimation

For orientation estimation, it is recommended that horizon detection be applied to images obtained from the October expedition in order to determine the ships orientation. Data obtained through this process could then be compared to the corresponding IMU data. Assuming that the IMU data is correct, this comparison could then be used to assess the accuracy of horizon detection for orientation estimation.

Bibliography

[1] Past Events, *Open Day SA Agulhas II*, BCCSA, Jan. 2013. [Online]. Available: https://belgianchambersa.co.za/news/past-events/24-june-2017-open-day-sa-agulhas-ii/, [Accessed: 27 Jul. 2019].

[2] Q. Zhang and R. Skjetne, *Sea Ice Image Processing with MATLAB*. FL: CRC Press, 2018.

[3] J. Rabault, et al., "An Open Source, Versatile, Affordable Waves in Ice Instrument for Scientific Measurements in the Polar Regions," *Atmospheric and Oceanic Physics*, vol. 1, 2019.

[4] B. Weissling, et al., "EISCAM - Digital image acquisition and processing for sea ice parameters from ships, " *Cold Regions Science and Technology*, vol. 57, no. 1, 2009, pp. 49-60.

[5] A. Campbell, "Stereo Imaging Techniques for Three-Dimensional Surface Wave Measurement: Applications and Improvements," M.Sc. thesis, University of Wisconsin-Madison, WI, 2014.

[6] A. Benetazzo, et al., "A Variational Wave Acquisition Stereo System for the 3-D Reconstruction of Oceanic Sea States," in *Proc. of the ASME 2011 30th International Conference on Ocean, Offshore and Arctic Engineering*, Ocean Engineering: Rotterdam, 2011.

[7] F. Bergamasco, et al., "Multi-view horizon-driven sea plane estimation for stereo wave imaging on moving vessels," *Computers and Geosciences*, vol. 95, 2016, pp. 105-117.

[8] F. Bergamasco, et al., "WASS: An open-source pipeline for 3D stereo reconstruction of ocean waves," *Computers and Geosciences*, vol. 107, 2017, pp. 28-36.

[9] MV, Rohith et al. "Modified Region Growing for Stereo of Slant and Textureless Surfaces," *ISVC*, 2010.

[10] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *International Journal of Computer Vision*, vol. 47, no. 7, 2002.

[11] P. Munro and A. Gerdelan, "Stereo Vision Computer Depth Perception," 2006.

[12] N. Smolyanskiy, A. Kamenev and S. Birchfield, "On the Importance of Stereo for Accurate Depth Estimation: An Efficient Semi-Supervised Deep Neural Network Approach," in *Proc. Conf. on Computer Vision and Pattern Recognition*, IEEE/CVF, Jun. 2018.

[13] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice-Hall, 1998.

[14] A. Fusiello, "Tutorial on Rectification of Stereo Images," 1999.

[15] OpenCV, *Camera Calibration and 3D Reconstruction*, OpenCV, Dec. 2015. [Online]. Available: https://docs.opencv.org/3.1.0/d9/db7/tutorial_py_table_of_contents_calib3d.html, [Accessed: 27 Jul. 2019].

[16] I. Markelic, "Tutorial on Homographies".

[17] S. Pizzo et al., "Roll and Pitch Estimation Using Visual Horizon Recognition," in *Proc. Conf. on Augmented and Virtual Reality*, AVR, Dec. 2014.

[18] M. Kok, J. Hol and T. Schon, "Using Inertial Sensors for Position and Orientation Estimation," *Foundations and Trends in Signal Processing*, vol. 11, no. 1-2, 2017, pp. 1-153.

[19] *3-Space Sensors*, Yost Labs. [Online]. Avaliable: https://yostlabs.com/3-space-sensors/ [Accessed: 15 Sep. 2019].

[20] M. Schwendeman and J. Thomson, "A Horizon-Tracking Method for Shipboard Video Stabilization and Rectification," *Journal of Atmospheric and Oceanic Technology*, vol. 32, 2014, pp. 164-176.

[21] Ship Year, *Video: STX SA Agulhas II Documentary*, Ship Year, Jan. 2013. [Online]. Available: http://www.shipyearonline.co.za/articles/documentary-stx-sa-agulhas-2-4332.html, [Accessed: 27 Jul. 2019].

[22] J. Bouguet, *Camera Calibration Toolbox for Matlab*, Oct. 2015. [Online]. Avaliable: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html#examples [Accessed: 13 Sep. 2019].

[23] J. Snell, *Emissivity: Understanding How it Affects Your Thermal Images*, Thermal Imaging Blog, Sep. 2010. [Online]. Available: http://thermal-imaging-blog.com/index.php/2010/09/01/emissivity-understanding-how-it-affects-your-thermal-images/#.XZS3oFKB27M, [Accessed: 2 Oct. 2019].

[24] S. Mattoccia, Class Lecture, Topic:"Stereo Vision: Algorithms and Applications." Department of Computer Science, University of Bologna, Jan. 2013.

[25] G. Somanath, MV. Rohith and C. Geiger, "Reconstruction of snow and ice surfaces using multiple view vision techniques", in *Proc. 65th Eastern Snow Conf.*, Vermont, 2008.

[26] D. Scharstein and H. Hirschmller, "Evaluation of Cost Functions for Stereo Matching, " in *Proc. Conf. on Computer Vision and Pattern Recognition*, Minneapolis: IEEE, 2007.

[27] MV, Rohith et al. "Stereo analysis of low textured regions with application towards sea-ice reconstruction," in *Proc. Conf. Image Processing, Computer Vision, & Pattern Recognition*, Las Vegas: IPCV, 2009.

[28] MV, Rohith et al. "Towards estimation of dense disparities from stereo images containing large textureless regions," in *Proc. 19th ICPR Conf. Pattern Recognition*, IEEE, 2008.

[29] D. Scharstein, R. Szeliski and H. Hirschmller, *Middlebury Stereo Vision Page*. [Online]. Available: vision.middlebury.edu/stereo/ [Accessed: 3 Aug. 2019].

Appendix A

Stereo Vision

The following section gives an overview of the stages of stereo vision not elaborated on in the Literature Review.

Rectification

The calibration procedure for rectification was discussed in the Literature Review chapter. Proceeding this, the following information is also relevant to rectification.

Epipolar Geometry

The real argument for rectification is only understood when we have an understanding of epipolar geometry. It explains how the correspondence problem can be simplified and is the fundamental geometry of stereo vision, as expressed in Fig. A.1.

Epipolar geometry dictates that each point on a reference image plane will correspond to some line on a target image plane. So instead of locating corresponding pixels by searching an entire image, we can constrain the search to a single line [13].

Epipolar line: This constrains the possible correspondences between a pixel on the reference image to a line of pixels on the target image.

Epipole: This defines the position on one image plane where the second camera centre would be seen (if the image plane was extended far enough, as in most cases it will not

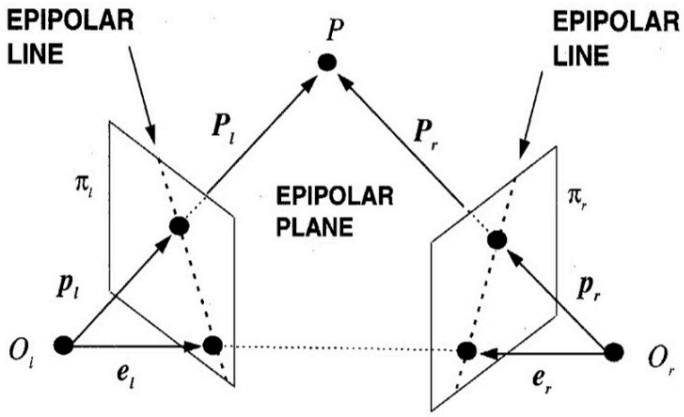


Figure A.1: A diagram showing the epipolar geometry of a stereo system [13].

actually be visible). All epipolar lines on an image plane intersect at this epipole.

Epipolar plane: This is the plane connecting the epipoles on both image planes and the point in 3D space being assessed.

The primary goal of rectification is to align the two image planes so that the epipolar lines become collinear and horizontal, with pixel (x_r, y_r) in the reference image corresponding to a pixel along the horizontal line $y_t = y_r$ in the target image. This simplifies correspondence to a 1D search problem along the x-axis.

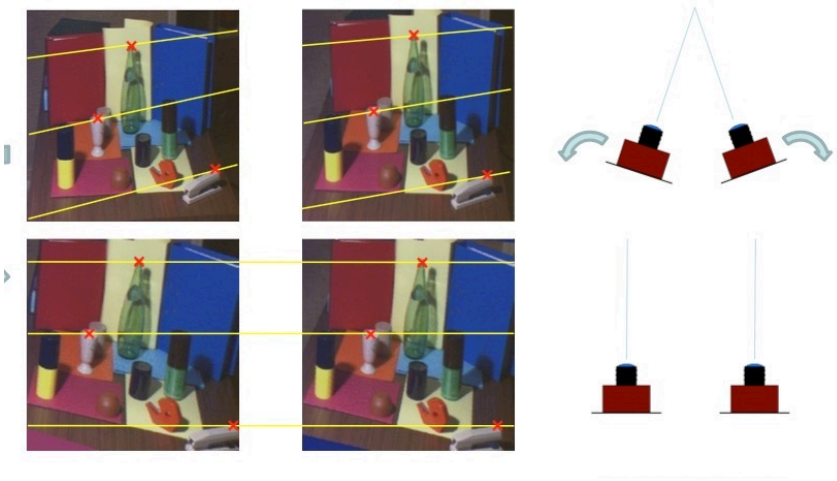


Figure A.2: An image showing the rectification step in which images are aligned so that corresponding pixels lie along the same horizontal lines [24].

The Rectifying Transformation

In order to rectify the stereo images, we need to define two new perspective matrices, that will preserve the optical centres but project the two image planes parallel to the baseline and each other [14]. The new **P** matrices are computed using the existing **P**

matrices as a starting point, and solving a system of linear equations which are defined by the constraints of both rectification and those needed to obtain a unique solution [14]. Once the new **P** matrices have been found, we can define the transformation matrix **T** to apply to the images that will result in rectification.

Correspondence

Correspondence is essentially a search problem, where given a point or element in the reference image, one must find the corresponding point or element in the target image. Also referred to as stereo matching, it is considered the most important and challenging step in stereo vision.

Two decisions need to be made, namely, the image element or feature to match between images, and how to measure their similarity [13]. The algorithms can be classified into two general classes, window-based and feature-based [10].

Window-based methods employ windows of fixed or variable size to match pixels from the two images based on their intensities. The similarity between windows or matching cost is maximised using either normalized cross-correlation or the sum of squared differences [13]. Correlation methods are easier to implement, but do not perform well on lowly textured images and are inadequate for image pairs from very different viewpoints [13]. Larger windows will give smoother disparity maps, but result in a loss of detail, whereas smaller windows preserve detail but produce noisy disparity maps.

Feature-based methods solve correspondence for a sparse set of features by finding feature descriptors, and corresponding them through minimising distance or using binary matching costs [13], [10]. Features could include edge points, lines, corners and even segments, and the best methods are determined by the type of objects you are imaging [13],[27]. Feature-based methods perform well when prior information is known of the scene being imaged, and have superior performance when there are changes in light intensity across image pairs. However, they require an additional computation for identifying features, and generally produce sparser disparity maps than correlation methods [13].

The correspondence algorithm selected needs to account for occlusions, the pixels that appear in only one of the two images, as well as spurious matches, falsely matched pairs due to noise [13]. These two potential problems can be avoided by employing epipolar constraints and left-right consistency constraints [13].

Finding correspondence in images of sea ice has to take into account that there will not be much texture or color variation to rely on [25], and therefore feature-based methods are a better starting point.

Reconstruction

The primary goal of reconstruction is to use the solution of the correspondence problem to compute the disparity map, from which the depth of individual pixels in 3D space can be found through triangulation.

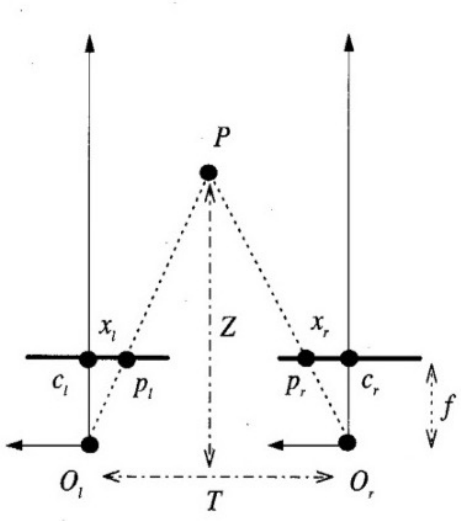


Figure A.3: An image showing how depth is calculated from the disparity of corresponding pixels [13].

Disparity was a term first used in human vision literature to describe the difference in position of a point in 3D space when viewed by the left and right eyes [10]. In Computer Vision, this becomes the difference in position of corresponding pixels on the two image planes. Using a rectified stereo pair, this difference is reduced to a horizontal displacement, along only one axis.

The accuracy of 3D reconstruction from stereo pairs is dependent on whether the intrinsic and extrinsic parameters are known. We will deal with the case where they are known, from which it is possible to obtain absolute coordinates [13].

The disparity map found can be characterised as either sparse, when only a small set of correspondences are found between pixels, or dense, when the majority of pixels are matched and thus disparity is found for all pixels [10]. If only sparse correspondences are obtained, generally some interpolation step for disparity refinement will be integrated

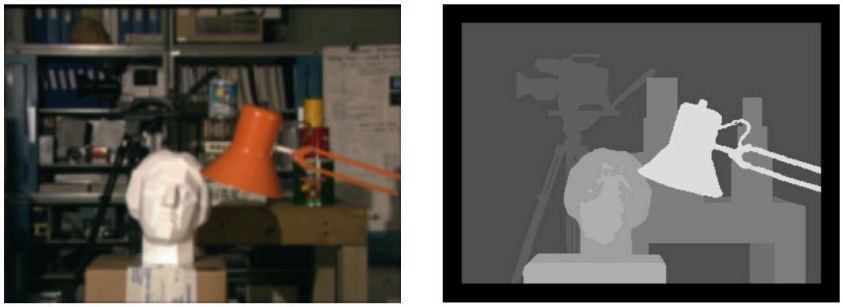


Figure A.4: Images showing an image and its disparity map. Original image (left) and disparity map (right).

into reconstruction.

Determining a pixels location in 3D space is done by triangulation, as seen in Fig. A.3. We can solve for similar triangles to get

$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z} \tag{A.1}$$

Solving for Z leaves us with

$$Z = f\frac{b}{d} \tag{A.2}$$

where the disparity, $d = x_r - x_l$.

Stereo Matching Algorithms

The majority of stereo vision algorithms rely on the color or intensity of pixels to reconstruct 3D models. This poses a particular challenge in the field of stereo vision for sea ice topography, due to the large areas of low colour variation [25] and radiometric differences between the the two images caused by slightly different camera settings, vignetting, image noise or non-Lambertian surfaces, which affect image similarity because of the difference in reflected light from differing angles [26]. This niche application of stereo vision has not been widely studied [25], however, recent papers have explored techniques to improve stereo matching for sea ice images, and their algorithms and results will be briefly mentioned.

Stereo matching algorithms are applied after the calibration problem has been solved to rectified images, and generally incorporate both the correspondence and reconstruction problem. The numerous stereo algorithms available are designed based on the environment being imaged, and present trade-offs in speed, accuracy and viability [6].

Stereo algorithms are generally defined as local or global [10]. Local algorithms are similar to the window-based correspondence method mentioned previously, where disparity is computed using intensity values within a finite window [10]. Global algorithms approach disparity computation as an optimization problem, and try to minimize a global cost function. There are also semi-global algorithms, that do not make explicit use of a global function to minimize, but mimic iterative optimization algorithms [10].

A taxonomy of stereo matching algorithms is presented in [10], and identifies that the majority of stereo matching algorithms perform some or all of the following four steps:

1. **Matching cost computation**
   SSD, SAD and NCC are the most commonly used matching costs. Others include the binary matching of features such as edges, however, these do not produce dense disparity maps and non-parametric measures such as rank and census transforms.

2. **Cost aggregation**
   Generally costs are aggregated by summing or averaging cost over some support region. The support regions are windows that could be finite, multiple with different anchor points, shiftable, adaptively sized or based on connected components.

3. **Optimization**
   Local algorithms usually use a winner-take-all (WTA) approach, assigning the minimum cost disparity value to each pixel. Global methods use their energy-minimization approach in this step. There are multiple algorithms that are used to find the minimum such as graph-cut or finding the minimum-cost path through corresponding scanlines.

4. **Disparity refinement**
   This step is used when image based rendering of the disparity map is required and aims to spurious mismatches and fill holes left by occlusions. Sub-pixel refinement is performed using iterative gradient descent, median filters, cross-checking, left-right consistency checks (bidirectional matching), surface fitting or interpolation.

Stereo Algorithms for Sea Ice Topography

As mentioned previously, stereo algorithms generally do not perform well in areas of low texture or colour variation. Three papers, all by similar authors, have covered various methods for stereo vision applied specifically to sea ice topography, and particularly try to solve the problems associated with it.

In [28], they identified that images with lowly textured regions could benefit from an approach where disparity for different regions is computed using different techniques. They first segment images into three regions, unambiguous, occluded but textured and low color variation (LCV) regions. Disparity is computed for unambiguous regions, and then interpolated into the other two regions using a finite element based diffusion solver. They concluded that their method of disparity diffusion performed better than other techniques in estimating disparity for LCV regions, however, misclassified some textured regions with reliable disparity estimates as LCV regions.

In [27] they propose a novel approach that combines entropy based filtering for segmentation and two levels of belief propagation. They first prefilter the image using maximum entropy based histogram equalization, to enhance the difference in the colour sections of the image. They then perform graph based segmentation. This allows them to perform disparity estimations based on segments, rather than on pixel windows, which they implement in two stages, first using loopy belief propagation, and then belief propagation minimization of a cost function for disparity refinement.

An important observation made in [27] is that using the entropy filtered, graph-based method for segmentation was successful in following depth discontinuities, but that it may be useful to perform segmentation using a variety of algorithms, such as mean-shift and graph-based, and evaluate disparity using this combination.

In [9], they propose a first-best region growing algorithm for finding dense disparity. Their algorithm uses Harris detectors for feature detection, and normalized cross correlation to correspond them. These sparse correspondences are then grown into regions that do not enter textureless areas, and interpolates matches for the large planar surfaces in the image by solving a minimization problem [9]. They concluded that their algorithm performed well, but suggest that future work should combine region growing with segmentation methods to estimate a more robust disparity.

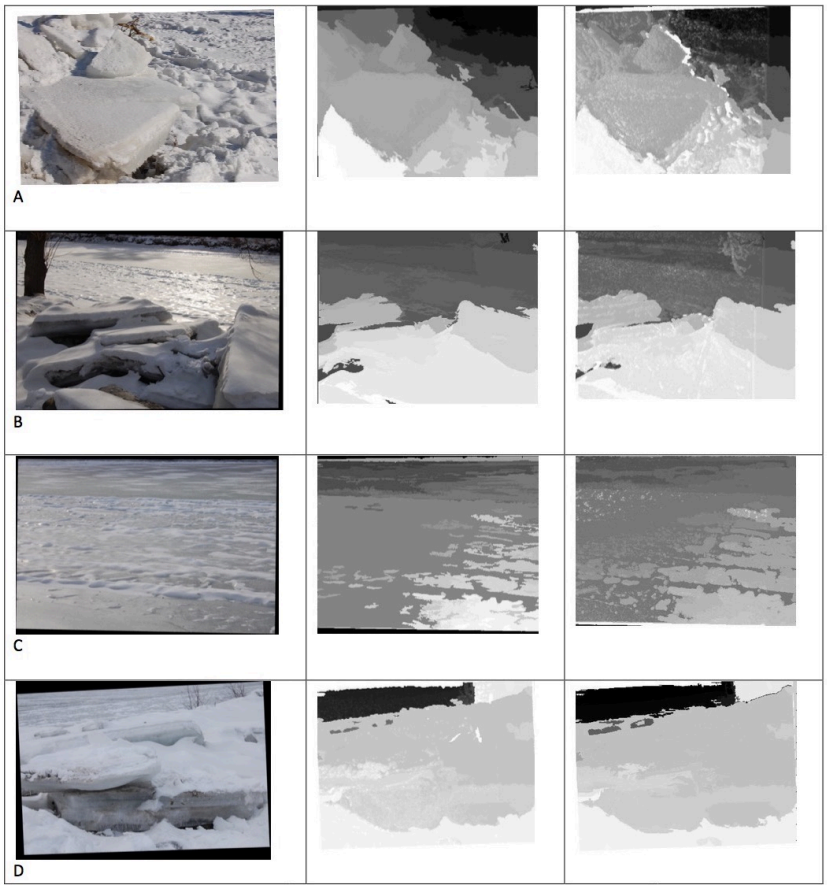


Figure A.5: Images showing the results of the stereo matching algorithm proposed in [27] when applied to ice and snow covered surfaces [27].

Evaluation Metrics

The Middlebury stereo vision site provides datasets, with ground truth data, and a framework for benchmarking stereo algorithms. They developed disparity accuracy measures to evaluate performance with respect to ground truth data [29].

These measures are computed for the whole image as well as specific regions to assess how algorithms handle typical problematic areas of stereo vision [10]. These regions are:

- Textureless regions ($\mathcal{T}$)
- Occluded regions ($\mathcal{O}$)
- Depth discontinuity regions ($\mathcal{D}$)

The measures used are:

1. The RMS error between computed disparity $d_C(x, y)$ and the ground truth $d_T(x, y)$

$$R = (\frac{1}{N}\Sigma_{(x,y)}|d_c(x, y) - d_T(x, y)|^2)^{\frac{1}{2}} \tag{A.3}$$

   where N is the total number of pixels in the region being assessed.

2. The percentage of Bad Matched Pixels B

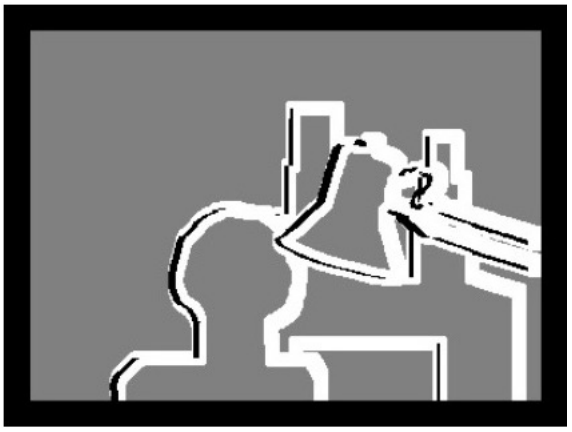
$$B = \frac{1}{N}\Sigma_{(x,y)}(|d_c(x, y) - d_T(x, y)| > \delta_d) \tag{A.4}$$

   where $\delta_d = 0.5, 1, 2, 4$ is the disparity error tolerance or threshold

(a) Original image

(b) Ground truth disparity map

(c) Textureless (white) and occluded (black) regions



(d) Depth discontinuity (white) and occluded (black) regions

Figure A.6: Images showing the evaluated regions on the Middlebury Stereo Vision site.

Appendix B

Stereo System Manual

The following section provides detailed instructions for setting up, installing and operating the stereo system. All the relevant manuals, datasheets and code can be found in this Google Drive folder.

Laptop

The system has been designed to run on Linux, as this presents the best option in terms of accessibility and speed. The current laptop is partitioned with both Linux and Windows, which is useful as the calibration software for the IMU is not compatible with Linux. The system can function on an alternative operating system if necessary, however, as the acquisition code uses a software development kit (SDK) that is designed for Linux, different applications will have to be used to run the system and performance will not be to the same standard. Instructions for Windows therefore branch off for certain aspects of installation.

Cameras

Connecting to the Cameras

The cameras have two connections each, one for power and one for communication. The power cable is connected to the hardware trigger, which must also be connected to mains. Communication with the Laptop is via Ethernet. Currently, Ethernet to USB adapters

are used for both cameras as the laptop's Ethernet port is not working. Note that the cameras need to be connected to separate ports — this only applies if you are using a multi-USB adapter, in which case, both cameras cannot be connected to the same adapter.

The cameras have been configured to fixed IP addresses. The details of setting up the right camera is shown in Fig. B.1. Configuration for the left camera is the same, with the exception of the IP address which is 192.168.5.1.

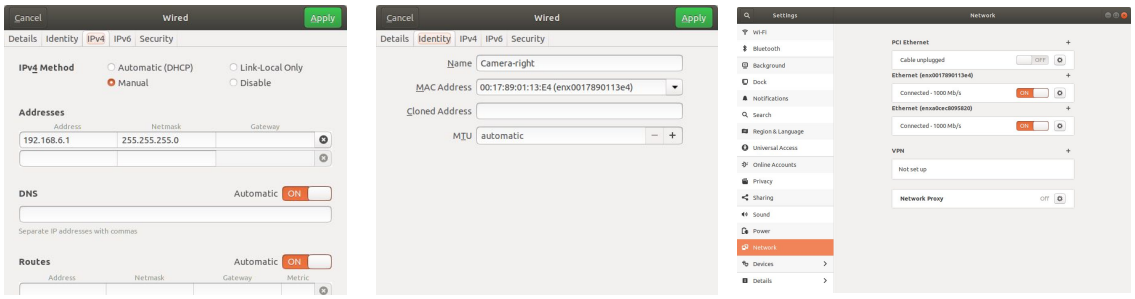


Figure B.1: Images showing the ethernet configuration on Linux for the right camera. Configure the IPv4 settings (left), ensure all details are correct (centre) and check it is connected (right).

The IP addresses and connection settings of the cameras can be changed in Windows, using the GigECam IP Config, which can be downloaded here.

Dependencies

The following packages are necessary for the Python acquisition script to run in Linux:

- OpenCV
- git
- g++-
- cmake
- pkg-config
- libudev-dev
- libtinyxml-dev
- libgstreamer1.0-dev

- libglib2.0-dev
- libgirepository1.0-dev
- libusb-1.0-0-dev
- libzip-dev
- python3-setuptools
- gstreamer-1.0
- libusb-1.0
- libglib2.0
- libgirepository1.0
- libudev
- libtinyxml
- libzip
- libnotify

The SDK then needs to be built and installed. For help with this entire process, I would recommend following the instructions detailed here: tiscamera Github Page. This page also presents coding samples for operating the cameras which can be explored if using the cameras for a different purpose.

If using Windows, IC Capture, an application for operating the cameras which is very simple to use, can be downloaded here: IC capture.

Calibration

Once the cameras are set up, if Matlab is available, they can be calibrated following this procedure: Matlab Camera Calibration Toolbox. Alternatively, calibration can be performed in Python, by following this tutorial: Python OpenCV Camera Calibration Tutorial.

IMU

Connecting to the IMU

The IMU can be connected to the laptop directly through a USB port. Note that there have been problems identified when connecting the IMU to the laptop through a multi-USB adapter, and it is therefore best to connect the IMU directly to a port on the laptop.

Dependencies

The IMU acquisition code requires the Threespace Library to execute in Python. Installation instructions can be followed here: threespace API. Once installed, IMU data can be obtained using the acquisition script, however, it should be noted that the API does not provide a method for a Linux-based system to connect to the port of the IMU. An alternative method for doing this is set up in the Linux acquisition code. It may also be necessary to change the permissions of the port to access the IMU — this is detailed in the Linux acquisition script. If running the system in Windows, an alternative script which can be run through the command prompt is provided in the Google Drive folder.

Calibration

The IMU is calibrated using an application provided by Yost Labs. It can be downloaded here: 3-Space Software Suite.

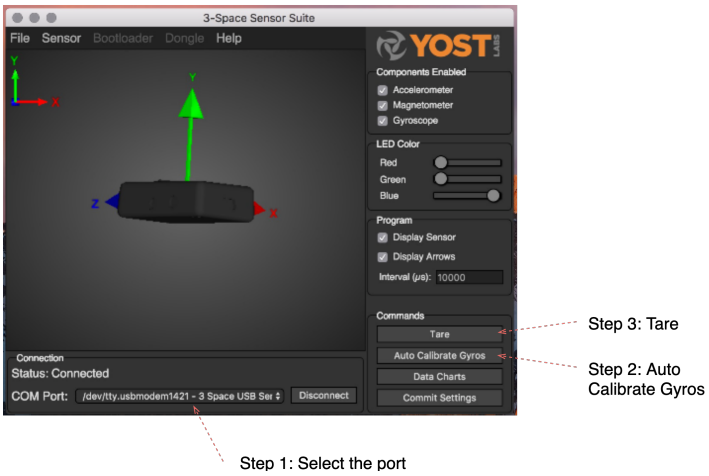


Figure B.2: Images showing the calibration of the IMU using the 3-Space Software Suite Application. Select the port and then press Auto Calibrate Gyros and then Tare.

Unfortunately, the application is not available on Linux, and therefore the Windows or Mac operating system needs to be used for this process. The IMU should only be calibrated once in position on the ship and instructions for calibration are shown in Fig. B.2. The IMU settings can be changed in the application to align the axes differently, however, it is recommended for the z-axis to be pointed upwards, and a second axis to be directed along the longitudinal axis of the ship.

In Windows, the software can also be used to log IMU data, if preferred to running the Python script.

External Trigger

The external trigger is used in both systems to provide power to the cameras, but only additionally operates as a trigger if running the system in Windows, as the application is not compatible with Linux. If using Windows, the trigger application can be installed from a ZIP folder found in the Google Drive. The application is simple to use, and only requires the selection of a frame rate and to press start. The ideal frame rate for this system is 1 fps.

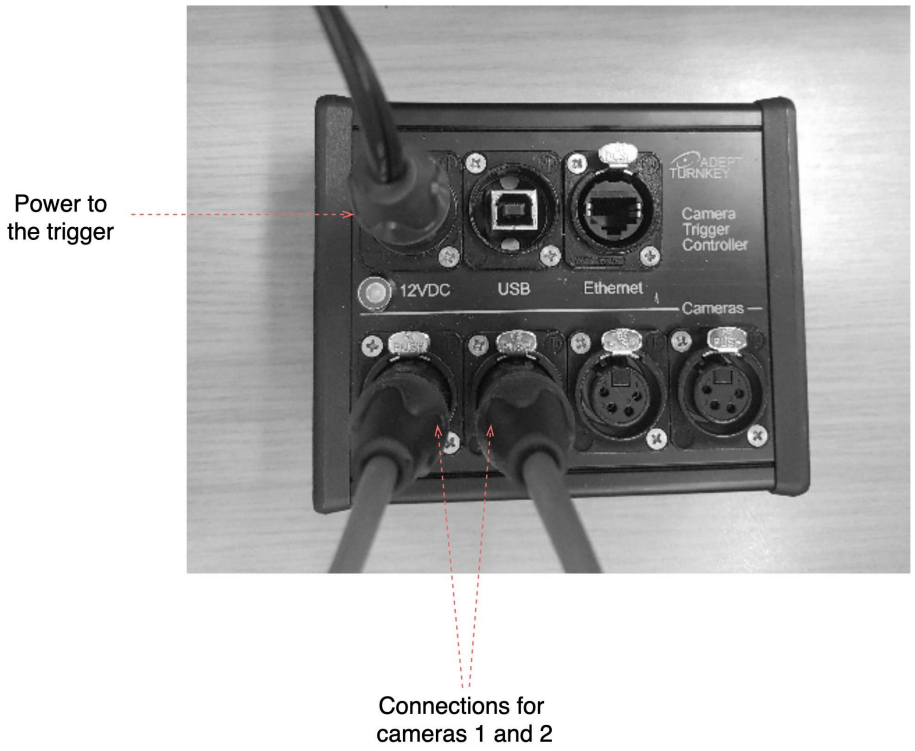


Figure B.3: Image showing the hardware trigger and its connections.

Heaters

The heaters should preferably be installed in the camera housing prior to transporting the equipment to the ship, however, it can also be done once on board. The assembly guide can be found in the Google Drive folder.

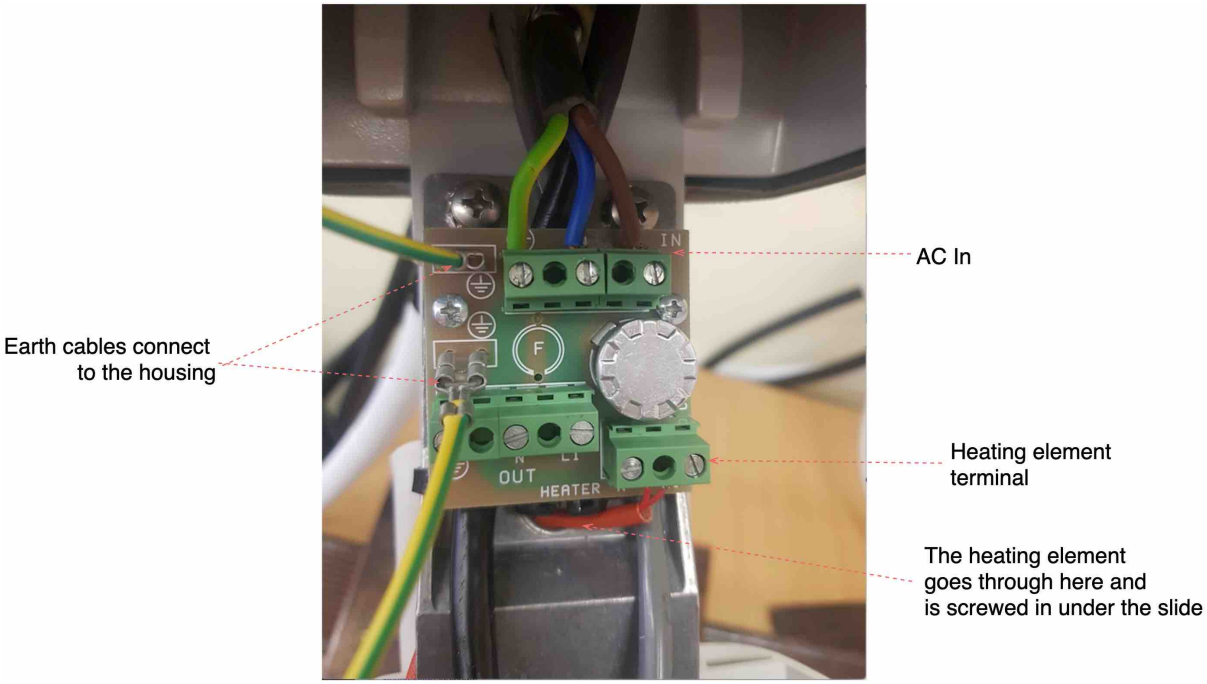


Figure B.4: Image showing the heater installation in the camera housing.

Note that the heaters turn on when the temperature of the circuit is below approximately 15°C, however, it is very difficult to feel them working through the housing as they do not make contact with the external part of the casing. If necessary, the best way to test them is to uninstall and disconnect them and leave them in a plastic bag in a freezer for roughly 10 minutes. They should then be quickly reconnected and the heating element checked to ensure it is warming up. Be careful if doing this, as the heating element heats up rapidly to temperatures above 100°C.

Ship Installation

Laptop, IMU, Hardware Trigger and External Drives

In order to keep the system contained and protected, the laptop, IMU, hardware trigger and external hard drives are all placed into a single box which stays in the Birder hut

(Fig. B.5) during acquisition. The box is tied down to the desk in the Birder hut using string or duct tape, to ensure it does not move around when there is significant wave motion. The connections between devices and sensors for both Linux- and Windows-based acquisition are detailed in Fig B.6.

Figure B.5: An image showing the location of the Birder hut and Monkey bridge on the ship, as well as where the cameras are installed.

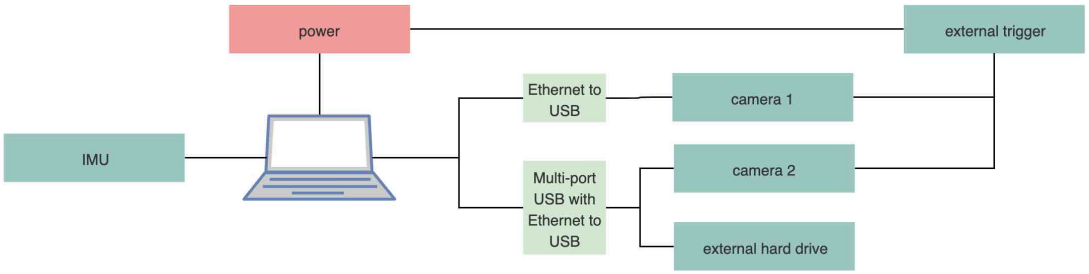


Figure B.6: A diagram showing the connections between devices and sensors for the system.

Cameras

The cameras are mounted across the corners of the railings on the Monkey bridge. Their cables run into the Birder hut, where the laptop is positioned. This is shown in Fig. B.7.

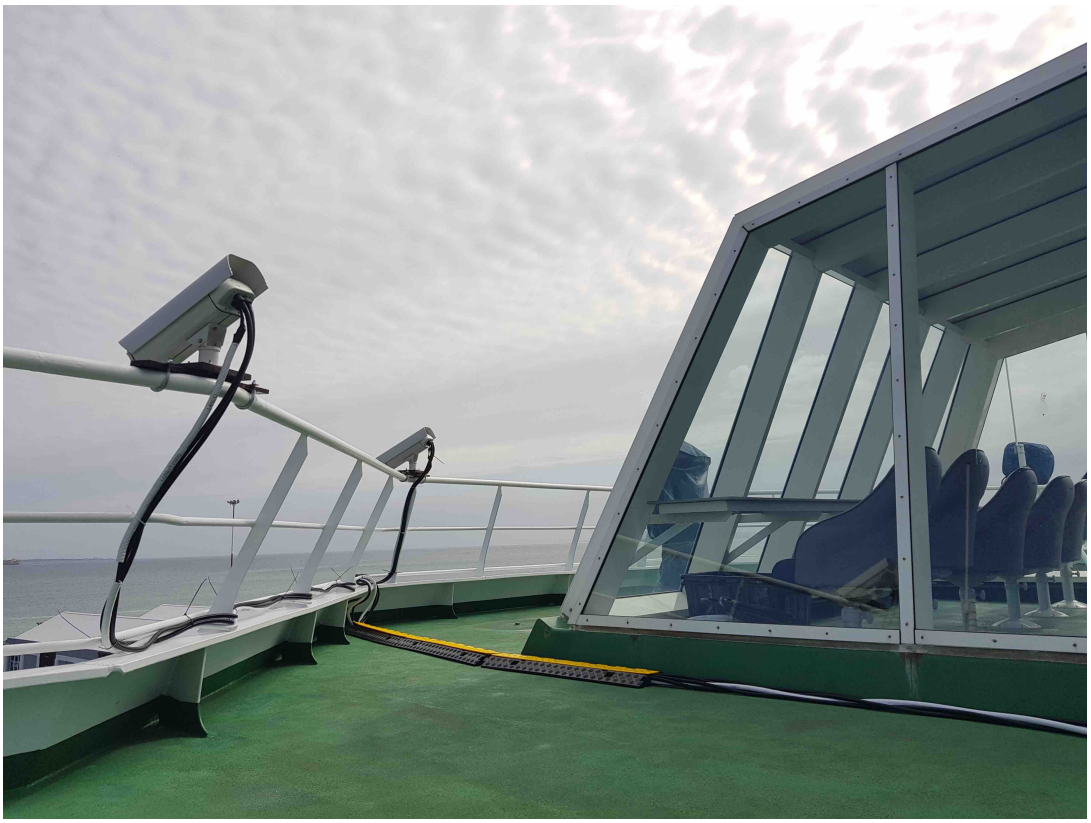


Figure B.7: An image showing the way the cameras are mounted on the ship. The cables run into the Birder hut (on the right), where the laptop will be located in a box.

Operating the system

Ensure the time on the laptop is correct before starting an acquisition session as all data obtained will be using the laptop's clock time for labelling.

Linux

The main acquisition script is named program.py. The amount of time or number of images to save to a single folder can be adjusted in the acquisition script, as well as the number of images to acquire before stopping an acquisition session. This is commented in the script for ease of use. Also ensure that the name of the external hard drive the data is being saved to is correct. If everything is correctly connected, the script can simply be run through the terminal by typing "python program.py" (Fig. B.8). You may be asked to provide your password in order to change the permissions for the IMU port, after which acquisition should begin. Check the "left", "right" and "imu" folders to ensure everything is operating correctly (Fig. B.9). To stop acquisition, type "ctrl c" in the terminal.

```
user@user:~$ python program.py
WARNING: No additional utils are loaded!!!!!
tcambin serial="44814140" name=source ! video/x-raw,format=GRAY8,width=2448,heig
ht=2048,framerate=10/1 ! appsink name=sink
tcambin serial="44814142" name=source ! video/x-raw,format=GRAY8,width=2448,heig
ht=2048,framerate=10/1 ! appsink name=sink
[sudo] password for user:
...
...
...
...
...
Starting acquisition now...
Press ctrl+c to stop
```

Figure B.8: Image showing the operation of the system through the terminal in Linux.

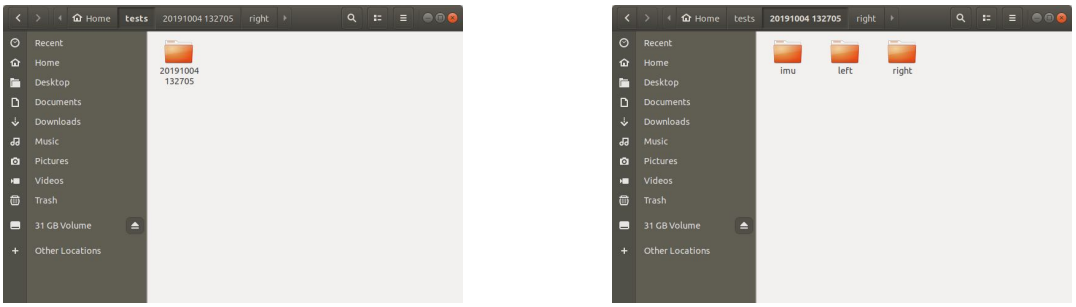


Figure B.9: Images showing the folder structure of acquired data. Each acquisition session is labelled by a timestamp (left) and in that folder are separate ones for the IMU data and left and right cameras (right).

Windows

Windows acquisition requires slightly more effort than Linux. IC capture should be opened, and the cameras must be selected. Open the hardware trigger application, select 1 fps and press start. Start the image acquisition in IC capture, and ensure that the external hardware trigger icon is selected in the top left.

IMU acquisition can either be done through the 3-Space Software Suite Application, or using the Python script. To run it with the Python script, open the command prompt, and type ”python IMU.py” to start acquisition.

Note that the main disadvantage of using Windows is that the IMU and image data will have to be post-calibrated, using their acquisition timestamps. There is currently no script for this as it was not done in this project.