SHARC Project

Ice Floe Size and Shape Characterisation from LiDAR Scans

Daniel Jones
(JNSDAN013)

Department of Electrical Engineering
University of Cape Town
Rondebosch, Cape Town
South Africa

Supervisor: Robyn Verrinder

October 2023

EEE4022S final year project report submitted in partial fulfilment of the requirements for the degree of **Bachelor of Science in Electrical Engineering/Electrical & Computer Engineering/Mechatronics** in the Department of Electrical Engineering at the University of Cape Town

Keywords: LiDAR; Motion Compensation; Feature Extraction; Point Clouds

Declaration

I, Daniel Jones, hereby:

1. grant the University of Cape Town free licence to reproduce the above thesis in whole or in part, for the purpose of research;
2. declare that:
   (a) this thesis is my own unaided work, both in concept and execution, and apart from the normal guidance from my supervisor, I have received no assistance except as stated below:
   (b) neither the substance nor any part of the above thesis has been submitted in the past, or is being, or is to be submitted for a degree at this University or at any other university, except as stated below.

This report is **17096** words.

Daniel Jones

Department of Electrical Engineering
University of Cape Town

Monday 30th October, 2023

Abstract

SHARC Project

Daniel Jones

Monday 30th October, 2023

This project embarked on the challenge of devising an initial version of a processing pipeline to discern the size and shape characteristics of sea ice floes within the Marginal Ice Zone (MIZ) of the Antarctic region. The endeavour was narrowed down to extracting parameters for rectangular prisms on flat terrain in a static reference frame, with the sole motion being attributed to the LiDAR technology employed. The objective was to lay down the basic framework of the pipeline, allowing for future enhancements. A two-fold processing pipeline, informed by relevant literature, was established, encompassing feature extraction and motion compensation. The feature extraction algorithm leveraged mathematical methodologies like RANSAC, Euclidean clustering, point cloud denoising, and a novel approach for parameter extraction from a prism-shaped point cloud, inclusive of a confidence metric to gauge the probable accuracy of measurements. Motion compensation was tackled using an Iterative Closest Point (ICP) algorithm aided by surface normals and Initial Measurement Unit (IMU) data for initial transformation, enhancing point cloud registration. The pipeline's efficacy was validated through simulated experiments prior to physical testing with a Livox Avia LiDAR scanner. These experiments scrutinized the feature extraction, motion compensation, and the impact of target distance on the pipeline's performance. Findings revealed that a medium filter applied on approximately 20 concatenated point cloud frames minimized relative error in static and motion scenarios. However, the algorithm's performance declined at increased distances, indicating a requirement for more frames and a refined filtering approach. The confidence metric displayed inadequacy in representing the algorithm's accuracy, suggesting the necessity for revision. Despite some shortcomings, the project achieved its primary goal of developing a pipeline to characterize Antarctic sea ice, laying a foundation for future work to build upon the findings and improve the pipeline's robustness in real-world Antarctic conditions. This initial exploration is a stepping stone towards precise sea ice characterization, which is imperative for understanding and monitoring changes in polar regions.

Acknowledgements

In many ways, this final year project was the culmination of all that I had learned throughout my four years studying to be an electrical engineer. It did not disappoint. I loved the whole process and feel as though it provided me with a taste of the true engineering experience that I initially signed up for in 2020. I have truly loved my time at the university, and I am excited to see what the future holds for me as an engineer going into a rapidly changing world. I wish to play my part in making the planet a better and more fascinating place for everybody.

It goes without saying that the character of my experience throughout my degree was in no small way governed by the people who joined alongside me on the journey. Consequently, many thank yous are in order.

Firstly, with regard to this project, I would like to extend a massive thank you to Agoritsa Spirakis, who, out of her own volition, provided me with so much support and guidance. I am so grateful for the help that she gave me, from suggesting literature to assisting me in understanding the LiDAR; her contribution cannot be overstated.

Secondly, I would like to thank my supervisor Robyn Verrinder who guided me along through the process, offering me invaluable sanity checks along the way. I appreciate her approach to the mentorship she gave me, ensuring that I did find my own way but never allowed me to get lost. Consequently, I really feel that this project was my own, and for that, I am grateful.

I would like to thank Brendon Daniels for his endless patience and kindness in assisting me in the 3D printing lab and Liam Clark for developing a Stewart platform for me to repair in the first place. Liam's documentation was immaculate, and thus I had no problem getting the platform up and running again.

A big thank you goes out to one of my oldest friends, Mishay Naidoo, for lending me his laptop, thus saving me many lonely hours in Red Lap. I would also like to thank Natasha Soldin for assisting me in the range experiments by running up and down the Green Mile. I hope it was as fun to do as it was to watch :)

To my newfound Jones family, my friendship with all of you has certainly been I highlight for me. A special mention to the tattoo group, I will remember you forever, whether I like it or not...

Lastly, to my boys that I live with, David, Mishay, Rory and Hamish. You are all simply the best. Without you, my university experience would be half of what it was. The endless positivity in our digs meant the world to me. I truly lucked out in the friendship department. I love you all and can't wait to adventure more with you in the future.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This report details the testing and development of a processing pipeline which is capable of distinguishing objects from each other within a point cloud before proceeding to extract the length, breadth and height of one of these objects as chosen by the user. It is further required that this pipeline be capable of accomplishing this task when the LiDAR used to capture the point cloud data is placed within a moving reference frame. These two requirements are derived from the eventual goal of implementing this pipeline on data collected from LiDAR located on the hull of a ship to measure the shape of the pancake ice floes in the Marginal Ice Zone in Antarctica. Included in this report is an investigation into the literature as well as recommendations of what work can be done to further the project.

Ultimately, this project aims to establish a rough skeleton of this pipeline, such that each component can be replaced by improved and more sophisticated techniques.

1.1 Background

The Antarctic sea ice plays a pivotal role in the climate regulation of planet Earth; from reflecting solar radiation to absorbing carbon, this ice is critical to the survival of a myriad of local and global ecosystems [14]. However, many of the current models and techniques used to analyse and predict the behaviour of this ice are based on those originally derived from the research conducted on *Arctic* sea ice. Due to the fundamental differences in the constitution of the Arctic sea ice versus the Antarctic sea ice, these models have proven inadequate, leading to models that incorrectly predict the Antarctic sea ice's future behaviour [18].

Due to the significant impact that Antarctic sea ice has on the global environment, there has been a marked increase in interest amongst the scientific community to develop more suitable methods of observing this ice. This includes evaluating various techniques ranging from remote sensing through satellite imagery and in-situ measurements with thermal and stereo cameras.

Another form of in-situ measurement, and the basis of this report, is Light Detection and Ranging (LiDAR). This method utilises an active sensor that emits a laser pulse and subsequently records the time that pulse returns from an object. By adjusting the direction of the laser between subsequent pulses and using this time-of-flight of the pulse, a three-dimensional map of an area can be established, often referred to as a point cloud. LiDAR utilises short wavelength light, thus enabling it to capture the finer details of an object accurately in comparison to alternatives such as radar [25]. However, radar is more penetrative than LiDAR; thus, clouds

and fog can often impede the view of a LiDAR sensor.

Consequently, LiDAR measurements are occasionally taken from a ship's hull to avoid the obstacles of cloud cover and fog. However, this reference frame is non-inertial, and as a result, this motion will have to be compensated for in the analysis of the data. This motion data is generally captured through an Inertial Measurement Unit (IMU) sensor. This device captures both the linear accelerations and angular velocities of the object to which it is attached.

The union of these two technologies, accompanied by the knowledge required to interpret the data they produced, could yield valuable insight into the hidden dynamics of the Antarctic sea ice.

1.2 Problem statement

This engineering report addresses the challenge of accurately characterising the physical parameters (size, shape, and thickness) of pancake ice floes in the Marginal Ice Zone (MIZ) of the Antarctic region, a crucial area for global climate regulation and carbon dioxide absorption. The Antarctic sea ice is subjected to various dynamic environmental factors, including wind, waves, and temperature, making it difficult to document and analyse, especially compared to its Northern Arctic counterpart [18]. Existing monitoring techniques from the Arctic region prove inadequate due to these unique challenges. Previous attempts to gather data have employed technologies such as radar, stereo imaging, and LiDAR, with the latter being the focus of this report. However, data captured from a ship's hull necessitates the development of algorithms to extract ice parameters from a moving reference frame. To address this issue, the report proposes creating a motion compensation algorithm using Iterative ICP and reverse transformations based on IMU data. Following motion compensation, a feature extraction algorithm incorporating RAndom SAmple Consensus (RANSAC) and Euclidean clustering will be implemented to derive the pancake ice floes' characteristics. The developed pipeline will operate on the scenario depicted in 1.2, where the LiDAR data is collected.

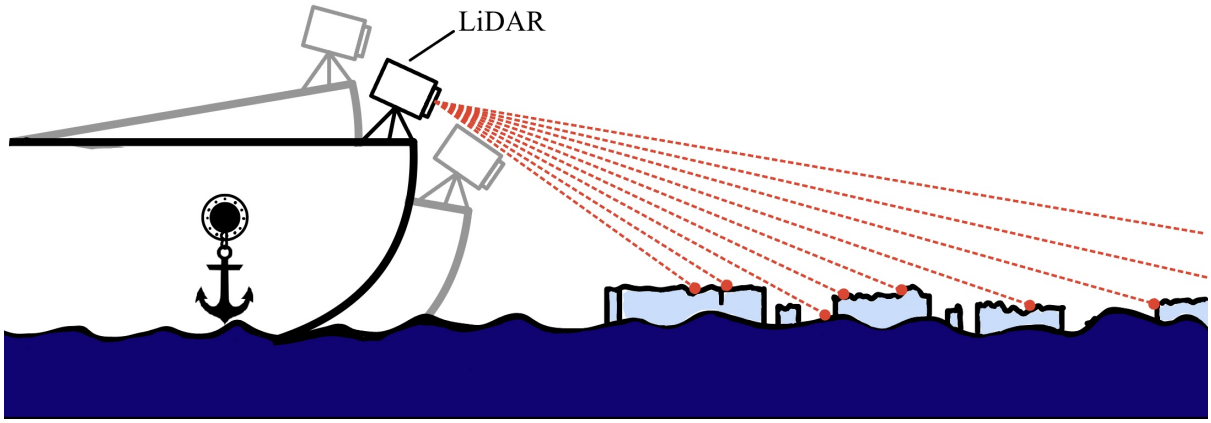


Figure 1.1: Figure demonstrating the scenario in which the LiDAR data is collected, upon which the developed pipeline will operate.

1.3 Project objectives

This project aims to take the initial steps in developing an algorithm for extracting the macroscopic shape parameters of an object from a moving reference frame using point cloud data obtained from a LiDAR scanner. In this context, macroscopic refers to the more prominent features of an object, such as its length, breadth and height. These findings, along with the methodology and research behind them, will be compiled into a comprehensive report before the 31st of October, 2023.

1.4 Scope, limitations and assumptions

This project aims to take the initial steps in developing a processing pipeline to achieve the goals outlined in the Project objectives. To this end, no numeric requirements are provided. Rather this report serves to produce an initial attempt at solving the problems, thus establishing a benchmark to be improved upon in further work. Thus, it is not expected that any stage of this pipeline will be particularly sophisticated. Instead, this project aims to establish the overarching sequence of steps required such that each individual step of the process could be easily replaced with an improved and more sophisticated method in future work.

Consequently, certain simplifying assumptions are made concerning the scenarios in which the processing pipeline will be deployed. Firstly, the feature extraction component will be limited only to extracting the length, breadth and height of rectangular prisms. Additionally, only scenarios in which the LiDAR scanner is placed within a moving reference frame are considered. Systems in which both the target object and LiDAR are moving are left for further work.

There are significant time constraints on this project. A period spanning from the 16th of July 2023 to the 31st of October 2023 is given to research, develop and test this pipeline and document these processes within this report. This severely limits the quantity and variety of testing that is possible.

Additionally, there are some hardware limitations present as well. The only Stewart platform available throughout this project was developed by Liam Clark in 2018. Additionally, there will be no experimentation regarding the type of IMU or LiDAR used. For this project, a Livox Avia LiDAR rig will be used along with its onboard IMU, the BMI088.

1.5 Plan of development

This report begins with a review of the literature regarding the Antarctic region and the existing techniques for characterising the sea ice in this region.

Following this, a detailed description of the mathematical principles and tools used to solve the problems outlined in the Problem statement is given.

With a deeper understanding of these principles at hand, the overall processing pipeline is discussed, showcasing how each of the mathematical tools discussed previously interact with one another to accomplish the goals of this project.

Before the methodology of the experiments conducted is discussed, a detailed breakdown of all the hardware and software resources is given. This includes significant information from data

sheets as well as version numbers of the software that is used. This is done to ensure that if one wants to recreate the developed algorithm or any of the experiments conducted, they can select appropriate substitutes for any of the software or hardware components which they do not possess.

Next, in-depth descriptions of the simulations along with the physical experiments that were conducted are provided. This includes diagrams of the experimental setups along with hyper-parameters and what values were used as inputs within the experiments.

The results of these experiments are then showcased in detail, accompanied by any observations regarding trends or anomalies found within the data. Following this is an in-depth discussion of the results in which the pipeline's performance is evaluated, and possible causes of its anomalous behaviour are given.

Finally, the overall conclusions of the project are drawn, along with recommendations for what future work can be carried out on the project.

Chapter 2

Literature review

2.1 Introduction

The consulted literature justifies the need for a processing pipeline to study LiDAR data of pancake ice floes in Antarctica's Marginal Ice Zone (MIZ). Following this, sources regarding the climate of the Antarctic region and the parameters which impact the formation and melting of the pancake ice floes in the MIZ are then consulted. These sources will help inform what techniques should be used when studying this region, as well as what parameters of the ice floes provide meaningful insight into the underlying physics. Next, the literature regarding LiDAR and LiDAR techniques for feature extraction is scrutinised in greater depth. These technologies are the medium through which the ice floe characterisation will be carried out in this report. With the knowledge that the LiDAR system used to capture the data analysed in this report was mounted on a ship, it was found to be necessary to review literature discussing sensing techniques from non-inertial reference frames to establish a better understanding of how one would account for both the movement of the ship as well as the movement of the ice floe in the water. Finally, a broad investigation into the current literature regarding ice floe characterization in Antarctica using LiDAR is conducted. This chapter concludes with a critique of the literature discussed.

2.2 Why the Southern Ocean sea ice should be studied

Research conducted in the Antarctic region is an expensive and often dangerous endeavour. Therefore, it is worth briefly investigating the importance of the Antarctic sea ice and its impact on its broader ecosystems to justify the undertaking of this research.

The ice surrounding the Antarctic continent plays a vital role in the cooling of Earth's surfaces [28]. sea ice exhibits a shortwave albedo of 0.8-0.9 [34]. Shortwave albedo is the reflectance of the shortwave radiation off the Earth's surface, often represented as a number between 0 and 1 where 0 would be complete absorption and one total reflection [34]. This shortwave albedo plays a significant role in the warming of the planet. In comparison to the value of $\sim$0.1 exhibited by the surrounding seawater [34], it is clear that an absence of this ice would result in the greater absorption of incoming heat, leading to rising global temperatures [28].

The Antarctic region plays a vital role in the scheme of the global ecosystem. The continent and its surrounding ice and water absorb up to 10% of the planet's anthropogenic carbon dioxide

(CO_2) emissions [14]. This is a consequence of what is referred to as Antarctica's "carbon pump", which can be further distinguished into biological and physical pumps, both of which are deeply influenced by the surrounding sea ice [29].

The physical carbon pump is a consequence of the heat exchange and oceanic mixing, which strongly impact the solubility of CO_2 molecules. Both the heat exchange and mixing processes are directly affected by the quantity of sea ice [14]. The less ice present in these regions, the more ocean surface area is available to strong winds, intensifying oceanic mixing processes [29].

Similarly, the Southern Ocean is home to large phytoplankton blooms, which, in much the same way as plants or vegetation, absorb (CO_2) through photosynthesis [17]. Of course, the availability of sunlight determines the population of these phytoplankton blooms and the time of year they occur. As with the physical pump, a decrease in sea ice extent (SIE)[1] will allow for greater sunlight exposure, thus increasing the carrying capacity for phytoplankton in the Southern Ocean [17].

Aside from their carbon absorption properties, phytoplankton form a core component of the marine food chain [29], and it is unclear what increases in their population will have on the ecosystem. Although it may seem that more food is now available due to the increased population, when phytoplankton blooms die out, they are decomposed by bacteria which deplete the surrounding waters of oxygen, essentially suffocating the nearby marine wildlife [29].

The Arctic region has been the subject of greater study due to its relative accessibility compared to the turbulent Southern Ocean Seas surrounding the Antarctic continent. Consequently, attempts have been made to utilise the same models and techniques developed for the Arctic region to study Antarctica. However, this has proven to be infeasible due to the marked divergence in response to global warming between the two regions [34]. Over the past three decades, the Arctic SIE has seen dramatic and anomalous reductions [18]. By contrast, the SIE in the Southern Ocean saw a slight *increase* in the same time-frame [18]. Although the differences in ice structure and geography are notable, the difference in behaviour between the two regions is still poorly understood. There is a noteworthy lack of in situ measurements from the region, which are essential to understanding the dynamics of the sea ice and forming a ground truth for the satellite data being captured [34].

As the above literature shows, Antarctic sea ice is critically important to global ecosystems, and its demise could have dire consequences. As such, a pertinent need exists to understand this region better so that effective mitigation techniques can be implemented to prevent future calamities.

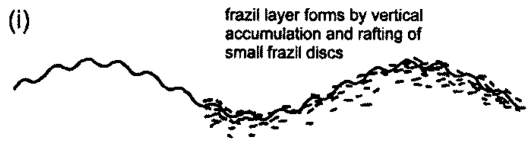
2.3 Antarctic pancake ice floes and what dictates their formation

The sea ice characterised in this paper, namely, the pancake ice floes found in the Marginal Ice Zone of the Southern Ocean, demonstrate unique behaviours that influence their formation processes, such as light permeability and how waves propagate through them. For this paper, it is essential to understand these dynamics to understand better which parameters, such as shape and thickness, are informative of these dynamics. This literature will inform the reader which parameters of the ice are to be prioritised in the characterisation process and potentially the form of the algorithms used to extract these parameters.

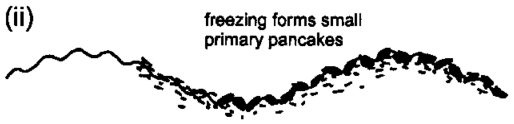
[1] sea ice Extent refers to the areas in which the concentration of sea ice is larger than 15%.

Shen et al. breaks the formation process of the pancake ice into three distinct stages: the Grease-Ice Stage, the Pancake Formation Stage, and the Composite Pancake Stage [33].

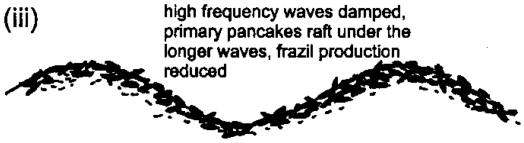
1. **Grease-ice stage:** As the sea water is super cooled, what are known as frazil crystals[2] begin to form. The smaller, high-frequency waves begin to corral the frazil ice crystals together, which combine to form a slush on the water's surface. This slush has the effect of attenuating the propagation of the low-frequency waves through the slush, as illustrated in figure 2.1a.
2. **Pancake Formation Stage:** As more frazil ice forms and combines its buoyancy increases, thus exposing some of the surface-lying frazil ice to the open air. The cooler air solidifies the frazil into a roughly circular sheet, the diameter of which is limited by the external forces being applied to the sheet through the movement of waves. This can be seen graphically in figure 2.1b.
3. **Composite Pancake Stage:** The pancakes are now pushed toward one another by the lower frequency waves, and loose, thin ice bonds form between each other, as seen in figure 2.1c.



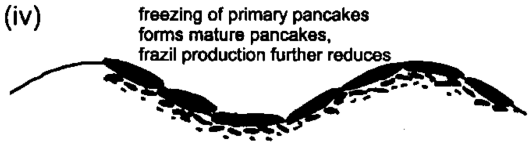
(a) Grease Ice Stage



(b) Pancake Formation Stage



(c) Composite Pancake Stage



(d) Rafting

Figure 2.1: Figure shown in Shen's et al. paper describing the stages of pancake ice formation. [33]

The process could be halted at any one of these stages if the energy is insufficient [33]. Additionally, these pancakes can increase in thickness through a process known as rafting, which entails a sufficiently high energy wave causing one ice pancake to beach itself on its neighbour. These two pancakes then freeze together, forming a thicker composite pancake as shown in figure 2.1d [33]. Mathematical models exist for these processes above; however, the details are considered outside this project's scope. For further information, the work of Shen et al. can be consulted [33].

Understanding these processes and what impacts them is vital to the study of the region, and accurately documenting the geometry of the ice is a crucial part of comprehending the underlying physical processes. However, the question of which parameters are most informative of these processes needs to be considered. The research conducted by Gherardi and Lagomarsino concluded that the thickness of the pancake ice floes was insufficient alone to describe the underlying physics [11]. Instead, the shape of the ice floes needed to be considered as well [11]. This means that the ability to measure specific lengths and distances along the ice flows is desirable. This will be taken into consideration in the development of the processing pipeline.

[2]Frazil crystals/ice are comprised of loosely fit and randomly orientated sub-millimetre sized ice crystals.

2.4 Current sea ice characterisation techniques

Besides LiDAR sensors, numerous other techniques and technologies are used for characterising sea ice. These technologies include laser-based altimetry, Synthetic Aperture Radar (SAR), and thermal and stereo camera imagery. However, due to its relevance and parallels to LiDAR, only satellite altimetry is considered in depth for this report. For information regarding thermal imaging, stereo photography, and Synthetic Aperture Radar (SAR), refer to the papers written by Kuhn [16], Niioko [26] and Kim et al. [15] respectively.

The expansive surface area of the polar ice caps, coupled with the adverse weather conditions surrounding them, make it nearly impossible to obtain holistic in situ documentation of the entirety of these regions, much less at the frequency required to observe the desired changing of parameters of pancake ice floes. This is the primary advantage of satellite-based observation techniques such as altimetry. Satellite-mounted sensors can capture the entirety of the polar regions in the order of magnitude of days [35], making them an invaluable tool in researching these polar regions.

Laser-based altimetry is a frequently used technique to characterise sea ice, particularly regarding its thickness. This technique involves a laser mounted on a satellite, which is directed directly downwards towards the Earth's surface. By recording the time (Δt) taken for a laser pulse to reflect off of the Earth's surface and return to the satellite, one can obtain the distance between that point and the satellite itself (D) using the simple equation $D = c\Delta(t)/2$ where (c) is the known value of the speed of light [35]. This is the same principle LiDAR uses to obtain distance information [25]. However, to determine the elevation of the contact point of the laser above the Earth's surface (E), the satellite's position in space has to be known within centimetre accuracy [35]. If this is the case, the value of (E) can be calculated using the following equation $E = h_{ellip} - D$ where (h_{ellip}) is the distance between the satellite's position and the Earth ellipsoid model [35]. For clarity, these distances are displayed visibly in figure 2.2.

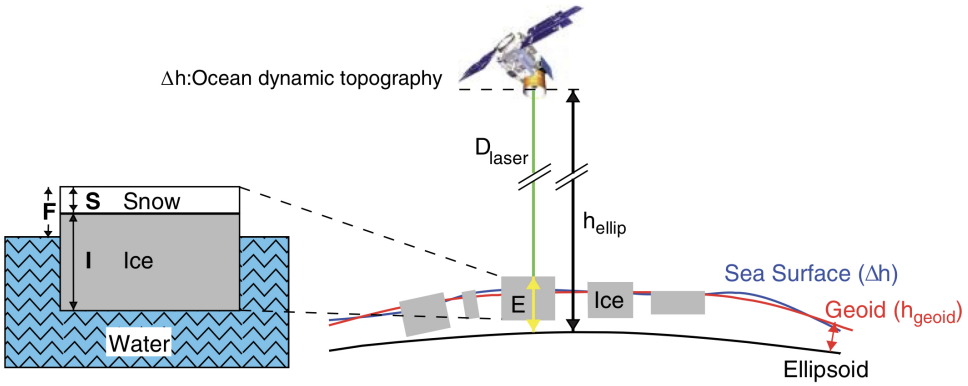


Figure 2.2: Graphic representation of distances involved in laser-based altimetry as found in Thomas's paper [35]

As previously mentioned, the primary characteristic of sea ice measured using altimetry is thickness, often denoted as (I). This value is indirectly obtained through the sea ice freeboard height (F)[3] combined with prerequisite knowledge of the densities of the seawater, ice and snow layered on top of it. A value of (F) can be obtained from the following expression where (SSH) refers to the sea surface height[4] and (h_{geoid}) refers to the Earth's geoid height $F =$

[3]This value refers to the height of the ice and snow content not submerged beneath the seawater.
[4]The SSH varies with tide, currents and atmospheric pressure.

$E - h_{geoid} - SSH$. Finally, with this calculated value of (F) used in conjunction with snow thickness (S) and the known densities (ρ) mentioned above, the sea ice thickness (I) can be determined using equation 2.1.

$$I = F_S \frac{\rho_W}{\rho_W - \rho_I} + S \frac{\rho_S - \rho_W}{\rho_W - \rho_I} \tag{2.1}$$

Some LiDAR sensors possess the ability to pierce through water [19], and thus, from the vantage point of a ship, may be capable of measuring sea ice thickness directly. However, not all LiDAR sensors are capable of seeing submerged objects, and in these cases, equation 2.1 may be helpful.

2.5 LiDAR techniques applicable to sea ice characterization

2.5.1 Different types of LiDAR

Although a LiDAR sensor is the technology used for data capture in this report, the technical details and implementation of LiDAR are not the focus of the research. Thus, this subsection only briefly discusses the various brands and implementations of LiDAR, along with the strengths and weaknesses of each type.

Light Detection and Ranging (LiDAR) is a form of active sensor[5] which uses short wavelength electromagnetic (EM) waves to obtain high-resolution data of the distance of objects and surfaces [25]. However, a consequence of the short wavelength used is that it does not pierce clouds or fog, which can sometimes limit the usage of LiDAR-based sensor systems [25]. Various LiDAR instruments exist and are usually distinguished from each other by their scanning methods and their dimensionality. This is illustrated by table 2.1.

After analysing these different LiDAR types, a solid-state LiDAR sensor appears to be well suited to ship-based LiDAR scanning of the pancake ice-floes. Since the LiDAR will be mounted on a ship, it will only be able to see the ice which is in front of it. Thus, the 360° field of view offered by mechanical LiDARs is just wasted functionality. This observation may prove helpful for future investigations.

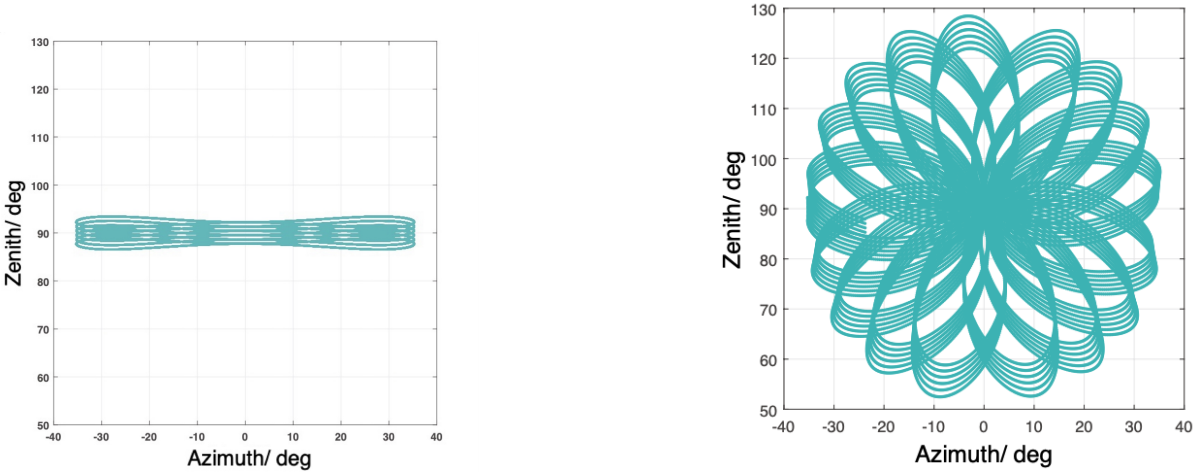
The fog penetrative abilities of the flash LiDAR may prove helpful as well, but its applicability may be limited by its short range as, at times, a ship's hull can be upwards of 20 m above the sea ice below.

[5] An active sensor refers to a sensor which produces its own energy pulse and measures the reflected from its target instead of passively measuring the energy generated from the target itself. [25].

Table 2.1: Different LiDAR scanning methods, along with their associated strengths and weaknesses, as described by McManamon [25]

Scanning method	Description	Strengths	Weaknesses
Mechanical	Uses lasers rotated about a central axis to obtain a point cloud of 360° view around the LiDAR.	Has a large field of view and a high signal to noise ratio.	It is bulky, expensive and susceptible to mechanical wear and tear.
Solid state	Uses a micro-electromechanical system to generate a point cloud in a particular window in front of the sensor.	Cost effective and small. Particularly when a 360 field of view is not necessary.	It has a limited field of view and a larger signal to noise ratio compared to the mechanical version.
Flash	Uses a singular flash to capture a window in front of the sensor.	Can penetrate fog and is cheap.	Possesses low range and has a high signal to noise ratio.

It is worth noting that for solid-state LiDAR instruments, the scanning pattern can be altered to generate different point clouds[6] depending on what is optimal for the use case. Some examples of this are shown in figure 2.3:



(a) Repetitive Point Cloud

(b) Non-Repetitive Point Cloud

Figure 2.3: Two examples of scanning patterns of a solid-state LiDAR system as found in the Livox Avia user manual [37]

These scanning patterns will produce point clouds with different densities at different locations. Experimentation with the optimal scanning pattern for the use case discussed in this report will prove valuable.

2.5.2 Feature extraction from LiDAR point cloud data

Merely capturing the point cloud data of the sea ice is not enough. The parameters of this sea ice need to be extracted to uncover the underlying physics. Cropp describes feature extraction

[6] In the context of LiDAR, a point cloud is the high-level output of a LiDAR system characterised by a series of points in three-dimensional space which describe the points at which the laser emitted by the LiDAR made contact with a surface.

as locating features such as length, height or cracks in point cloud data [4]. Cropp breaks feature extraction into three primary categories [4].

1. **Manual:** Features are identified and labelled by manually inspecting the data
2. **Cross-checked:** Here, an algorithm extracts the features but is still subject to manual cross-checking.
3. **Fully automated:** Here, an algorithm is entirely responsible for the feature extraction process. It is often desirable with fully automated feature extraction to include confidence metrics to provide the user with a gauge of how trustworthy the data is.

Automating the process of feature extraction is still an evolving field. It involves creating an algorithm that detects several features in a single point cloud. This can be a complicated task as it requires using mathematical tools that can make use of feature geometries to isolate each object. [4]. However, with the emergence of artificial intelligence (AI), more generally applicable feature extraction algorithms are now being developed [4]. Kim et al. used artificial intelligence to perform feature extraction on point cloud data of buildings [15].

Without artificial intelligence, however, one must develop intelligent strategies for autonomously extracting the desired features from a LiDAR scene. Sometimes, this is only possible for primitive shapes such as rectangles, cylinders or spheres. Ning et al. propose a means of extracting these sorts of primitive shapes from point cloud data by using a RAndom SAmple Consensus algorithm [27]. Ning et al. break up these primitive shapes into the following categories [27]:

- **Linear shapes:** Generally describes the boundary of an object comprised of a line.
- **Planar shapes:** A flat surface of an object that three points in three-dimensional space can fully define.
- **Cylindrical shapes:** Described as a series of points all a fixed radial distance from a line in space.
- **Spherical shapes:** Described by a sequence of points a fixed radial distance from a point in space.

By using these distinctions, these shapes could be detected in point cloud objects as shown in figure 2.4.

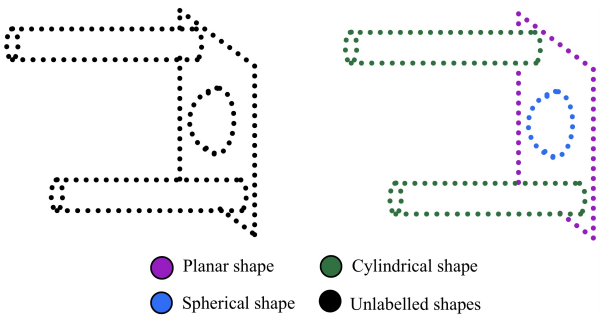


Figure 2.4: Figure showing how Ning's et al. algorithm extracts features from the point cloud, adapted from Ning's et al. paper [27].

In this way, the Ning et al. algorithm is capable of breaking up a more complex point cloud object into simpler primitive shapes. A similar algorithm was developed by Xu et al., which also used the RANSAC algorithm for feature extraction with the addition of a preprocessing algorithm, which improves the reliability of the feature extraction [39].

The two prominent techniques for autonomous feature extraction are artificial intelligence and RANSAC-based approaches. However, it appears that the RANSAC-based method only applies to objects of well-defined geometries.

2.6 Taking measurements from non-inertial reference frames

The techniques developed in this paper need to consider that the LiDAR sensor rig will be situated on a moving vessel, namely a ship. The ship's movement must be accounted for for an accurate interpretation of the collected LiDAR data. Accounting for these dynamics will require deploying appropriate coordinate systems and methods of describing orientation and transforming between reference frames.

2.6.1 Coordinate systems

In defining a coordinate system in terms of the moving reference frame of a ship, it is helpful first to define some terminology to describe the orientation and movement of the vessel succinctly. The motion of a boat in open water exhibits six degrees of freedom (DOF) given by the movements and rotation in the x, y and z directions. The names Fossen defines for each of these motions are shown in table 2.2 [10] and are illustrated graphically in figure 2.5.

DOF		Forces and moments	Linear and angular velocities	Positions and Euler angles
1	motions in the x direction (surge)	X	u	x
2	motions in the y direction (sway)	Y	v	y
3	motions in the z direction (heave)	Z	w	z
4	rotation about the x axis (roll, heel)	K	p	ϕ
5	rotation about the y axis (pitch, trim)	M	q	θ
6	rotation about the z axis (yaw)	N	r	ψ

Table 2.2: Name and symbol definitions of the 6 DOF of a ship as defined by Fossen [10]

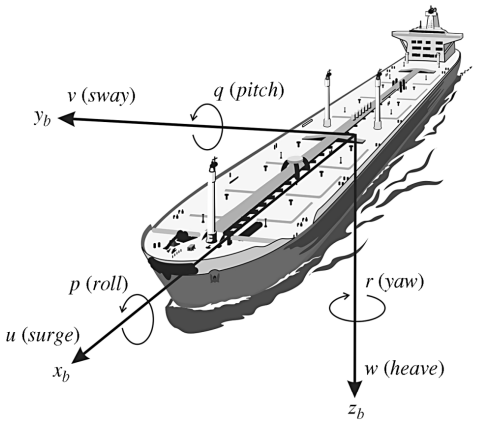


Figure 2.5: Graphic representation of the six DOF of a ship as illustrated by Fossen [10].

It can seen from the above literature that at any given moment, a ship could be undergoing motion subject to multiple degrees of freedom. It is thus necessary that the algorithm be capable of accounting for this.

Euler Angles

In relation to data captured from moving reference frames, it is often helpful to be able to rotate an object from one reference frame to another. A potential paradigm for achieving this is known as Euler angles, which are a set of three angles, roll, pitch and yaw, which describe the orientation of an object relative to a given reference frame [9]. This process of rotating between reference frames is represented mathematically through rotation matrices. A rotation matrix, sometimes denoted as $R_{i,\alpha}$ where (i) represents the axis of rotation and (α) represents the angle of rotation [9], is a square, orthogonal matrix[7] that when multiplied by a column vector in one reference frame, will output that column vector represented in another reference frame [9]. One can rotate any vector into any reference frame with a rotation in the x, y and z directions. An arbitrary example of this is shown below, representing a column vector v_1 in another reference frame as v_2 [9]:

$$v_2 = R_{z,\psi} R_{y,\theta} R_{x,\phi} v_1 \tag{2.2}$$

Critically, the order in which these rotations are applied is essential, meaning the resulting vector would be different if the order of rotations changed [9]. A drawback of this system is a phenomenon known as a gimbal lock. This occurs in systems that are subject to large angles of rotation. Essentially, a gimbal lock occurs when one of the axes aligns with another, essentially now representing the same axis [9]. This results in a loss of degree of freedom. In this scenario, not all rotations can be represented uniquely using Euler Angles. This leads to complications in the modelling process.

However, ships are very rarely subject to these large angles of rotation, and thus, the use of Euler angles would be suitable to this context.

[7] An orthogonal matrix is a square matrix in which the rows and columns are orthogonal unit vectors.

2.6.2 Point cloud registration

In dealing with point clouds captured from a moving reference frame, it is often helpful to be able to align two point clouds captured of the same object. Some of these techniques fall under the term *point cloud registration*, which is the process of aligning two point clouds describing the same object or scene [13].

Cohen breaks up point cloud registration into two main categories: optimisation-based techniques and feature-based techniques [3]. Cohen describes optimisation-based techniques as iterative processes [3]. The Iterative Closest Point Algorithm is named as a prominent example comprising of three steps [3]:

1. **Correspondence:** In this step, each point in one point cloud is associated with the nearest point in the other.
2. **Transformation:** In this set, a transformation is created by minimising a cost function by getting the corresponding points from step 1 as close together as possible.
3. **Iterate:** Repeat steps 1 and 2 until the two point clouds are sufficiently aligned. This could be based on a threshold value or a maximum number of iterations.

This technique offers global accuracy and can provide highly accurate alignment, especially with well-defined point clouds [13]. However, this technique is susceptible to local minimums and thus often needs a good initial guess as a transformation. This initial transformation can be derived from IMU data. The ICP algorithm has been used in numerous cases in the fields of computer vision, such as performing LiDAR odometry and loop closure [5] or performing feature extraction on a field of peach trees [40].

The other brand of point cloud registration Cohen mentions is feature-based techniques [3]. Instead of making associations between the closest points between the two point clouds, associations are made between features [3]. These features could be anything, from shapes or vertices to colour or intensity (assuming that is available in the point cloud data) [3]. This technique is less susceptible to outliers and poor initial conditions [13]. However, this technique requires that distinctive features be present in the point cloud scene. If this is not the case, the alignment will be poor [13]. It is common to pair a feature-based point cloud recognition algorithm with a deep learning algorithm [3].

It is possible to combine optimisation-based techniques with feature-based techniques such that the associations between the point clouds take into account feature information, but the process is repeated iteratively. An example of this is the generalised ICP algorithm proposed by Segal et al. includes information about surface normals in the correspondence step [32].

2.7 LiDAR usage in the context of sea ice research

With an understanding of the functionality of the LiDAR sensor, the literature regarding the current applications of LiDAR sensors in the context of sea ice characterisation can now be considered. LiDAR sensors have been implemented in this context primarily in the form of airborne LiDAR. Furthermore, it is found by the following literature that LiDAR sensors situated on moving reference frames are generally accompanied by additional IMU and GPS sensors to enable optimal insight into the data.

In the research conducted by Forsberg et al., a GPS along with an Inertial Measurement Unit (IMU) was installed on the freight aeroplane from which the LiDAR measurements were being taken [8]. The purpose of these additional instruments is to account for the movement of the plane and to make use of the aircraft's geographical position in the calculations referenced in the Current sea ice characterisation techniques section. In a similar study conducted by Wang et al., in addition to the IMU and GPS, a high-resolution, geo-referenced camera was used as a Digital mapping System (DMS) [38]. The (DMS) was utilised to manually select leads in the ice to create ground truth data points to compare to an autonomous lead-detection algorithm [38].

Both studies referenced above aimed to determine the thickness of the sea ice from their respective airborne LiDAR rigs. Both studies found that the most significant source of error in the ice thickness calculations was the thickness of the snow. This was attributed to the fact that it was unpredictable how deep the laser from the LiDAR would pierce into the snow's surface [8, 38]

In a paper by Sandru et al., a mechanical LiDAR was mounted on a ship to provide the vessel with a navigational tool, enabling it to travel safely and efficiently through ice-filled waters [31]. Once again, this study used an IMU sensor to compensate for the motion of the ship [31]. This paper aimed to generate a map of the sea ice surrounding the ship rather than extract parameters like size and shape from the sea ice itself.

As seen in the literature cited above, LiDAR sensors seem to be primarily applied to the context of sea ice to determine sea ice thickness. This shows a gap in the literature concerning the extraction of other sea ice parameters, such as size and shape.

2.8 Critique of the literature

As seen in the literature, there is a definite lack of in-situ measurements of the sea ice in the Antarctic region as a result of the adverse climate surrounding the area. There is an acute need to acquire these measurements due to the critical role that the region plays in the global ecosystems that all of humanity depends on. Consequently, the need for the research conducted in this report can thus be evidenced.

When considering the formation processes of the pancake ice flows, it is observed that physically observable parameters can identify each stage of the process. Literature regarding these processes highlights that merely the thickness is insufficient to describe the underlying physics fully, meaning additional parameters such as shape need to be considered. This means that the algorithm developed in this report must extract specific lengths of the objects within the scene.

Regarding the alternative methods of studying sea ice, laser-based altimetry was found to be a helpful parallel to LiDAR technologies. It was observed that the presence of an in-situ LiDAR sensor has certain advantages over the satellite laser. However, some of the techniques developed for altimetry, such as determining thickness from freeboard height, may also be helpful to LiDAR-based research.

In investigating literature regarding the LiDAR technology itself, it was found that solid-state LiDAR systems are a suitable option for the study of sea ice from a ship's hull as they are cheaper than mechanical LiDAR but provide comparable fields of view due to the hull of the boat blocking half of a mechanical LiDAR's view anyway.

The literature regarding autonomous feature extraction puts forth artificial intelligence and RANSAC-based techniques as standard tools. The AI-based techniques were found to be gen-

erally applicable; however, they require extensive training data. The RANSAC approach provides a simple way of extracting parameters from objects of well-defined geometries. Given the project's scope, the RANSAC algorithm seems to adequately accomplish the goals defined within the objectives of this report. Additionally, it was found that confidence metrics commonly accompany autonomous feature extraction.

In addition to this, the literature showed two prominent methods of aligning point clouds to compensate for the motion of the LiDAR sensor. These were optimization and feature-based techniques. It was found that a combination of these coupled with an initial transformation from IMU data can be used to remove the artefacts introduced into the data through motion.

Finally, the literature regarding LiDAR-based sea ice characterization in the Antarctic region primarily is concerned with airborne LiDAR rigs, once again emphasizing the lack of in-situ measurement. Additionally, most of these applications are concerned with the thickness of the sea ice, showing a gap in the literature concerning more detailed parameters such as shape. This further justifies the need for the research conducted in this report.

Chapter 3

Motion compensation and feature extraction design

The purpose of this chapter is to justify the design choices made and to outline the relevant theory that forms the basis of the algorithms and equations utilized by the data processing pipeline in this report. The precise sequence with which these principles are applied to the data is left for the subsequent Processing pipeline chapter. The task that this pipeline must accomplish is further subdivided into two primary objectives. These tasks are to firstly compensate for any artefacts introduced into the data due to the movement of the LiDAR sensor and secondly to extract the desired measurements of the objects in the frame. The mathematical tools used to accomplish each task are discussed in detail in the subsequent sections. Many of the principles that are discussed are visual in nature and are best explained with the aid of diagrams.

3.1 Motion compensation and point cloud registration

One LiDAR scan rarely contains enough data points to accurately extract the parameters of an object, especially if that object is a distance away from the LiDAR scanner. Consequently, to increase the density of points, multiple point cloud frames can be concatenated. However, if the LiDAR sensor is placed within a moving reference frame, consecutive frames will not align. The solution proposed in this report is to establish what transformation best aligns the two points clouds using the Iterative Closest Point (ICP) algorithm with an initial guess based on IMU readings.

3.1.1 Rotation and translation matrices

To align shifted point clouds, it is necessary to implement a mathematical tool for rotating and translating every point within a point cloud. It is desired that this transformation maintain the Euclidean distance between each pair of points within the point cloud, so as to not to alter its properties. An ideal candidate for accomplishing this task is using a forward 3-D rigid transformation [24]. This transform, represented by the symbol (A) in equation 3.1 rotates the point located at $(u, v, w)^T$ in 3D space to the position $(x, y, z)^T$ [32].

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = A \times \begin{pmatrix} u \\ v \\ w \\ 1 \end{pmatrix} \tag{3.1}$$

The matrix (A) contains both the rotational and translational components of the transformation and takes the following form:

$$A = \begin{pmatrix} R(1,1) & R(1,2) & R(1,3) & t_x \\ R(2,1) & R(2,2) & R(2,3) & t_y \\ R(3,1) & R(3,2) & R(3,3) & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

The values of $R(i,j)$ are the values of a 3D rotation matrix, the inner workings of which can be found in the Supplementary theory section of the Appendix. The values of t_x, t_y and t_z represent the translations in the x, y and z directions respectively. This matrix is what the ICP algorithm in the next section will alter in equation 3.3 to align a moving point cloud to a reference point cloud.

3.1.2 Iterative Closest Point (ICP)

With the ability to now translate and rotate point clouds, what's left is to establish an estimation of what transformation optimally aligns the two point clouds. To accomplish this, an ICP algorithm was implemented, the inner workings of which are explained below with the assistance of some diagrams. The language in this subsection refers to a reference point cloud (B), which is the point cloud to which the algorithm is trying to register with a moving point cloud (A).

ICP is an image registration[1] technique frequently utilized within the domains of computer vision. There are many different variations of the ICP algorithm. However, almost all of them contain two primary steps [32]:

1. Associate a subset of points within one cloud with points within the other.
2. Determine an optimal transformation that minimizes the distances between the points in step 1.

The first step is common to most variations of the ICP algorithm. The purpose of this step is to establish a metric by which the efficacy of the second step can be evaluated. This association is achieved by simply associating the points in the moving point cloud with the nearest points within the reference point cloud, as demonstrated in figure 3.1. The calculation of the nearest points is accomplished using k-d trees, which strategically partition the point cloud data such that the computation of the nearest neighbours does not require calculations of the distance between every point within the two point clouds [12]. This significantly optimizes the association process. For more information regarding the k-d trees algorithm, refer to the cited article written by Hristo Hristov [12].

[1] Image registration is the process of aligning two images into a singular image if one of the images is shifted, rotated or scaled [20].

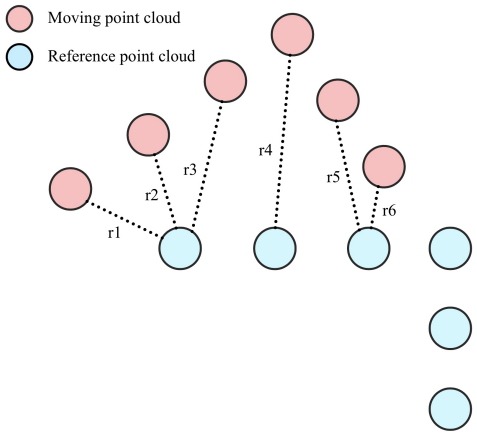


Figure 3.1: Figure illustrating how points in the moving point cloud are associated with points within the reference point cloud adapted from Segal's paper on ICP. [32]

As seen in figure 3.1, the associations made do not necessarily correspond to the correct features in each point cloud. However, ICP is an iterative process, and these associations will improve with each iteration.

The sum of the (r_i) values is essentially what is minimized in the second step of the ICP algorithm. It is at this stage where the different brands of the ICP algorithm diverge. For more information on the other variants of the ICP algorithm, refer to the cited paper written by Rusinkiewicz [30].

In this report, the error metric used to find the optimal transformation is referred to as the "plane-to-plane" metric [32]. Instead of treating the point clouds as points floating in three-dimensional space, this brand of ICP takes advantage of the fact that each point is constrained to the surface of an object [32]. The advantage of the plane-to-plane metric is that although the two point clouds are different scans and thus are unlikely to have perfectly corresponding points, it is likely that there will be points with matching surface normals [32]. Making use of this fact makes the algorithm more generalizable [32]. For these reasons, the plane-to-plane metric was used in this report. Using Principal Component Analysis (PCA) on the 20 surrounding points, the surface normals of each point can be calculated [1]. These normals are then used to establish covariance matrices for i_{th} points in both the moving and reference point clouds as represented by the notation C_i^A and C_i^B, respectively. These covariance matrices have large variances along the object's surface and minimal variances along the direction of the surface normal [32]. This is represented graphically in figure 3.2 where distances far from the point cloud point along the dotted lines represent large variance.

With an intuition regarding the constitution of these covariance matrices now in place, the cost function which will be used to determine the optimal transformation can be introduced:

$$\mathbf{T} = \mathbf{argmin}_{\mathbf{T}} \sum_i d_i^{(\mathbf{T})^T} (C_i^B + \mathbf{T} C_i^A \mathbf{T}^T)^{-1} d_i^{(\mathbf{T})} \tag{3.3}$$

In equation 3.3 above, $(\mathbf{T})$ represents the transformation matrix that is used to re-align the moving point cloud with the reference. The symbol $(d_i^{(\mathbf{T})} = b_i - \mathbf{T}a_i)$ which represents the distance between a point in the reference point cloud (b_i) and a point in the moving point cloud (a_i) after the transformation $(\mathbf{T})$ has been applied to (a_i) [32]. Essentially, the equation

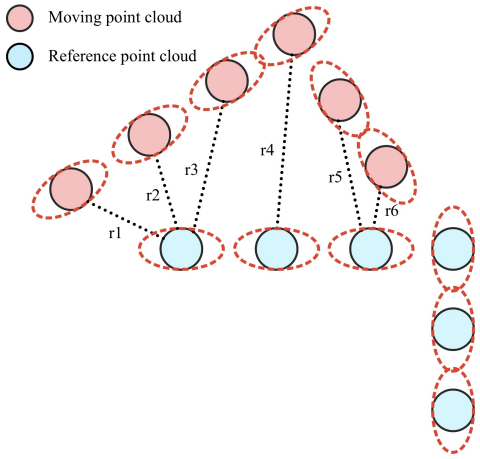


Figure 3.2: Figure illustrating graphically the covariance matrices for each point in a point cloud adapted from Segal's paper on ICP. [32]

functions by minimizing the sum of the square of the $d_i^{(\mathbf{T})}$ values by altering the value of ($\mathbf{T}$) through the Gauss-Newton method [32]. The covariance matrices act as weights of the cost function. In this way, the covariance matrices will discount incorrect associations made between the two clouds by weighting these associations with a minuscule number due to differing normals, suggesting they belong to different object features. They thus should not be considered a valid association [32].

Steps 1 and 2 of the algorithm will either run for a fixed number of iterations or until a user-defined threshold is reached within the cost function. In the algorithm developed for this report, the maximum number of iterations was set to 500.

This process can be sped up significantly by using a reasonable initial estimate of the transformation ($\mathbf{T_0}$). In this report, this initial transformation is formulated through IMU data collected from the LiDAR scanner. The steps to obtain ($\mathbf{T_0}$) are explained in detail in Appendix B.

3.2 Feature extraction

The feature extraction component of the algorithm is what ultimately obtains the measurements of the objects of interest. To accomplish this, it must first pre-process the point cloud to get it into a usable format. This means distinguishing each object in the point cloud scene from each other using Euclidean distance clustering. Next, by taking advantage of the known geometries of rectangular prisms, the object's dimensions can thus be extracted.

3.2.1 RANdom SAmple Consensus (RANSAC)

A crucial step that must occur before Euclidean distance clustering and parameter extraction can take place is the removal of the ground plane points from the LiDAR point cloud data. Without this process, all objects placed on the floor will be interpreted as the same object by the Euclidean clustering algorithm.

The RANdom SAmple Consensus (RANSAC) algorithm developed by Fischler and Bolles [7] is the approach taken to extract the ground plane point from the point cloud data in this

report. This method was adopted as it is well suited to the simplified scenes considered in the scope of the report and, additionally, is robust in the presence of outlying data points. The computer vision community widely adopts the RANSAC algorithm and is frequently used to fit a model to data which contains outliers. Derpanis describes the RANSAC algorithm as a five-step process [6]:

1. Select the minimum number of points from the data set required to define the model. For this project, three points are required to fully define a plane in three-dimensional space.
2. Using these points, determine the parameters of the plane in the form $Ax + By + Cz = D$.
3. Determine how many other points within the data set fall within a given tolerance (ϵ) of the plane defined in step 2 as illustrated in figure 3.3. Points that fall within this tolerance band are referred to as inliers, and those which do not are known as outliers.
4. If the ratio between the number of inliers to the number of outliers reaches a given threshold value (τ). The parameters of the plane can be recalculated using all of the inlying points.
5. If the threshold is not reached, repeat steps 1 through 4 for an user-defined N iterations.

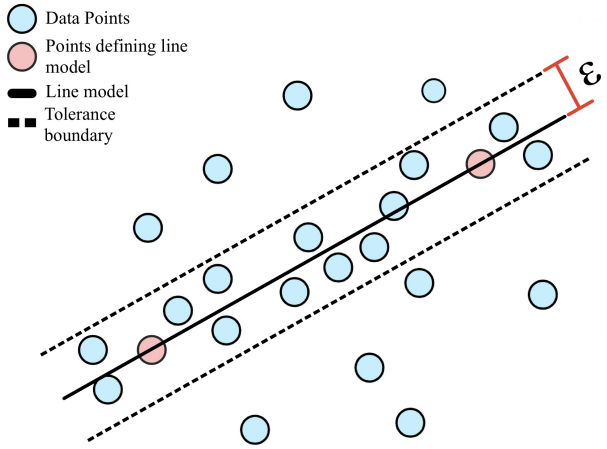


Figure 3.3: Figure showing an example of a model of a line defined using the RANSAC algorithm

This algorithm can be adapted to extract the ground plane of the point cloud data. Given that we know the normal of the plane we are trying to extract, in this case, the floor with a normal of $0\hat{i} + 0\hat{j} + 1\hat{k}$, we can discard any combination of points which produce a plane which differs from this normal more than a given angular tolerance (δ) [22]. Additionally, instead of waiting for a desired threshold (τ), always run the total N number of iterations and choose the model which produced the greatest τ value [22]. The rationale behind this adaption is that in a given LiDAR scene, out of planes which have normals parallel to the vector $0\hat{i} + 0\hat{j} + 1\hat{k}$, the ground plane would produce the highest threshold as it takes up the most significant proportion of the LiDAR's field of view [22].

All that remains is extracting all the inliers from the original point cloud, thus removing the ground plane.

3.2.2 Euclidean distance clustering

Once objects in the point cloud data are no longer joined via the ground plane, they can be segmented into distinct and separate items. This is done to allow the algorithm user to extract the parameters of a specific object of their choosing within the point cloud.

In this report, the technique used to accomplish this segmentation is referred to as Euclidean distance clustering. This algorithm was chosen due to its intuitive operation, allowing for easy adjustment to fit the needs of the pipeline. In order for Euclidean clustering to work as desired, some prior knowledge of the expected distances between the objects that one is trying to segment is required [23]. Additionally, one can specify the minimum number of points required before a cluster can be considered an object [23]. This functionality is added to reduce the size of the data by removing smaller objects present in a scene that are not of interest.

The first step in this algorithm is to assign null labels to all points within the point cloud [23]. Next, select one of these points and locate all other points within a sphere around that point. The sphere's radius is generally set to the smallest expected distance between two objects in the scene. From here, one of four actions is taken. If all of the other points in the sphere have null labels, label the selected point and all other points within the sphere with a new and unique label [23] as illustrated in figure 3.4 where (N) represents points with null labels:

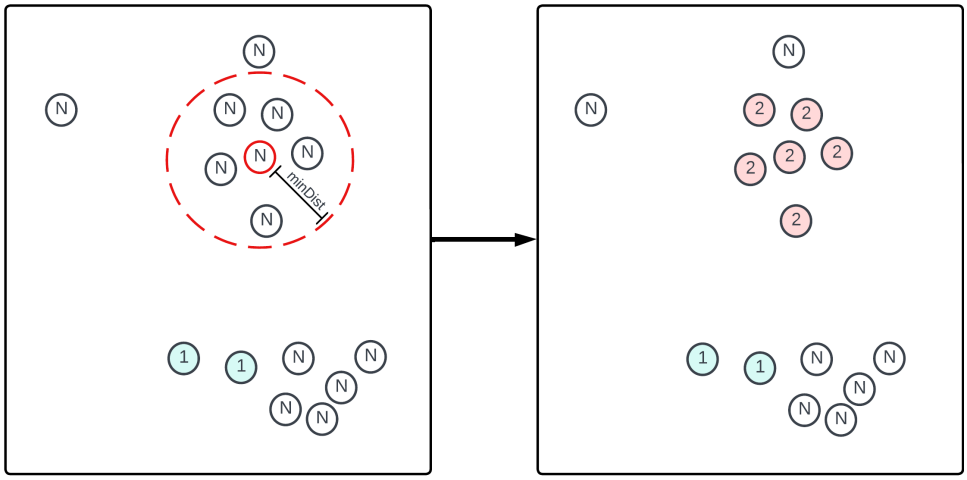


Figure 3.4: Figure illustrating how labels are assigned when all points in the search radius have null values.

If one or more of the points in the search radius already has a label assigned to it, then the selected point, as well as all other points with null labels in the search radius, are assigned to this label [23] as shown in figure 3.5.

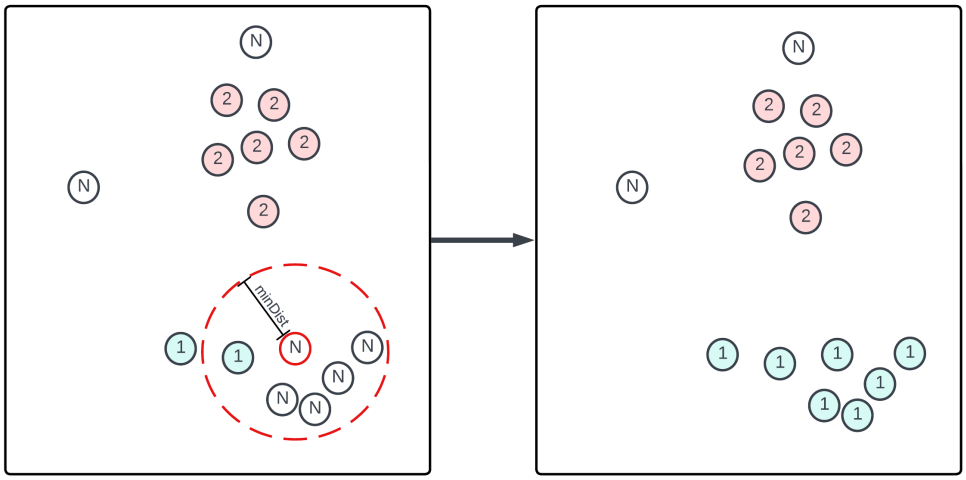


Figure 3.5: Figure illustrating how labels are assigned when some of the points in the search radius already have an assigned label.

However, suppose points with different labels are found in the search radius. In that case, the selected point, as well as all other points within the search radius, are re-assigned to the lowest value label within the search radius. This lowest value label is then propagated back through the other previously labelled points [23]. This is demonstrated in figure 3.6.

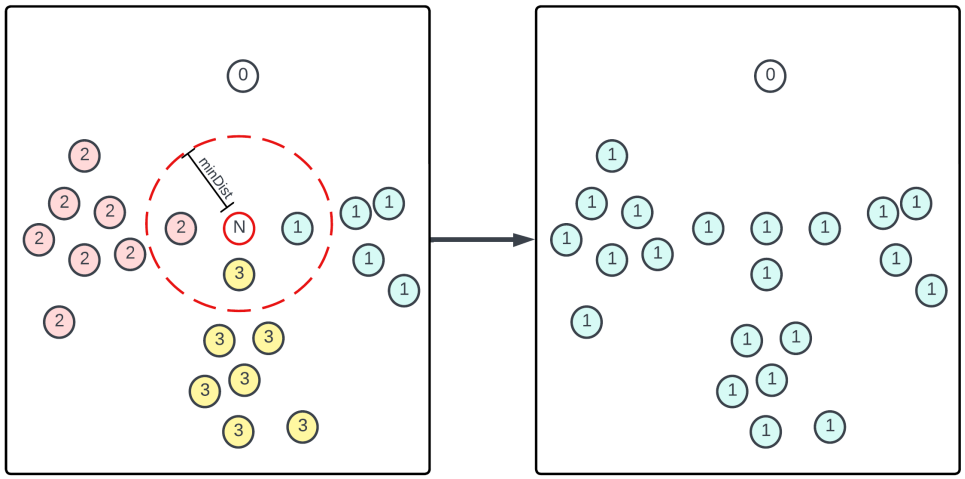


Figure 3.6: Figure illustrating how labels are assigned when some of the points in the search radius have already been assigned different labels.

In the situation in which there are no other points found within the search radius, the selected point will be assigned a zero label [23] as can be identified in figure 3.6.

Once no more null labels remain within the point cloud, all the points with a zero label are discarded. Additionally, all labels which contain fewer points than the user-defined minimum are discarded as well [23]. This allows the user to select a general size of the object they are looking for by discarding small objects containing fewer points.

Using the algorithm described above, the points in the point can now be associated to distinct objects.

3.2.3 Point cloud denoising

A critical tool in analysing point cloud data is the ability to remove noise points which are inevitably present. In this report, a statistical denoising algorithm is used, which compares the mean distance between a point and its neighbours with the mean distance of the other points to their neighbours to determine if the given point is noise or not [21].

There are two primary hyper-parameters of this algorithm. The first of which is the number of neighbouring points considered[2] by the algorithm. The more neighbours that are considered, the less susceptible the algorithm will be to noise, with the trade-off of increased processing time [21]. In this report, tests are run with this hyper-parameter set to 10, 100 and 500 neighbours, corresponding to mild, medium and aggressive filters.

The second hyperparameter is the threshold that determines whether or not a point is classified as noise [21]. The threshold is set by default in such a way that if the average distance between a point and its neighbouring points is one standard deviation away from the average distance of all the other points to their neighbours, then that point is classified as noise. [21]. In this report, this hyper-parameter is left at its default value.

3.2.4 Rectangular prism parameter extraction

With the point cloud segmented into various objects, the final task of the feature extraction component of the algorithm is to obtain the parameters of the objects within the scene. The scope of this project is constrained to only extracting the parameters of objects that showcase the geometry of rectangular prisms. Consequently, the algorithm assumes all objects have rectangular prism geometry. If this is indeed the case, then the rectangle's corners will always either be a minimum or maximum value. This is illustrated graphically in the figure 3.7.

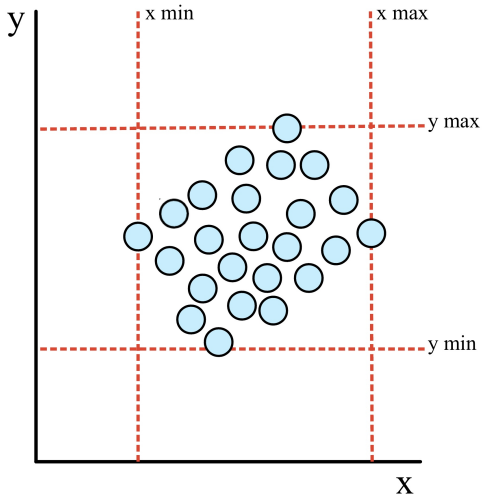


Figure 3.7: Figure illustrating the top of a rectangular prism point cloud and how the vertices are either maximums or minimums.

Thus, the corners of the rectangular prism in the x-y plane can be found by indexing through the points in the object point cloud and selecting the points with the maximum and minimum x and y values. Additionally, the points with the maximum and minimum z-values are located

[2]Often referred to as **numNeighbours** in the remainder of the report.

as well. With this information in hand, a model of the rectangular prism can be made by projecting each of the maximums and minimums in the x and y plane to both the maximum and minimum z-values. The box parameters can then easily be extracted by determining the Euclidean distance between these points.

A confidence metric was added to this algorithm to give the user a sense of how much a given measurement can be trusted. Given that the algorithm is designed only to extract the parameters of rectangular prisms, this confidence metric is based on how closely an object resembles the geometry of a rectangular prism. This was achieved by measuring how close the angles between the maximum and minimums in the x-y plane are to 90°. The closer all four angles are to 90°, the closer the confidence of the measurement will be to 100%.

To accomplish this, the absolute difference between the angle of each corner of the rectangular prism model (θ_i) and 90° is obtained and normalised ($\Delta\theta_{norm_i}$) as per equation 3.4.

$$\Delta\theta_{norm_i} = 1 - \frac{|\theta_i - 90°|}{90°} \tag{3.4}$$

Each normalised difference is then averaged to obtain a confidence percentage ($C_{\%}$).

$$C_{\%} = \frac{(\Delta\theta_{norm_1} + \Delta\theta_{norm_2} + \Delta\theta_{norm_3} + \Delta\theta_{norm_4}) \times 100\%}{4} \tag{3.5}$$

Chapter 4

Processing pipeline

With the relevant mathematical techniques and algorithms explained in detail in chapter 3, the manner in which these various principles interact with each other to accomplish the overall goal of the project is detailed by the flowchart alongside in figure4.1. The two hyper-parameters that are changed throughout the evaluation of the algorithm are **numFrames** and **numNeighbours**.

The **numFrames** variable determines how many frames are concatenated together by the ICP algorithm whilst the **numNeighbours** variable determines how many neighbouring points are to be considered in the Point cloud denoising algorithm. Many other hyperparameters could be altered, but through preliminary investigations, these were found to be the most impactful. Thus, experimentation with the other hyperparameters is left for Further work.

Once these variables have been defined, the algorithm proceeds to extract the point cloud and IMU data from the .bag files and aligns the desired number of frames using the ICP algorithm mentioned previously. The combined point cloud is then segmented into its various objects and then displays to the user a colour-coded plot of the objects as seen in figure 4.2. By observing which colour corresponds to which object in the legend of figure 4.2, the user can then select which object they would like the parameters of. This functionality was included to allow the user to easily select which object they want the parameters

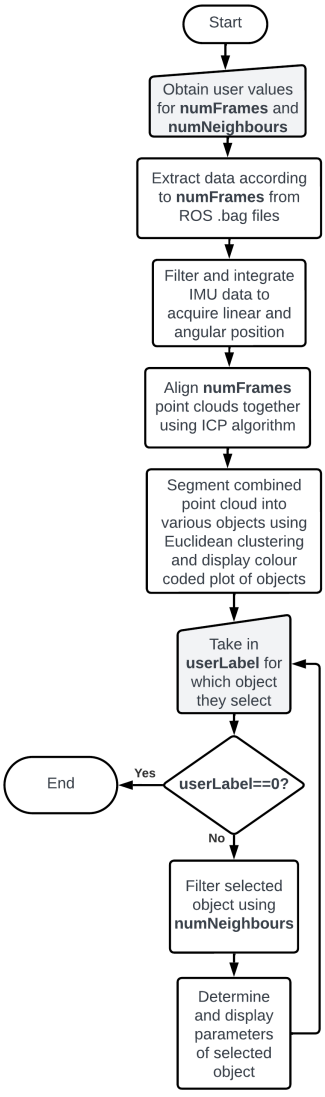


Figure 4.1: Overall algorithm processing pipeline.

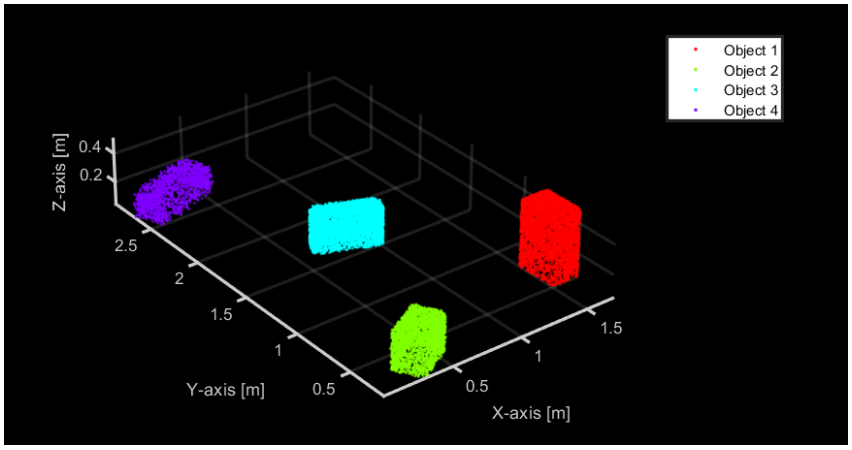


Figure 4.2: Figure showing the colour-coded plot displayed to the user, allowing them to select which object they want the dimensions of.

of in a scene that could contain multiple objects.

The algorithm will then filter outliers from the selected object in accordance with the **numNeighbours** variable defined earlier. After this filtration, the parameters of the selected object will be extracted using the Rectangular prism parameter extraction algorithm detailed previously. The length, breadth and height of the rectangular prism are then displayed in the terminal, along with a plot showing the algorithm's estimated model of the rectangular prism and the associated confidence in the measurement. An example of this is shown in 4.3 where the red lines and white points constitute the estimated model.

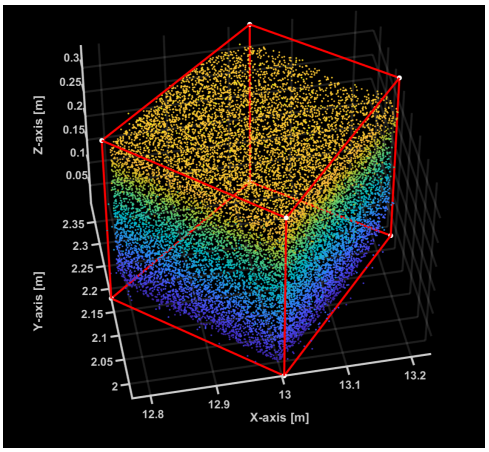


Figure 4.3: Figure showing an example of the rectangular prism model applied to an object.

The addition of this functionality is included to give the user additional insight into the behaviour of the algorithm.

The user can continue to select objects until they end the algorithm's running by entering a zero into the terminal.

The full code implementation and instructions on how to use the custom functions can be found in the project's GitHub Repository.

Chapter 5

Description of hardware and software

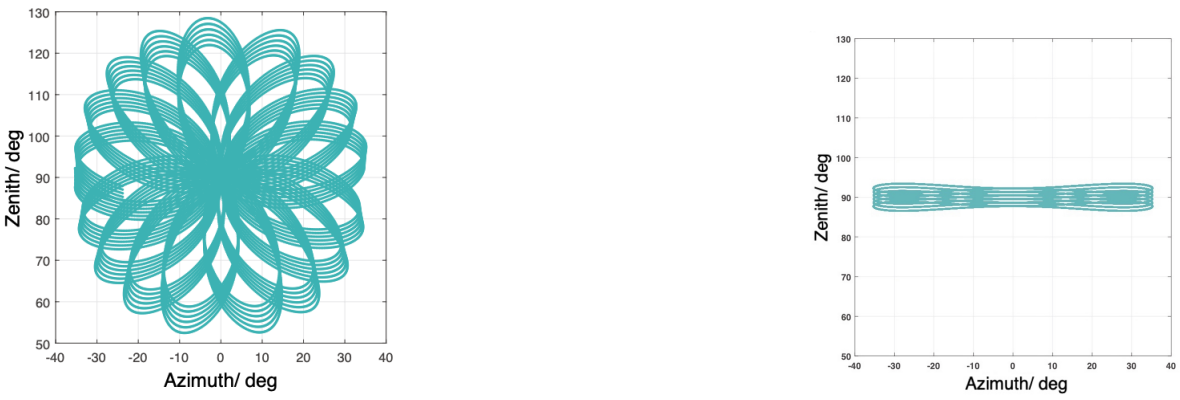
This chapter outlines the necessary hardware and software that was utilized throughout the execution of this project. This entails defining the relevant specifications of these hardware elements and disclosing details such as the version numbers of the utilized software. This provides further insight into the results of the investigation and ensures that an appropriate substitute can be selected in the event that one wishes to replicate the experiments detailed in this report.

5.1 Hardware

5.1.1 Livox Avia

A Livox Avia LiDAR system was the apparatus used to acquire the point cloud data for this project. The use of this specific hardware was not a design choice but rather what was available at the University of Cape Town at the time of the undertaking of this project.

The Livox Avia is a solid-state LiDAR system capable of producing two distinct scanning patterns, as displayed in the figure below:



(a) Non-repetitive pattern

(b) Repetitive pattern

Figure 5.1: The two distinct scanning patterns of the Livox Avia [37]

After completing a pattern cycle, it will rotate the pattern slightly upon the next cycle. Thus, the device will achieve a 100% FOV coverage after approximately 1.0 seconds [37]. This means that to acquire a complete point cloud of a target object, **ten consecutive point cloud frames will need to be concatenated**.

According to the Livox Avia user manual, the Field Of View (FOV) of the Non-repetitive pattern is 70.40° and 77.20°, whilst the FOV of the Repetitive pattern is 70.40° horizontally and 4.50° vertically [37]. The core specifications of the Livox Avia are tabulated in table 5.1.1 below. A complete list of specifications can be found in Full hardware and software specifications.

Table 5.1: Significant Specifications of the Livox Avia LiDAR [37]

Detection range @ 100 klx	190 m @ 10% reflectivity 230 m @ 20% reflectivity 320 m @ 80% reflectivity
Distance random error	(1σ @ 20 m)<2 cm
Angular Random Error	$1\sigma < 0.05°$
Pattern frequency	10 Hz
Wavelength	905 nm

5.1.2 BMI088 Inertial Measurement Unit (IMU)

The BMI088 IMU built into the Livox Avia was used to track the motion of the LiDAR and subsequently produce the corresponding transformations from the data. The BMI088 can measure 6 DoF with its 16-bit triaxial accelerometer and gyroscope and has built-in vibration suppression [36]. This device is capable of multiple output data rates, but for the purposes of this project, this rate was fixed at **200 Hz**. This value is considered to be sufficiently faster than the data rate of the LiDAR, and thus, experimentation with this data rate is considered out of the scope of this project.

5.1.3 Stewart platform

A Stewart platform was used in this project to allow for controlled manipulation of the motion of the LiDAR sensor in an attempt to simulate the movement of waves.

A Stewart platform consists of a flat surface situated on top of six actuators, which can either be pistons or motors. By engaging these actuators in specific sequences, the platform can perform motion exhibiting six degrees of freedom. The Stewart platform used in this project was developed by Liam Clark [2] as part of his thesis aimed at creating a physical wave simulator.

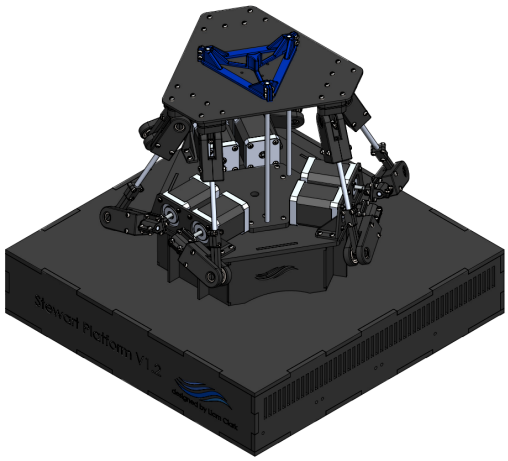


Figure 5.2: 3D Model of Stewart platform developed by Liam Clark [2]

To adapt the platform to the uses of this project, some of the damaged parts were replaced, and the motor mounts were adjusted by adding an additional plane to prevent bending, as shown in the figure 5.3.

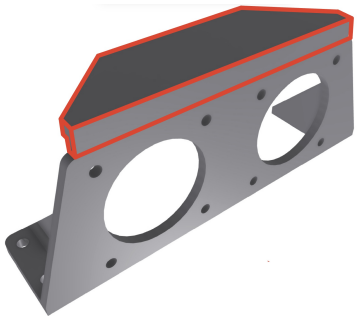


Figure 5.3: Stewart platform motor mounts where features added to the original design are outlined in red.

The motion of the platform is easily adjusted by altering the MATLAB scripts found in Clark's GitHub repository which allows for direct communication between one's computer and the platform through a USB connection.

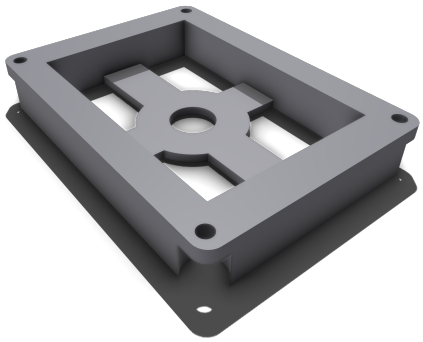


Figure 5.4: Custom mount designed to attach Livox Avia to the Stewart platform.

Additionally, to fix the Livox Avia to the Stewart platform, a custom mount was designed and 3D printed, as shown in figure 5.4.

5.1.4 Intel NUC

An Intel NUC with an i5 processor and 8 GB of RAM was used to collect the data directly from the LiDAR sensor through an ethernet cable. The NUC ran Ubuntu 20.04 and Robot Operating System 1 (ROS1).

5.1.5 Lenovo Legion laptop

All algorithms and data processing were done using a Lenovo Legion laptop with an i5-8300H CPU and 16 GB of RAM. At the time of this project, the laptop was running Windows 11 Pro version 22H2.

5.2 Software

5.2.1 MATLAB2023a

The processing of the point cloud and IMU data, as well as the design of the overall characterisation algorithm, was accomplished in MATLAB2023a. The specific libraries that require installation in order to run the algorithms designed in this report are listed below:

Table 5.2: Table showing the MATLAB packages required to run the scripts developed in this report providing brief explanations of each package's function

MATLAB package	Description
ROS Toolbox	The ROS Toolbox is required for this project to allow for the extraction of the point cloud and IMU data recorded on the NUC which is stored in the .bag format by default.
Computer Vision Toolbox	The Computer Vision Toolbox assists with point cloud manipulation, allowing for the implimentation of filtering registration, and clustering.
Image Processing Toolbox	The Image Processing Toolbox was used to create reverse transformations from the IMU data to allign point clouds captured from a moving reference frame.

Chapter 6

Experimental process

Since the goal of this project was to develop the first iteration of the processing pipeline, a holistic testing methodology was taken. This means that the pipeline was tested across a wide range of experiments and parameters rather than honing in on one particular aspect of the pipeline. This was done to ensure the system's behaviour as a whole was understood before improvements could be made to each submodule in future work. The approach to test and validate the feature extraction algorithm in this report was to divide the task into milestones of increasing complexity and difficulty. This ensured that each algorithm part was understood thoroughly, resulting in a robust final product. To this end, the experimental process is divided into Feature extraction and Motion compensation. Each of these processes is first tested and validated in a MATLAB simulation before being tested and validated through physical experiments. Lastly, an experiment investigating the range capabilities of the system was conducted as well. This chapter explains in detail the steps taken to develop each simulation and the setups and procedures of the physical experiments conducted.

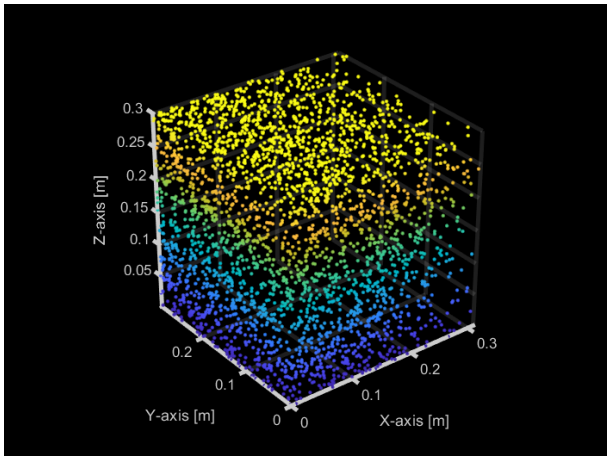
6.1 Feature extraction

The object characterization component of the algorithm aims to extract the desired features of the target object, namely length, breadth, and height. This requires that the algorithm be capable of discerning one object from another in a given scene and extracting the aforementioned parameters to a desired degree of accuracy. As mentioned, this functionality is first tested through a MATLAB simulation before being replicated in a laboratory-based experiment.

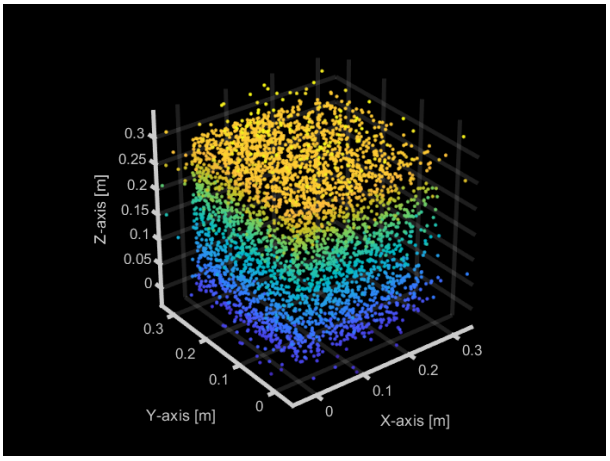
6.1.1 Feature extraction simulation

Full versions of the commented MATLAB scripts can be found here, however an overview of how they were developed and the rational behind the design decisions is provided below.

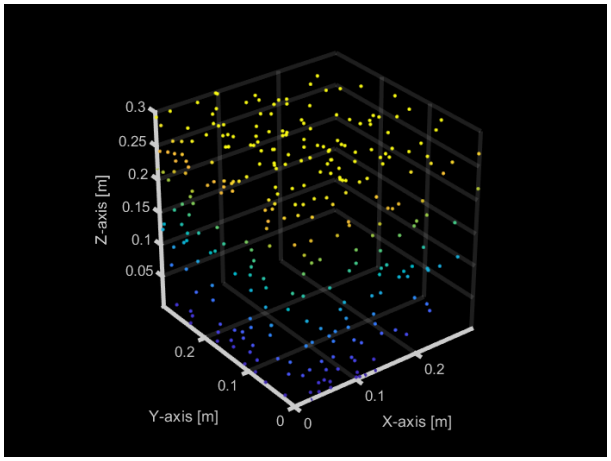
The first step in the simulation process is to develop point cloud objects from which the desired features will be extracted. To accomplish this, a **cubeGen()** function was developed. This function which takes in a length, breadth, height, translation, and rotation as inputs, and generates a point cloud object at the desired location with the corresponding rotation according to the specified dimensions. The points along the surface of the point cloud object become less densely distributed the further they are along the point cloud object. This simulates how



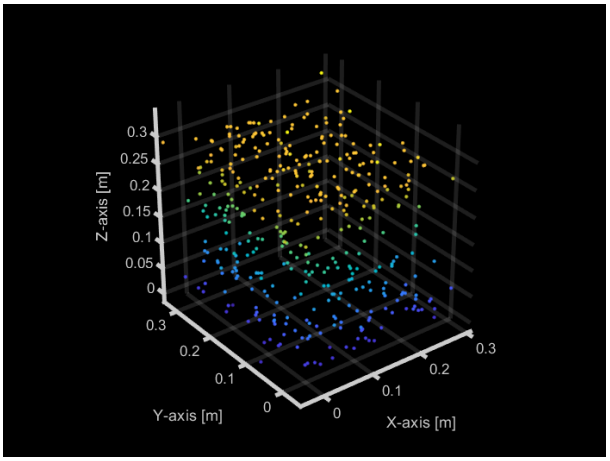
(a) A cuboid point cloud object with a density of 1000 points per plane and zero error



(b) A cuboid point cloud object with a density of 1000 points per plane and an error of 0.05



(c) A cuboid point cloud object with a density of 100 points per plane and zero error



(d) A cuboid point cloud object with a density of 100 points per plane and an error of 0.05

Figure 6.1: Figure displaying the effects of altering the density and error in point cloud objects generated by the cubeGen() function

points further away from a LiDAR scanner are further apart. Additionally, only three sides of the rectangular prism are generated as only three sides will be visible at any given time to a LiDAR scanner. Finally, a degree of Gaussian noise is applied to the point cloud object to simulate the noise points frequently observed in point cloud data generated by LiDAR scanners. The extent of the noise as well as the density of points, can easily be adjusted by changing the **error**[1] and **numPoints** variables in the **cubeGen()** function to values of one's choosing. A variety of example point cloud objects showcasing different densities and errors can be seen in figure 6.1.

A scene consisting of multiple of these point cloud objects such as those seen in figure 6.1 is then arranged where each object consists of different densities and errors as well as having a unique positioning, rotation, and dimension. To ensure that the scene replicates the actual LiDAR data, a ground plane is added to the scene as well. An example scene is shown in figure 6.2.

The scene is then fed into the feature extraction algorithm, after which the algorithm's performance is critically evaluated, with comparisons of performance made between the different objects based on their varying point densities. The primary metric by which the algorithm is

[1]The error variable linearly scales the Gaussian noise present in the point cloud object.

evaluated is the accuracy with which it estimates the physical parameters of the point cloud objects.

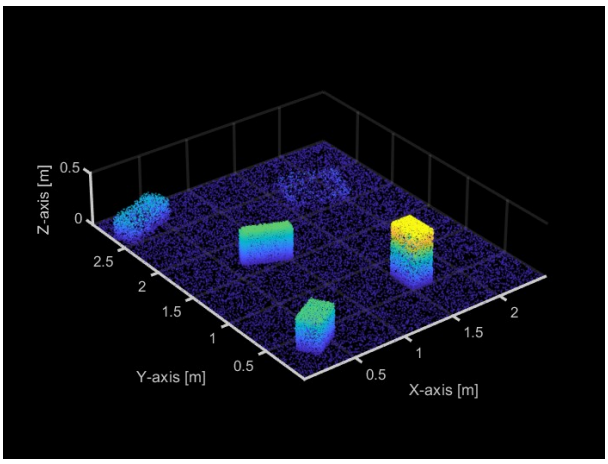
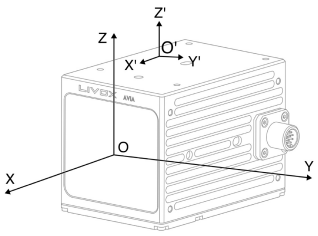
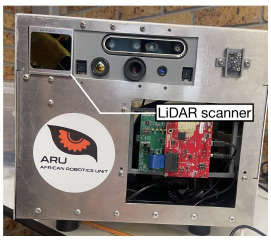


Figure 6.2: Figure showing an example of the simulated scene upon which the object characterization component of the feature extraction algorithm is tested

6.1.2 Feature extraction laboratory experiment

To test the feature extraction components of the algorithm's actual LiDAR data, the LiDAR scanner was placed on a stationary stand in a laboratory as shown in figure 6.3. The LiDAR scanner itself is placed at the origin in all measurements as highlighted in figure 6.3a. Using a stand ensures that the LiDAR experiences no motion whilst recording, allowing the data collected to be used to develop the feature extraction algorithm without artefacts due to motion being present in the data.





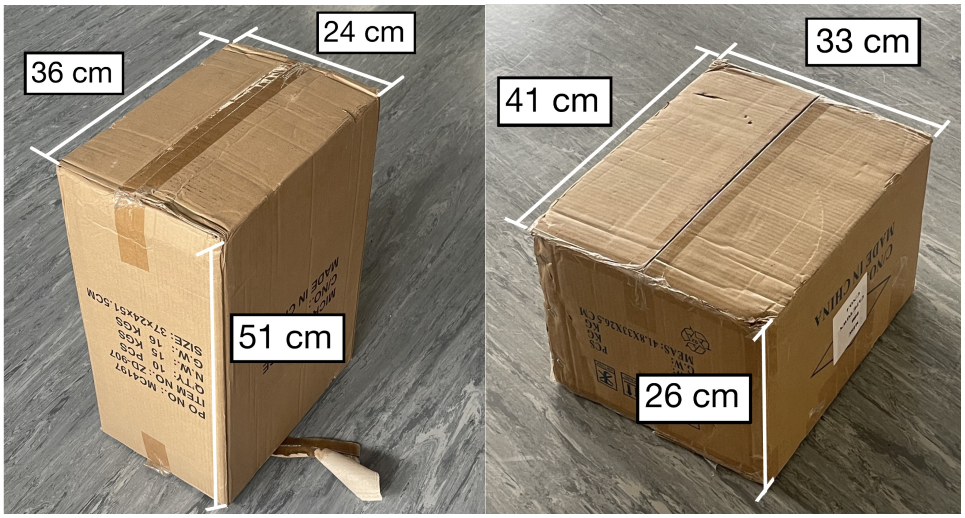
(a) LiDAR stand and LiDAR coordinate system.

(b) Laboratory where LiDAR is situated.

Figure 6.3: Figure contrasting the behaviour of mild and aggressive filtering on sparse and dense point clouds.

Cardboard boxes of known dimensions, shown in figure 6.4, are used as the target objects in this part of the experiment. The LiDAR data is then run through the feature extraction algorithm with using a mild, medium and aggressive filter as well as over a range of number of concatenated frames[2]. The algorithm is then assessed for each hyper-parameter combination concerning processing time and accuracy.

[2]As mentioned previously, these parameters were chosen for experimental variation, as preliminary testing indicating that they impacted the pipeline's accuracy most significantly.



(a) Box A **(b)** Box B

Figure 6.4: Figure displaying dimensions and textures of cardboard boxes used for the feature extraction experiments

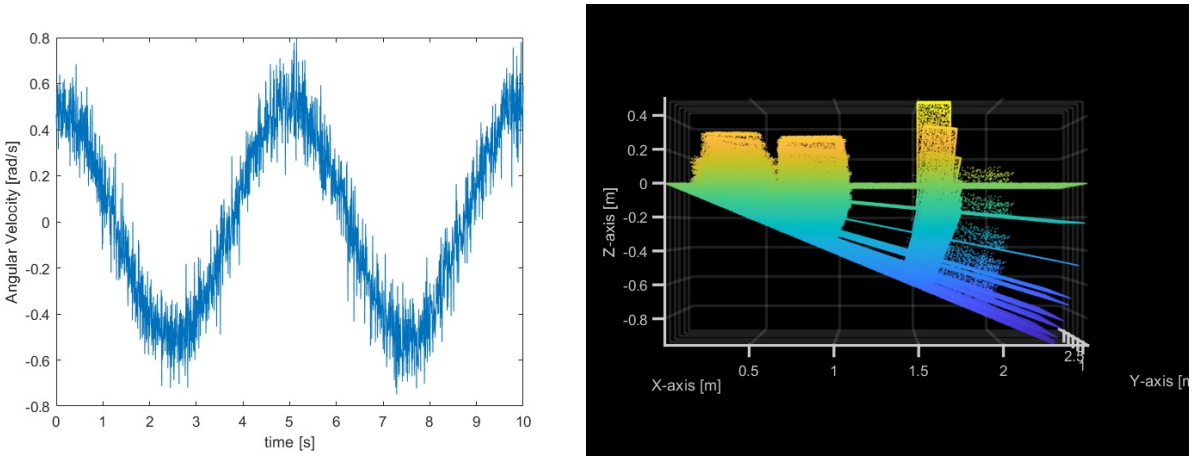
6.2 Motion compensation

The motion compensation component of the data processing pipeline needs to be able to align point clouds that have been shifted due to motion. This needs to be done such that they can be concatenated together, thus increasing the point density on the target object and, by extension, the accuracy of the parameter estimation. This component of the algorithm is first tested in MATLAB simulation before placing the LiDAR scanner within a moving reference frame and capturing real-world data.

6.2.1 Motion compensation simulation

The motion compensation simulation builds upon the simulation developed in the Feature extraction simulation. Once a scene, as seen in figure 6.2, has been created, the user can then use the **createMotionFrames()** function developed for this project to create a series of point clouds which simulate the effect of the LiDAR scanner being placed in a moving reference frame with corresponding simulated IMU data. To use the **createMotionFrames()** function, the user must define angular velocities and linear accelerations[3] which are functions of time. It is also essential to specify a sampling time for these functions. For this experiment, a sampling time of 200Hz was chosen to match the sampling time of the BMI088 IMU built into the Livox Avia LiDAR. Gaussian noise is then added to these functions to simulate the noise present in real IMU data. An example of the simulated IMU data with the associated moving point clouds can be found in figure 6.5. This simulated data can then be used to test the efficacy of the motion compensation component of the algorithm for a variety of different motions.

[3]The choice of angular velocities and linear accelerations as the functions to describe the motion of the LiDAR was made to simulate the outputs of the BMI088 IMU in the Livox Avia



(a) Simulated IMU data of an angular velocity about the y-axis

(b) The uncompensated frames simulating motion of the LiDAR scanner

Figure 6.5: Figure displaying the simulated IMU and LiDAR point cloud motion data

6.2.2 Motion compensation laboratory experiment

To be able to reliably control the motion of the LiDAR sensor, it was mounted to a Stewart platform as shown in figure 6.6.

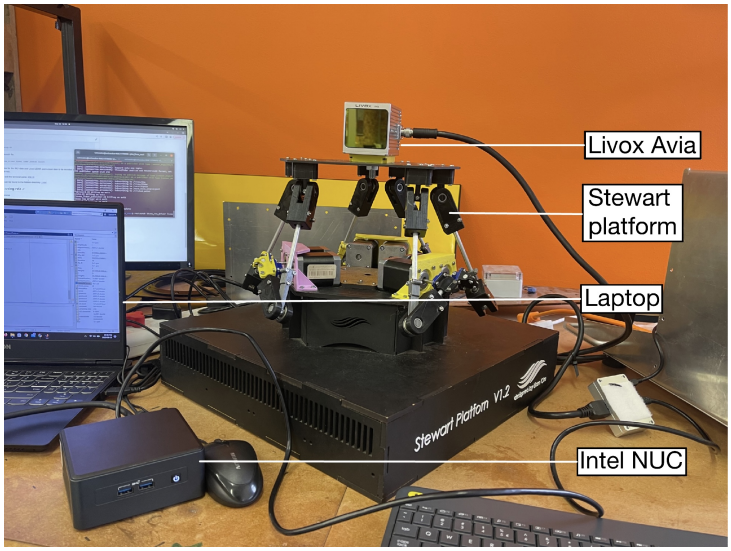


Figure 6.6: Figure showing the experimental setup of the motion compensation laboratory experiments.

The NUC captures and stores the point cloud and IMU data from the Livox Avia whilst the laptop sends movement instructions to the Steward platform via a USB connection.

The same cardboard boxes seen in 6.4 are used as the targets for this experiment as well. Similarly, the LiDAR was placed in the same laboratory seen in 6.7 A broad spectrum of motions was recorded, ranging from slow linear movement exhibiting 1 degree of motion to faster motion with both linear and rotational components. This is done to ensure a range of difficulty to test and develop the motion compensation algorithm. The motion functions fed

into the Stewart platform that was used for testing are shown in equations 6.1 and 6.2.

$$pitch_y = 10 \sin(0.33t)[degrees] \tag{6.1}$$

$$heave_z = 20 \sin(0.2t) + 150[mm] \tag{6.2}$$

This data is then fed through the data processing pipeline, the performance of which is analyzed concerning its processing time and accuracy for various combinations of filtering strengths and number of concatenated point cloud frames.

6.3 Range experimentation

The final experiments conducted in this project were designed to test the range capabilities of both the Livox Avia as well as the data processing pipeline.

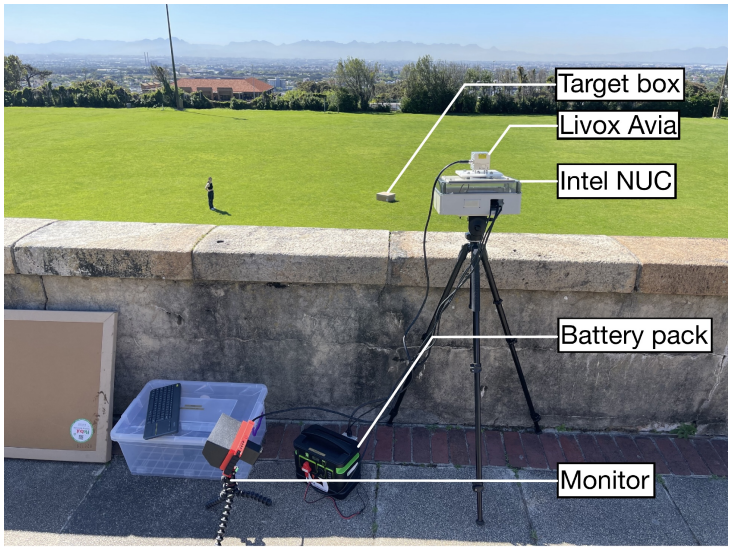


Figure 6.7: Figure showing the experimental setup of the motion compensation laboratory experiments.

In the setup shown in 6.7, above, the LiDAR is once again placed on a stationary mount 5.5m above flat rugby fields. The NUC is kept within a plastic container and is powered by a portable battery pack. A portable monitor is also used as a viewfinder for the LiDAR.

In this experiment, a larger cardboard box was used with a length, breadth, and height of 80 cm, 68 cm, and 38 cm, respectively, as shown in figure 6.8. Three data sets were recorded with the box at 20 m, 40 m, and 60 m horizontally from the LiDAR.

This data is then fed through the data processing pipeline, the performance of which is analyzed concerning its processing time and accuracy for various combinations of filtering strengths and number of concatenated point cloud frames.

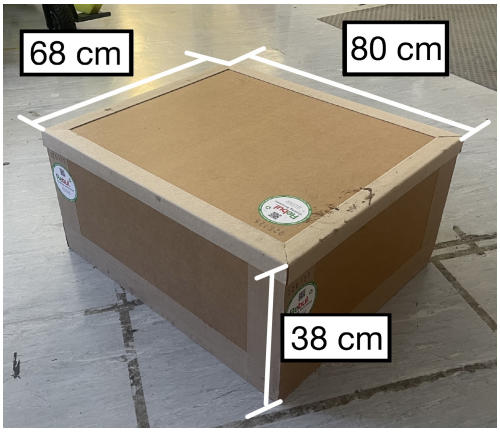


Figure 6.8: Box C

Chapter 7

Results and observations

In this chapter, the results of the experiments conducted throughout the Experimental process are displayed along with any observations regarding the data. As mentioned previously, three filters were applied to the data collected in all experiments. These filters are colour-coded in the plots as shown in figure 7.1. The term *neighbours* in figure 7.1 refers to the number of neighbouring points considered in the Point cloud denoising algorithm.

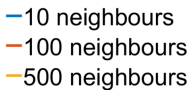


Figure 7.1: Legend of all plots pertaining to the accuracy, duration, or confidence of the algorithm

For ease of communication, the blue line (corresponding to 10 neighbouring points) is referred to as the mild filter, whilst the red line (100 neighbours) and yellow line (500 neighbours) are referred to as medium and aggressive filters, respectively.

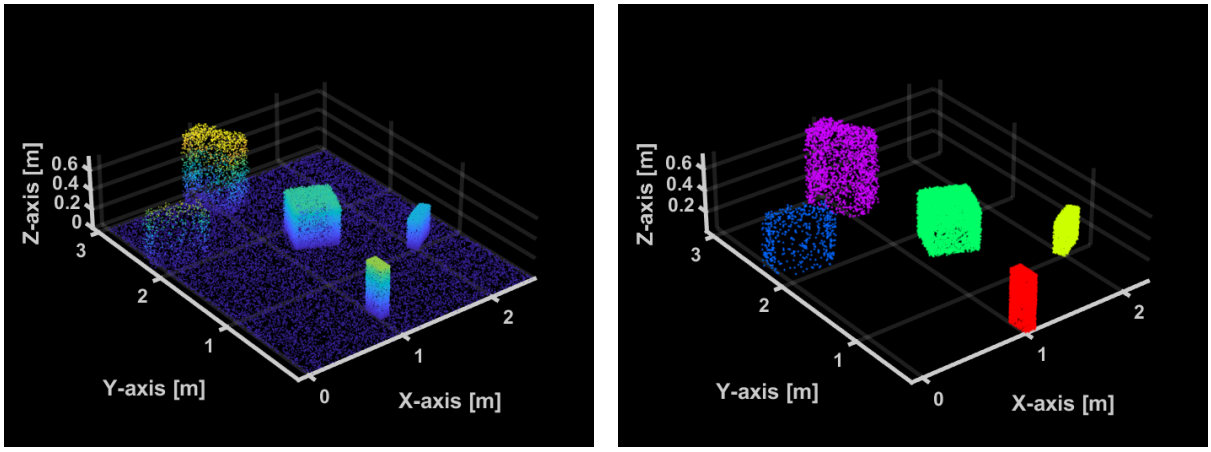
Some of the anomalies present in these results are accompanied by plots of the corresponding point cloud data such that these anomalies can be further scrutinized qualitatively.

7.1 Static feature extraction results

Shown here are the results and observations of the Feature extraction simulation and physical experimentation.

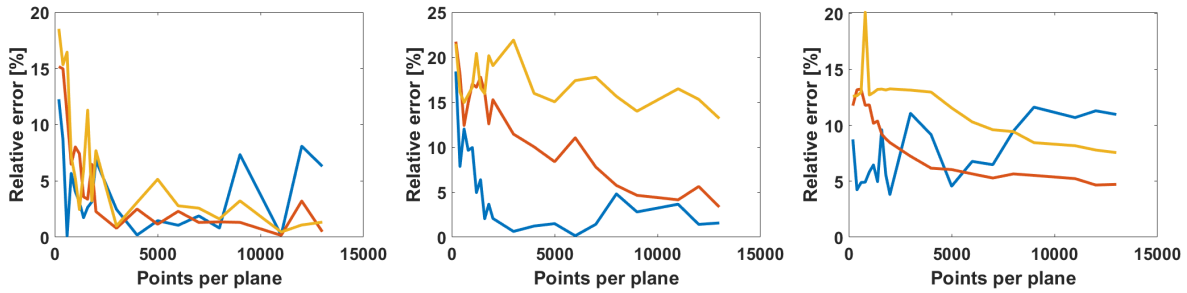
7.1.1 Simulation results

The efficacy of the Euclidean distance clustering algorithm on the stationary simulated data is shown in figure 7.2. It can be observed that after clustering, there is no remnant of the ground plane, and each object has been correctly segmented regardless of their varying densities.



(a) Simulated scene before clustering **(b)** Simulated scene after clustering

Figure 7.2: Figure showing the efficacy of the clustering algorithm.



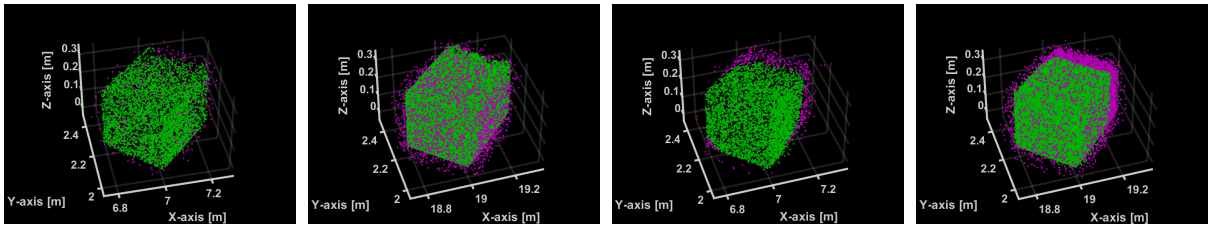
(a) Breadth of simulated box **(b)** Length of simulated box **(c)** Height of simulated box.

Figure 7.3: Relative error of the various dimensions of a simulated box as **numNeighbours** and points per plane are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters, respectively

A trend in the data in 7.3 shows that all filters behave worst when the point density of the simulated object is low, showcasing a relative error greater or equal to 10%. However, although the medium and aggressive filters show downward trends in error as the point cloud density increases, the mild filter reaches an error of less than 5% faster than they do. Additionally, as the point cloud density continues to increase, the mild filter begins to behave more sporadically, displaying an increasing trend in the relative error of its predictions. Interestingly, it can be seen that the relative error plots of the breadth of the simulated box differ in appearance from the plots of the length and height. For much of figure 7.3a, all three filters behave far more comparatively than for the length and height estimates.

To provide additional insight into the above data, the efficacy of the filtration and feature extraction algorithms is shown in figure 7.4 for both the mild and aggressive filters on low and high-density point cloud objects. In figures 7.4a to 7.4d, the green points represent the remaining point cloud after filtering and the purple points represent the points that are filtered out.

It can be identified in figure 7.4 that for low-density point cloud objects, the mild filter removes the noise points whilst the aggressive filter eliminates the noise points as well as some of the actual box itself. Additionally, for high-density objects, the model produced by the mild filter does not fit the box as tightly as the aggressive filter.

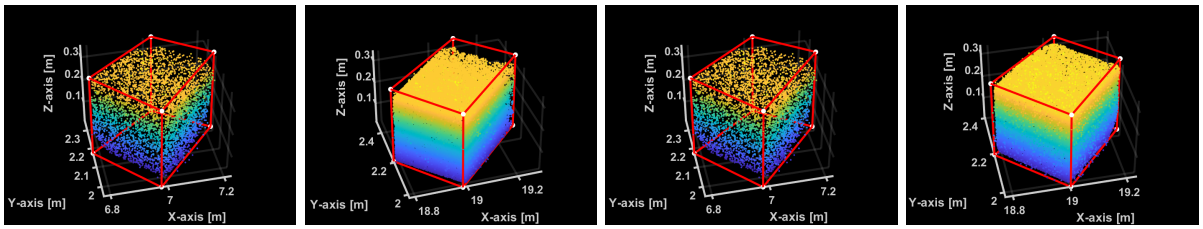


(a) Effect of mild filtering on low-density point cloud

(b) Effect of mild filtering on high-density point cloud

(c) Effect of aggressive filtering on low-density point cloud

(d) Effect of aggressive filtering on high-density point cloud



(e) Estimated model of low-density point cloud after mild filtering

(f) Estimated model of high-density point cloud after mild filtering

(g) Estimated model of low-density point cloud after aggressive filtering

(h) Estimated model of high-density point cloud after aggressive filterng

Figure 7.4: Figure contrasting the behaviour of mild and aggressive filtering on sparse and dense point clouds. Purple points represent filtered-out points

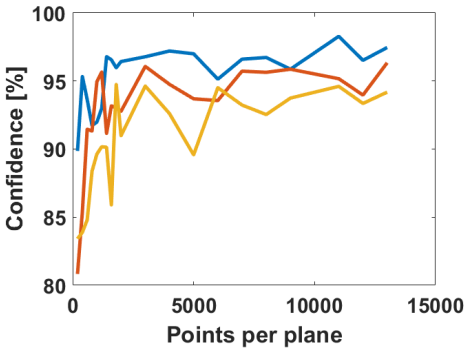
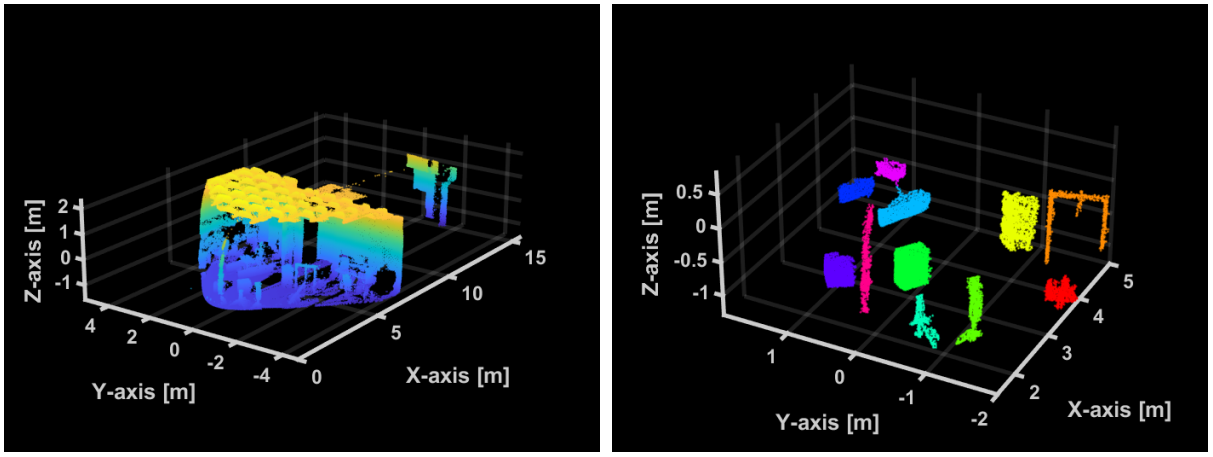


Figure 7.5: confidence in simulated box measurements as **numNeighbours** and points per plane variables are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

It can be observed in figure 7.5 that the confidence of all three filters increases as the density of the point cloud objects increases. The confidence level plateaus around 5000 points per plane at around 95%.

7.1.2 Physical experimentation results

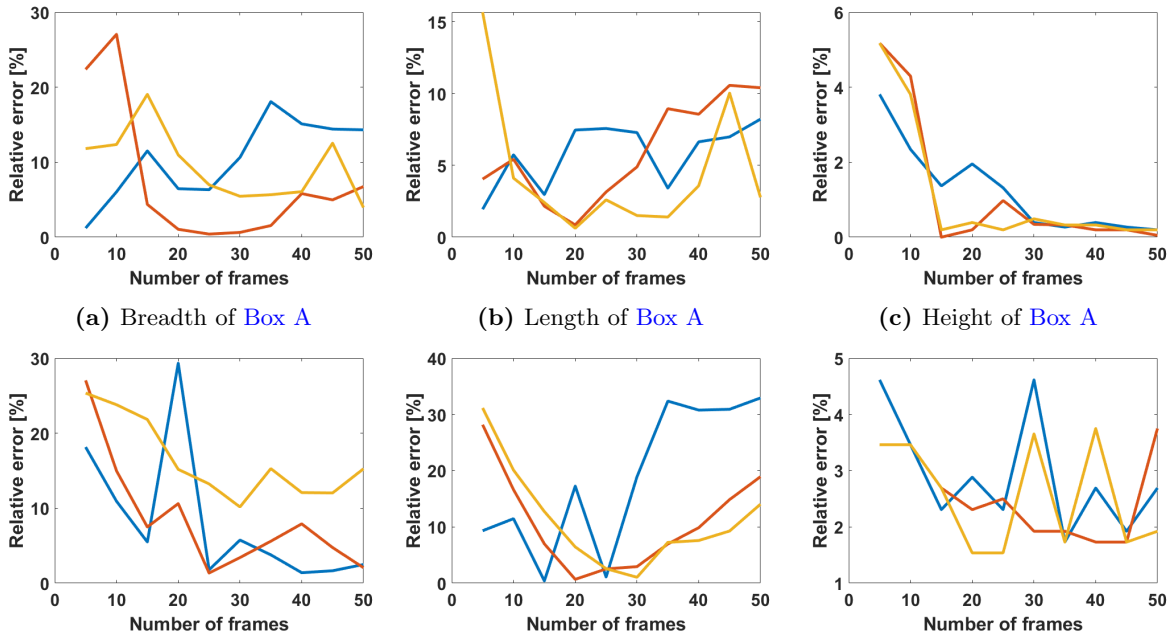
Once again, the efficacy of the Euclidean distance clustering algorithm can be qualitatively scrutinized in figure 7.6. Even when fifty point clouds were concatenated, the algorithm still successfully removed the ground plane and correctly segmented each object on the laboratory floor.



(a) Point cloud of the lab seen in figure 6.3b before clustering

(b) Point cloud of the lab seen in figure 6.3b after clustering

Figure 7.6: Figure showing the efficacy of the Euclidean distance clustering algorithm on the static physical data



(a) Breadth of Box A

(b) Length of Box A

(c) Height of Box A

(d) Breadth of Box B

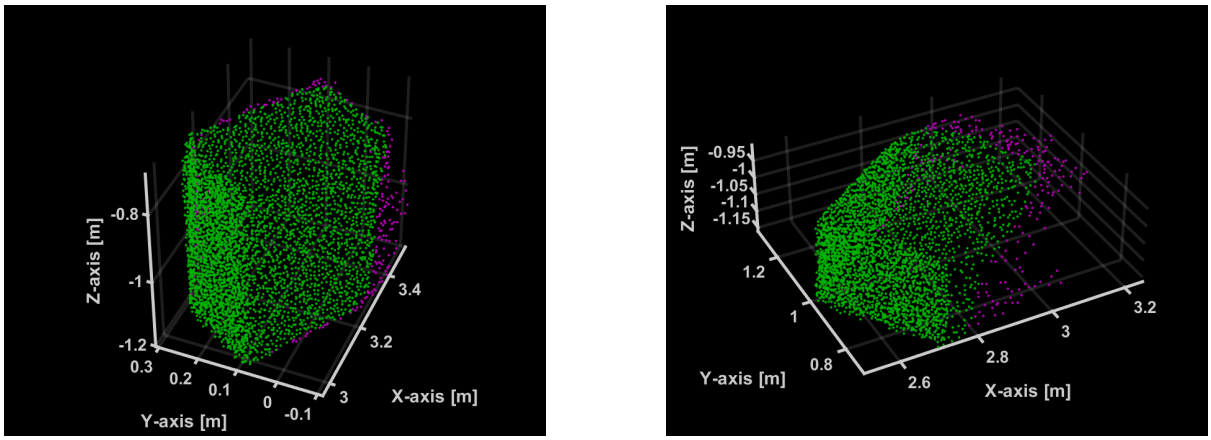
(e) Length of Box B

(f) Height of Box B

Figure 7.7: Relative error of the various dimensions of Box A and Box B as **numNeighbours** and **numFrames** variables are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

When comparing the relative errors of clustering algorithms on physical and simulated data, certain discrepancies and similarities can be identified. For the lengths and breadths of the two boxes above, it can be seen that the mild filter outperforms the medium and aggressive filters when the number of frames is less than ten. As the number of frames increases, so does the observed relative error in the mild filter's readings. Conversely, increasing frames decreases the error for the medium and aggressive filters until a point. As the number of frames increases further, the mild and aggressive filters increase, showcasing a trough shape. Noticeably, this trough shape is not present in the relative error of the heights of either box.

Additionally, it is observed that the relative error of the parameters of Box B behave more sporadically with increasing frames than what is observed in the behaviour of Box A . To gain additional insight into the causes of this behaviour, point clouds of segmented Box A and Box B are shown in figure 7.8.

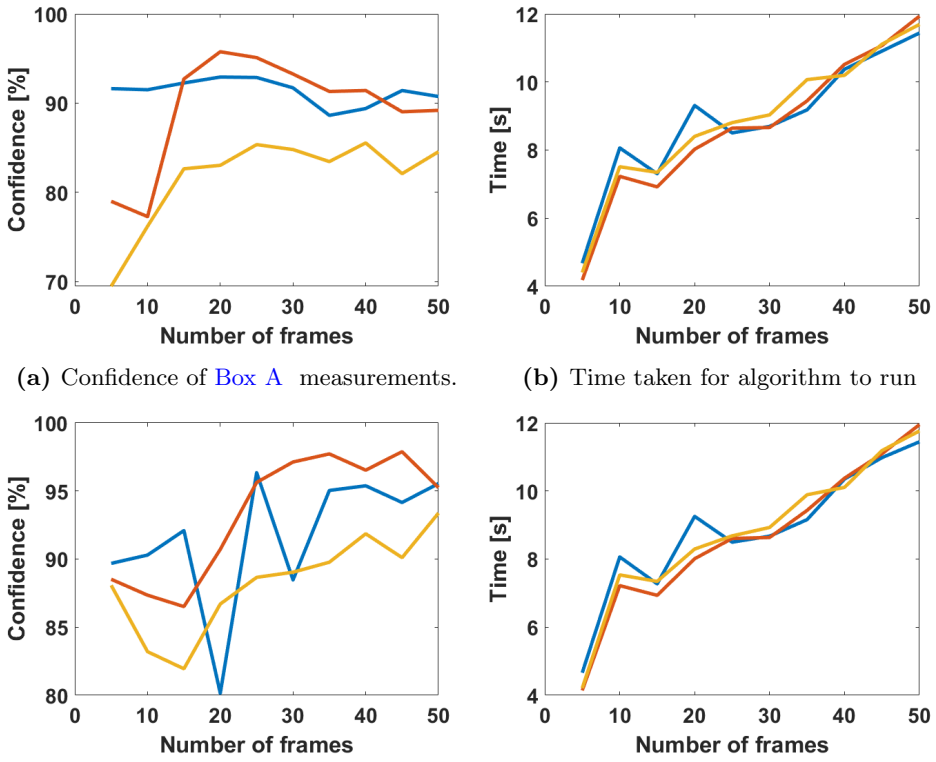


(a) Filtration process on Box A after 50 frames have been concatenated

(b) Filtration process on Box B after 50 frames have been concatenated

Figure 7.8: Figure qualitatively comparing Box A to Box B
Purple points represent filtered-out points

In figure 7.8, it can be seen that there are significantly more noise points surrounding the edge of Box B that is furthest from the LiDAR scanner as indicated by the cloud of purple points there.



(a) Confidence of Box A measurements.

(b) Time taken for algorithm to run

(c) Confidence of Box B measurements.

(d) Time taken for the algorithm to run

Figure 7.9: Confidence in Box A and Box B measurements and processing time as **numNeighbours** and **numFrames** variables are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters, respectively

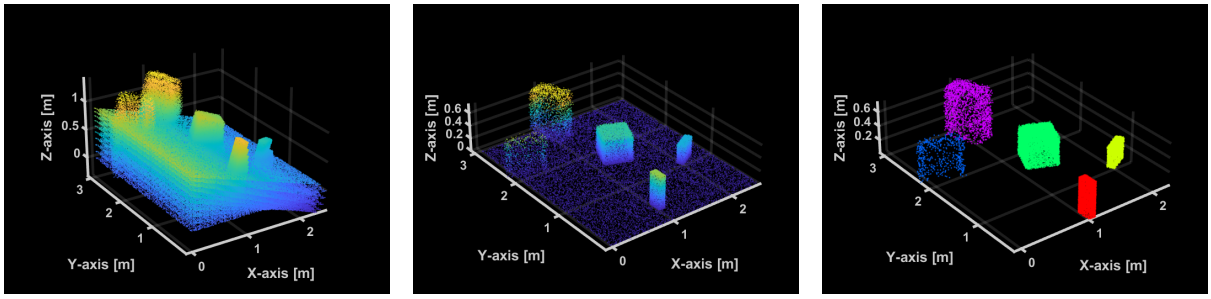
As depicted in figure 7.9, the time taken to execute the algorithm is nearly the same for both Box A and Box B for all three filters. They increase almost linearly until they reach a maximum processing time of 12 seconds[1] when 50 concatenated frames are used.

The increased volatility of the data captured of Box B can once again be identified in the confidence plots, especially with regard to the mild filter. It can be seen that the medium filter reaches a confidence of greater than 95% faster than the other two filters.

7.2 Motion compensation results

7.2.1 Simulation results

With the introduction of motion, the efficacy of the ICP motion compensation algorithm is first assessed qualitatively by comparing a series of concatenated frames before and after ICP compensation as shown in figure 7.10a and 7.10b. In this experiment, a 2-DoF motion was used, consisting of a heave (linear motion along the z-axis) and a pitch (angular rotation around the y-axis). The simulated IMU data is displayed in figure 7.11 along with the integrated signals to get them in the form of angular and linear positions.

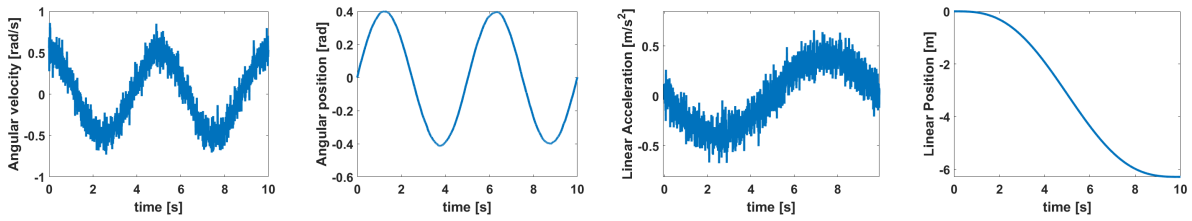


(a) Point cloud scene before ICP motion compensation

(b) Point cloud scene after motion compensation

(c) Point cloud scene after Euclidean clustering

Figure 7.10: Figure qualitatively showing the effect of the ICP motion compensation algorithm and the Euclidean clustering algorithm.



(a) Simulated angular velocity IMU data

(b) Simulated angular position after one integration wrt time

(c) Simulated linear acceleration IMU data

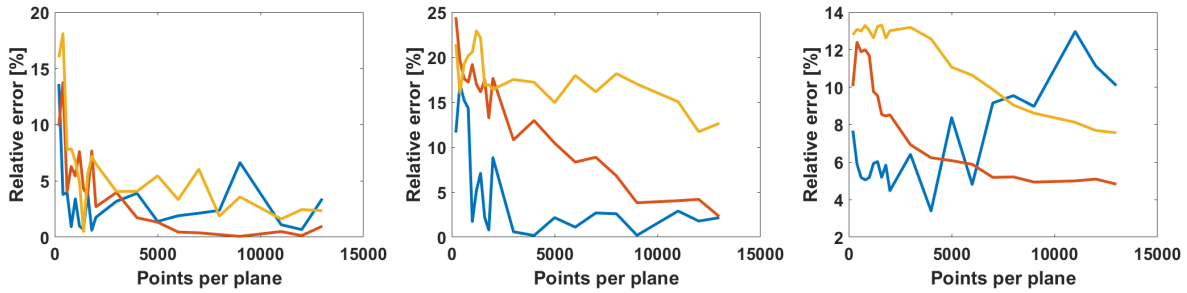
(d) Simulated linear position after two integrations wrt time

Figure 7.11: Figure showing the simulated IMU data along with the data after integration

In figure 7.10a, it can be seen that each point cloud frame is translated vertically and rotated compared to the next. After running this through the ICP algorithm, it is observed in figure

[1]It is essential to note that since the LiDAR scanner was stationary while capturing this data, there was no need to implement the ICP algorithm. This is reflected in the processing times.

7.10b that each point cloud frame is now stacked on top of one another, allowing for the segmentation algorithm to distinguish each of the objects as seen in figure 7.10c. Additionally, it is observed in figure 7.11 that simply integrating the signal concerning time removes the noise present in the IMU data.



(a) Breadth of simulated box **(b)** Length of simulated box **(c)** Height of simulated box.

Figure 7.12: Relative error of the various dimensions of a simulated box as **numNeighbours** and points per plane are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

After ICP compensation, it is observed that the feature extraction algorithm behaves almost identically as it did in the stationary simulation with regard to relative error and confidence.

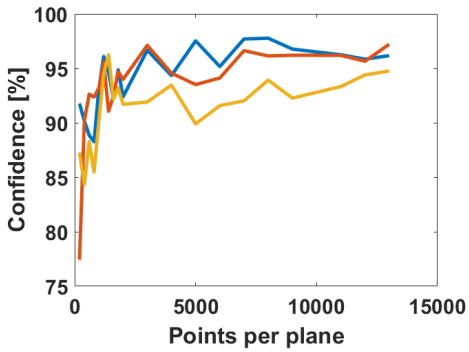
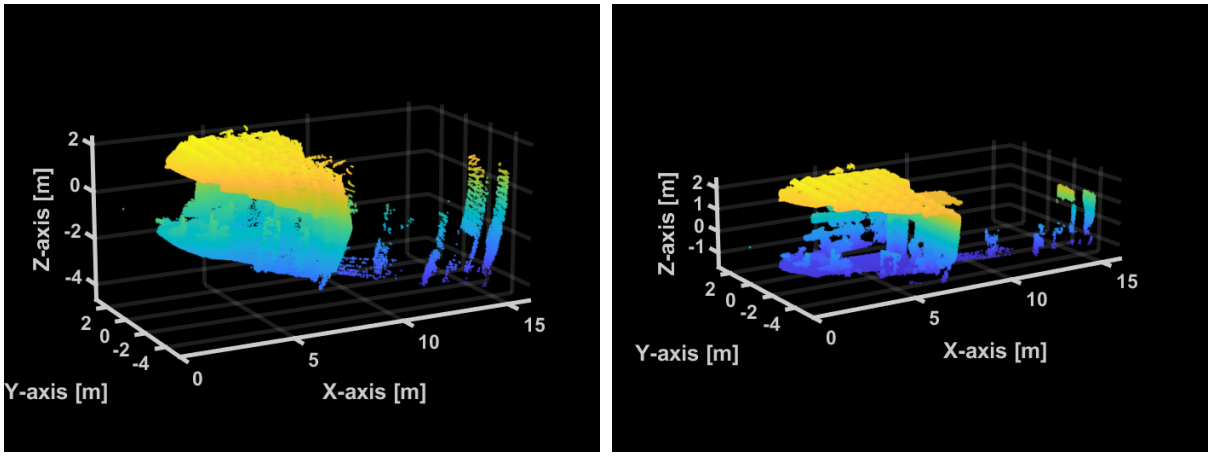


Figure 7.13: confidence in simulated box measurements as **numNeighbours** and points per plane variables are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

7.2.2 Physical experimentation results

As with the simulation, the efficacy of the ICP algorithm is first evaluated by comparing the point cloud frames of the lab before and after ICP compensation. This is shown in figure 7.14. It can be seen that before ICP compensation, the point cloud frames are fanned out, and following the ICP compensation, the frames are stacked on top of one another.

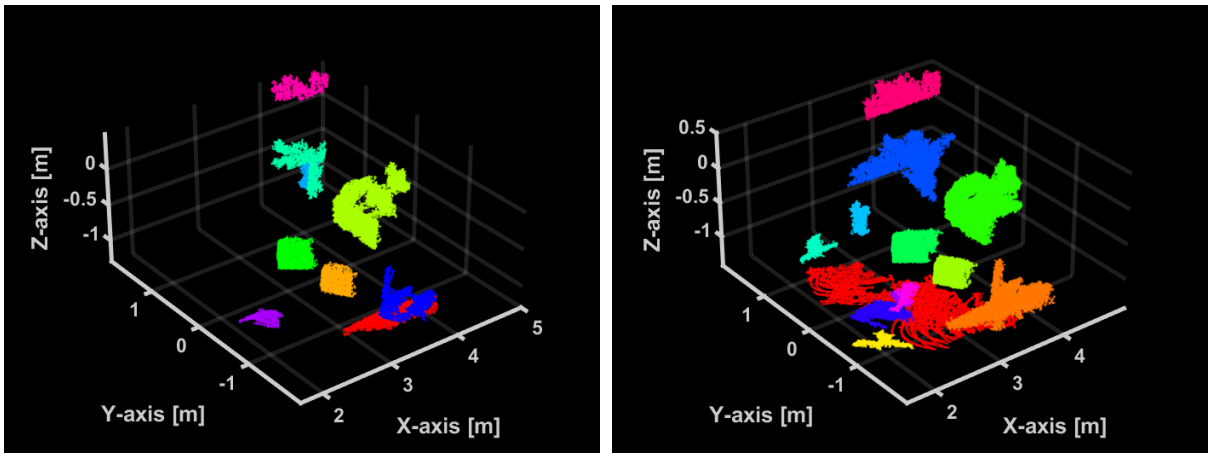
The aligning of the shifted point cloud frames allows for the successful segmentation of the point cloud as seen in figure 7.15a. However, when the number of frames approaches 50, some remnants of the ground plane are visible, as seen in 7.15b.



(a) 30 frame point cloud scene before ICP motion compensation

(b) 30 frame point cloud scene after motion compensation

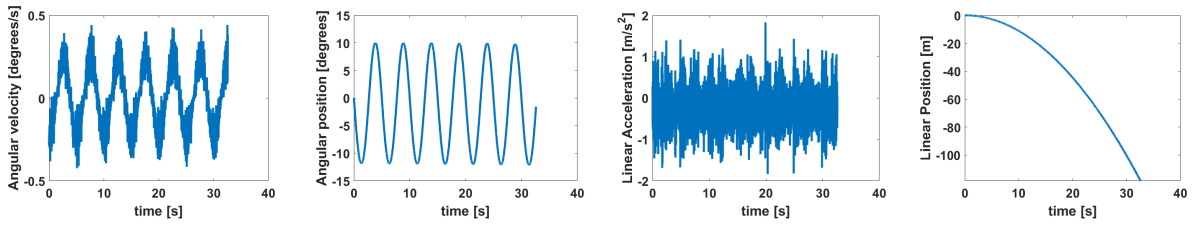
Figure 7.14: Figure qualitatively showing the effect of the ICP motion compensation algorithm.



(a) 30 frame point cloud scene after ICP motion compensation and Euclidean clustering

(b) 50 point cloud scene after motion compensation and Euclidean clustering

Figure 7.15: Figure qualitatively showing the effect of the ICP motion compensation algorithm and Euclidean clustering.



(a) Simulated angular velocity IMU data

(b) Simulated angular position after one integration wrt time

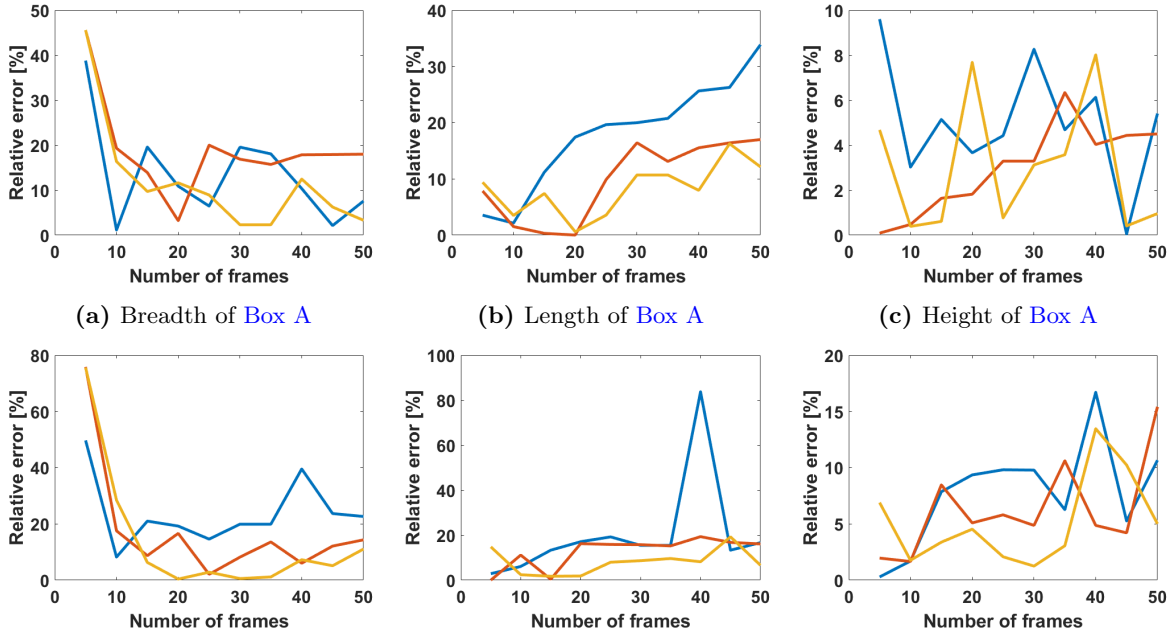
(c) Simulated linear acceleration IMU data

(d) Simulated linear position after two integrations wrt time

Figure 7.16: Figure showing the simulated IMU data along with the data after integration

The IMU readings corresponding to this motion can be seen in figure 7.16. Once again, it

appears that merely integrating the data with respect to time sufficiently removes the noise. However, it can be seen that the linear acceleration does not resemble the sinusoidal input to the Stewart platform. Instead, only noise is output by the IMU, resulting in a perpetually decreasing linear position. Consequently, the linear acceleration data was not used in the ICP algorithm for this section.

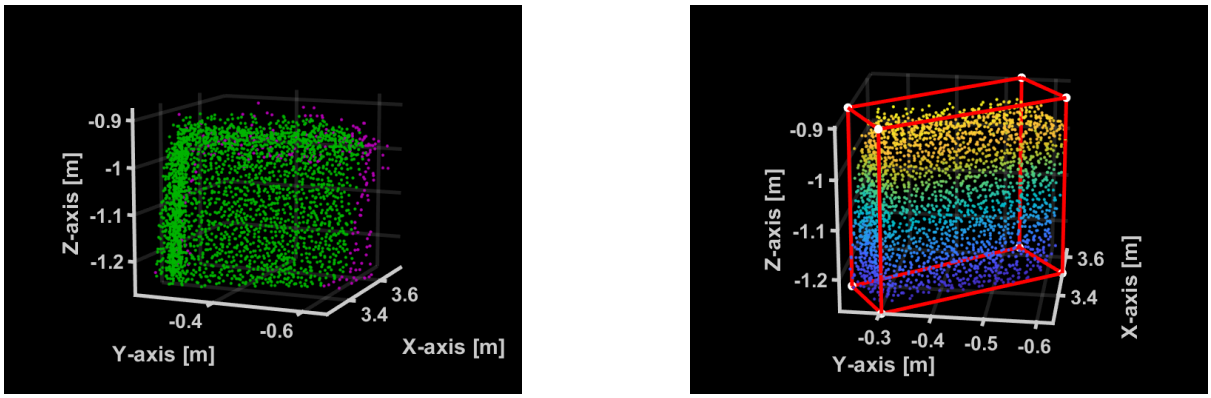


(a) Breadth of Box A **(b)** Length of Box A **(c)** Height of Box A

(d) Breadth of Box B **(e)** Length of Box B **(f)** Height of Box B .

Figure 7.17: Relative error of the various dimensions of Box A and Box B as **numNeighbours** and **numFrames** variables are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

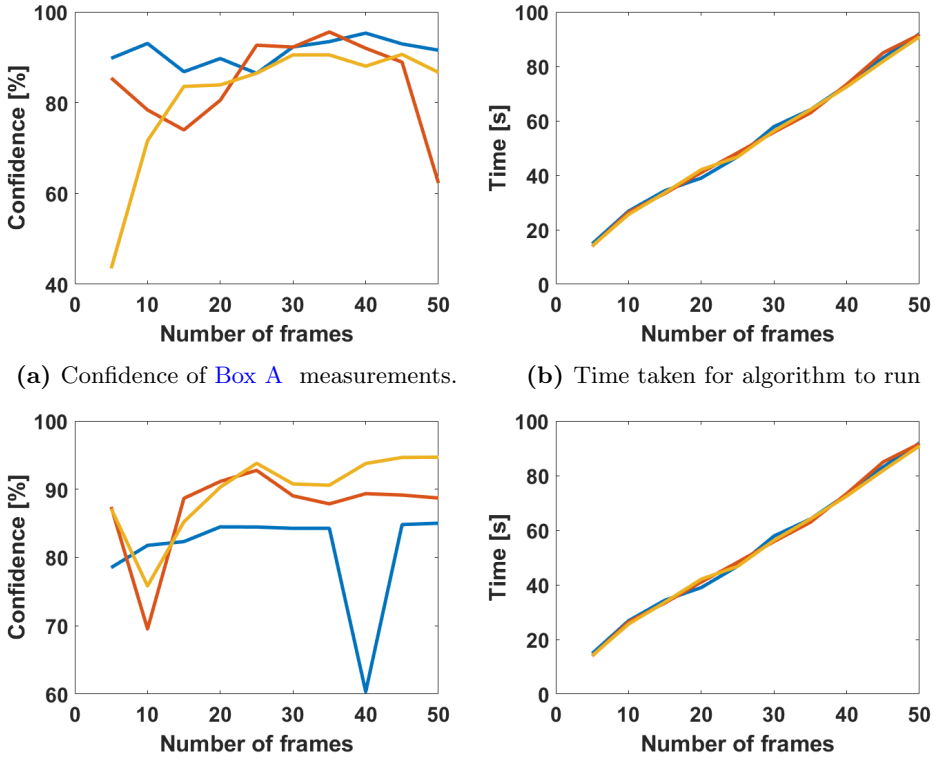
A few notable differences are found when comparing the relative error from the moving data to the stationary data. Firstly, it appears now as though the aggressive filter produces the slightest relative error even when the number of frames is low. Additionally, the sporadic behaviour visible in the relative error of the height of Box B in the stationary experiment is now observed in both boxes. To gain further insight into the causes of this, the effects of the filtration algorithm are shown in figure 7.18 below:



(a) Effect of aggressive filtering on Box A **(b)** Predicted model of Box A

Figure 7.18: Figure showing the effect of motion on the predicted model point cloud of Box A . Purple points represent filtered-out points.

It can be observed in figure 7.18a that there is a substantial degree of noise above the box that is not filtered out by the aggressive filter. Consequently, the model of the box estimated in 7.18b includes these noise points.



(a) Confidence of Box A measurements. **(b)** Time taken for algorithm to run

(c) Confidence of Box B measurements. **(d)** Time taken for the algorithm to run on

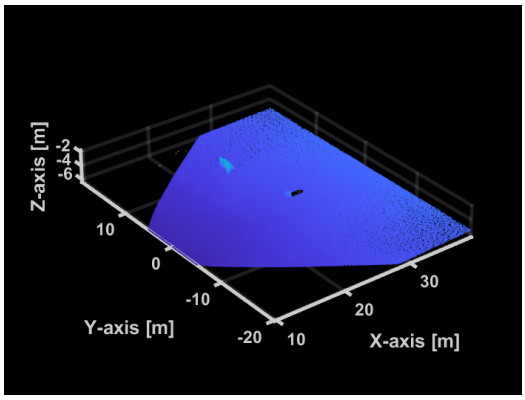
Figure 7.19: Confidence in Box A and Box B measurements and processing time as **numNeighbours** and **numFrames** variables are altered. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

Once again, the processing time is comparable between each filtration type, increasing linearly with the number of frames. However, it can be seen that with the inclusion of the ICP algorithm, the processing time is significantly longer than it was for the stationary experiments.

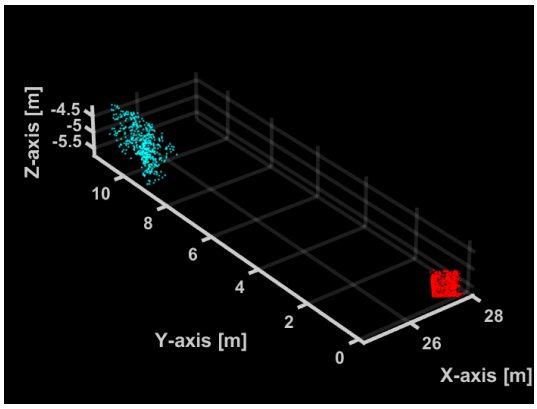
The confidence metric resembles that of the stationary experiments with a noticeable anomalous point occurring in figure 7.19c at 40 frames. Additionally, the confidence levels in both boxes seem to converge at 90% which is marginally less than the stationary results.

7.3 Range testing results

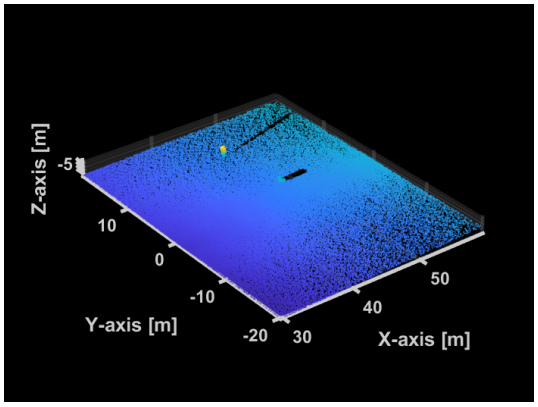
The algorithm's behaviour is now assessed as the target box is moved progressively further from the LiDAR scanner. Additionally, due to the increased separation between points at further distances, more frames may need to be concatenated to extract an accurate reading of Box C 's parameters. Consequently, a greater range of frames is considered, spanning from 10 to 100 for these experiments.



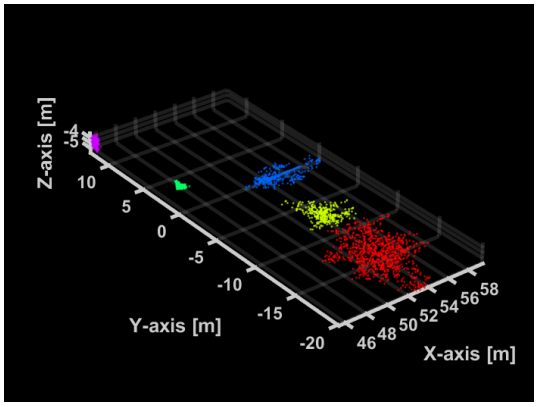
(a) Point Cloud of Box C placed at 20 m away from LiDAR before clustering



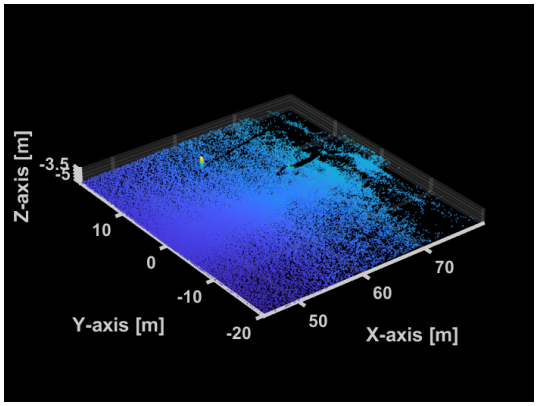
(b) Point Cloud of Box C placed at 20 m away from LiDAR after clustering



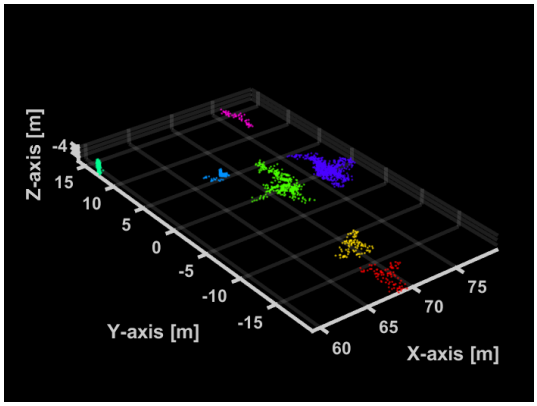
(c) Point Cloud of Box C placed at 40 m away from LiDAR before clustering



(d) Point Cloud of Box C placed at 40 m away from LiDAR after clustering



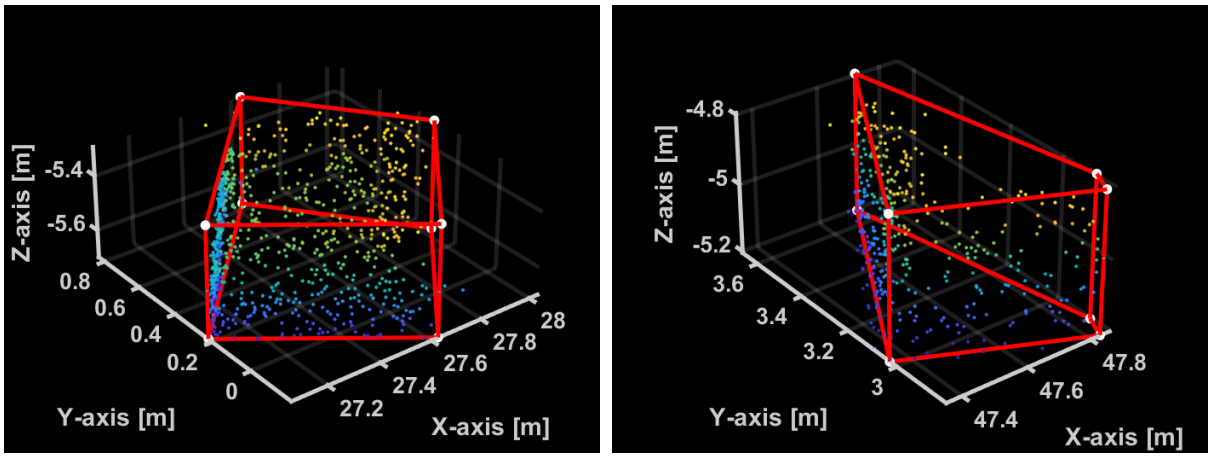
(e) Point Cloud of Box C placed at 60 m away from LiDAR before clustering



(f) Point Cloud of Box C placed at 60 m away from LiDAR after clustering

Figure 7.20: Figure showing the efficacy of the Euclidean distance clustering As the distance from the LiDAR to Box C is increased.

As seen in figure 7.20, as the box gets further from the LiDAR, more points from the ground plane remain after the data has undergone the clustering process. Additionally, with increasing distance, the density of the points in the point cloud decreases. This is particularly noticeable in the top left and right corners of the XY-plane of the figures 7.20a, 7.20c and 7.20e. The effect that this has on the estimated model of Box C is demonstrated more clearly in figure 7.21 which compares the estimated model at 20 m to the model estimated at 60 m.



(a) Estimated model of Box C placed at 20 m away from LiDAR

(b) Estimated model of Box C placed at 60 m away from LiDAR after clustering

Figure 7.21: Figure comparing the estimated model of Box C at 20 m to the model estimated at 60 m.

In analysing figure 7.21, it is observed that the decrease in point density is especially prevalent on the top surface of the box. This leads to the distorted estimated model visible in 7.21b.

As a side note, it can be seen that the height of the point cloud in 7.21a on the side furthest from the LiDAR is greater than the near side, as indicated by the yellow colouring of the points.

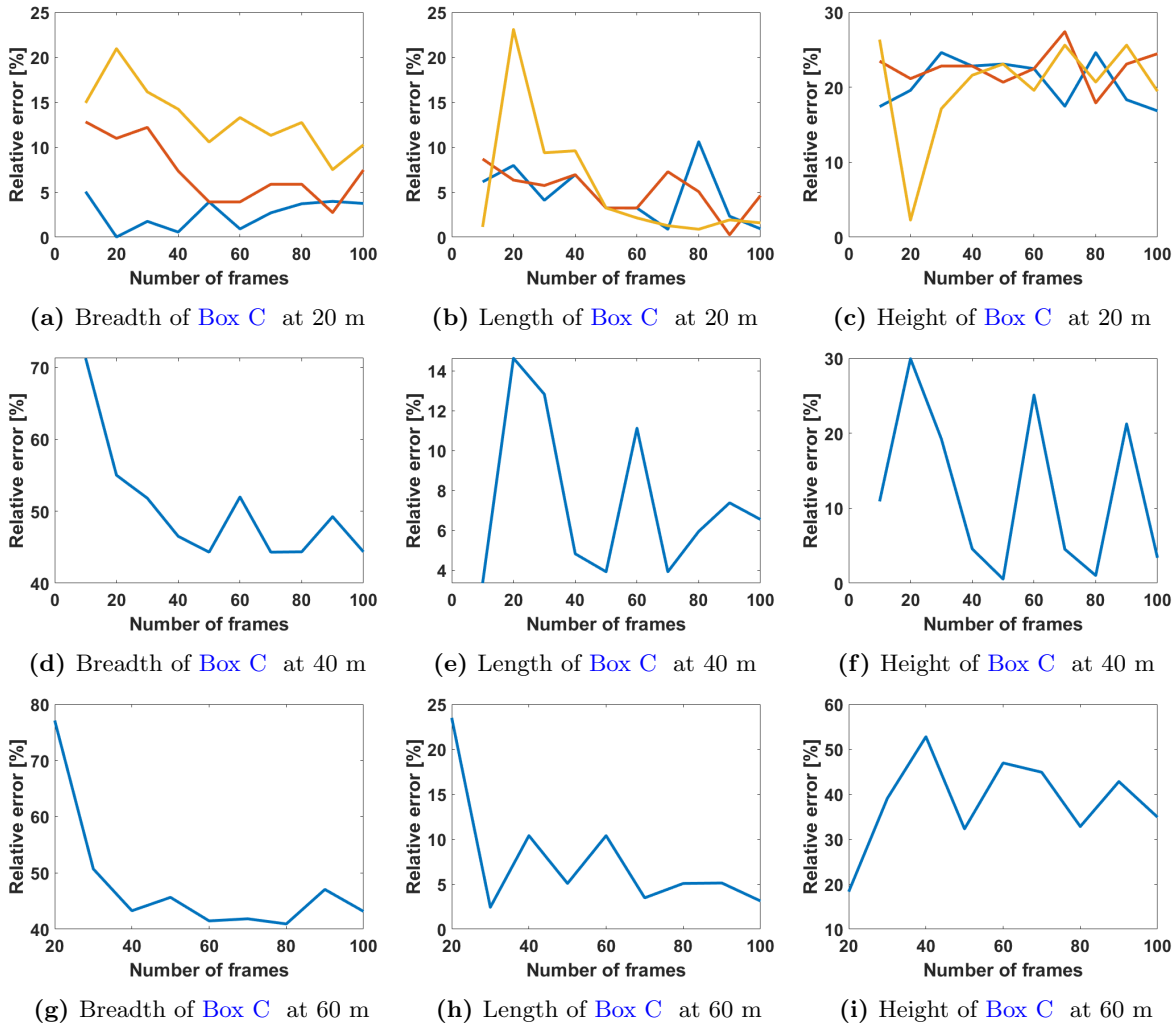
When analyzing the performance of the algorithm on the range testing data shown in figure 7.22, it was found that above 20 m, the medium and aggressive filters completely eradicate the entire point cloud of Box C . Thus, only the results produced by the mild filter were analysed for distances of 40 m and 60 m.

When considering the relative error of the breadth of Box C in figure 7.22a, it is observed that the mild filter reaches a minimum relative error at 20 frames before displaying a subtle increasing trend with increasing frames. Contrastingly, the medium and aggressive filters in figure 7.22a showcase a decreasing trend throughout.

It can be seen that between 20 m and 40 m, there is a significant increase in the relative errors of the model's predictions of the breadth of Box C . At 20 m, as the number of frames approaches 100, the relative error for all three filters tends to a value of less than 20%. It is also noted that the mild filter outperforms both the medium and aggressive filters at this distance for the entire range of frames. When the range is increased to 40 m, the lowest relative error that the algorithm converges to is 45%. This decrease in accuracy at greater distances is also observed, albeit to a lesser degree, in the length and height measurements of Box C .

Upon reviewing the height measurements of Box C , it becomes apparent that the algorithm's performance does not improve significantly when more frames are used. Additionally, there are noticeable instances of sporadic behaviour in the relative error measurements of Box C 's height

at a distance of 40 m.



(a) Breadth of Box C at 20 m **(b)** Length of Box C at 20 m **(c)** Height of Box C at 20 m

(d) Breadth of Box C at 40 m **(e)** Length of Box C at 40 m **(f)** Height of Box C at 40 m

(g) Breadth of Box C at 60 m **(h)** Length of Box C at 60 m **(i)** Height of Box C at 60 m

Figure 7.22: Relative error of the various dimensions of Box C as **numNeighbours**, **numFrames** variables and distance from the LiDAR scanner is increased. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

Finally, the confidence levels and processing times of the algorithm are inspected with respect to their behaviour as the distance to Box C is increased in figure 7.23.

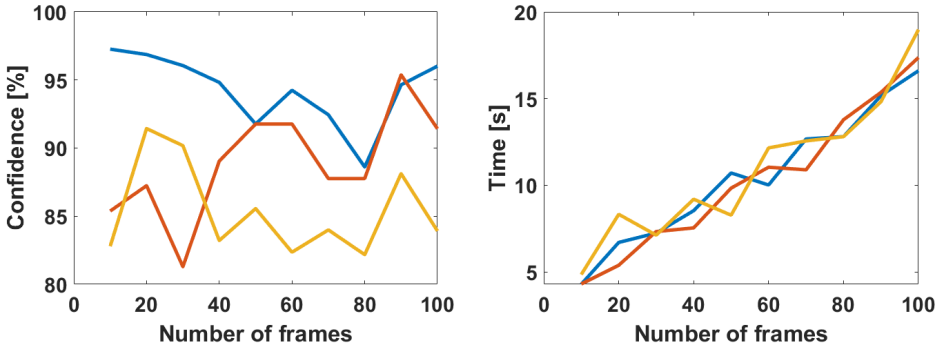
At 20 m, the mild filter shows the highest confidence in its measurements, averaging around a value of 95% confidence, whilst the medium filter starts at a confidence of 85%, gradually increasing to a value of approximately 90% at 100 frames. In contrast, the aggressive filter also begins at a confidence of 85% but does not improve as the number of frames increases.

The time taken for the algorithm to run at this distance is again similar for all three filters, showing linear behaviour with increasing frames. The behaviour persists regardless of the distance to the LiDAR scanner.

At 40m, the confidence level of the algorithm starts at a lower value of 40% before rapidly increasing to 70% at 30 frames and plateauing, showing no change up to 100 frames.

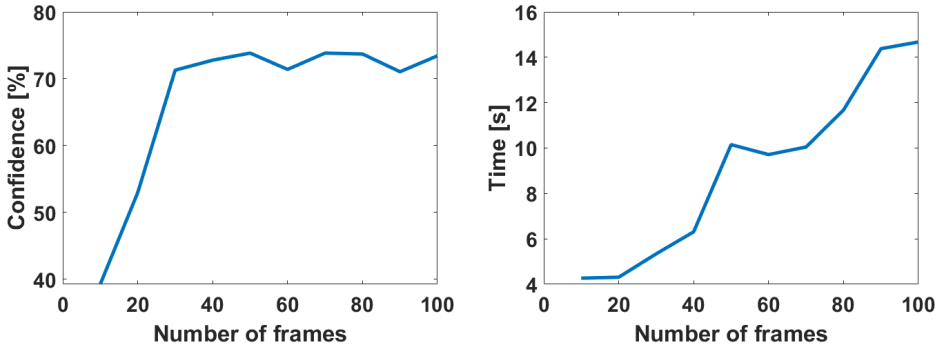
Interestingly, the confidence level starts at around 80% at 60 m before decreasing to a value of

60% at 100 frames. This is the first time the algorithm has produced a decreasing confidence level as the number of frames has increased.



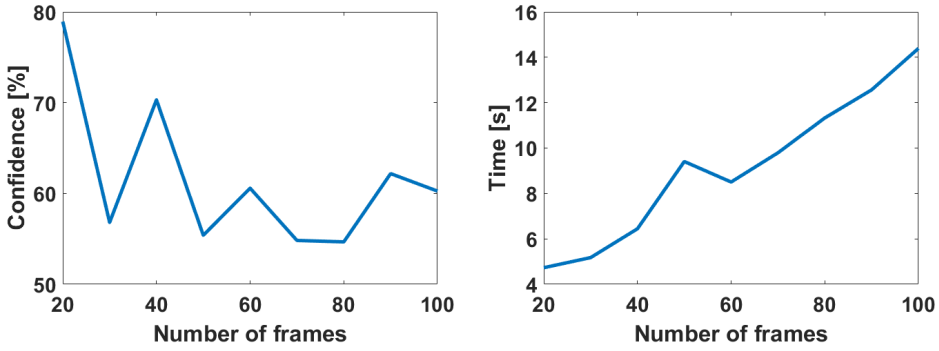
(a) Confidence of Box C measurements at 20 m.

(b) Time taken for algorithm to run at 20 m



(c) Confidence of Box C measurements at 40 m.

(d) Time taken for algorithm to run at 40 m



(e) Confidence of Box C measurements at 60 m.

(f) Time taken for algorithm to run at 60 m

Figure 7.23: Confidence in Box C measurements and processing time as **numNeighbours**, **numFrames** variables and distance from the LiDAR scanner is increased. The blue, red and yellow lines correspond to the mild, medium and aggressive filters respectively

Chapter 8

Discussion and analysis of algorithm performance

In Results and observations, the data collected throughout all experiments was displayed along with observations concerning the trends and anomalies on display. This chapter aims to discuss these results, critically analysing the efficacy of the processing pipeline by highlighting the areas in which it succeeded or failed. Additionally, explanations are provided regarding any abnormal behaviour observed in the results.

For clarity, this discussion is divided into different sections, beginning with the efficacy and observed shortcomings in the RANSAC and Euclidean Clustering algorithms.

8.1 Ground plane removal and Euclidean clustering

For both the stationary and dynamic simulated data and the stationary physical data, the RANSAC algorithm could successfully remove the entirety of the ground plane in the point cloud data. However, parts of the ground plane were still visible in the motion compensation physical data after being processed by the RANSAC algorithm. These ground plane points could potentially cause the Euclidean clustering algorithm to treat two separate objects as a singular object connected via the escaped ground plane points. These remaining ground plane points are attributed to imperfect frame alignment following the implementation of the ICP motion compensation algorithm.

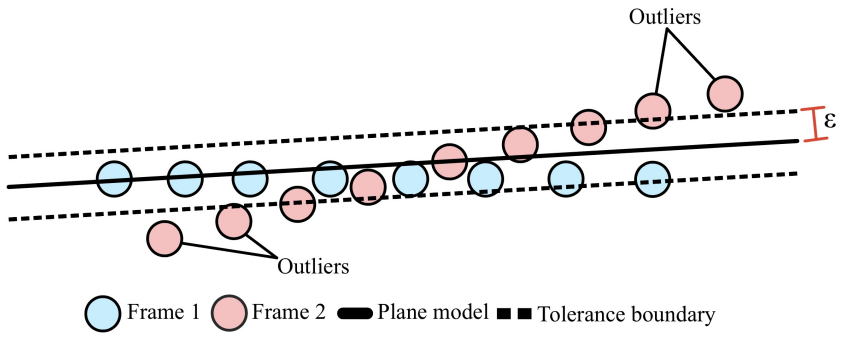


Figure 8.1: Figure demonstrating how point cloud misalignment leads to unsuccessful ground plane extraction

As demonstrated in figure 8.1, if the ICP algorithm fails to align two consecutive point cloud frames completely, the RANSAC algorithm will fit a plane to data that contains the most possible points from *either* plane. This results in a situation in which the fitted plane is not perfectly fitted to either point cloud, resulting in the outliers observed in figure 8.1. These outliers will escape the RANSAC algorithm, as was observed in the data.

This effect could be mitigated by increasing the size of the tolerance boundary (ϵ). However, the larger the value of (ϵ), the more the base of objects will be extracted along with the ground plane. Thus, this solution may only be effective if the object's height is large enough relative to the value of (ϵ). Alternatively, the number of iterations used by the ICP algorithm can be increased to achieve a better point cloud alignment. However, this would come at a cost of processing time.

With that said, ground plane points also escaped the RANSAC algorithm from the range testing data, which was captured from a *stationary* reference frame. Consequently, this behaviour cannot be attributed to the failures of the ICP algorithm. When analysing the range testing point cloud data, it was observed that the points from the ground plane only escaped the RANSAC algorithm at distances of 40 m and greater. This information suggests that these outlying ground plane points can be attributed to the angular random error of the Livox Avia referenced in table 5.1.1.

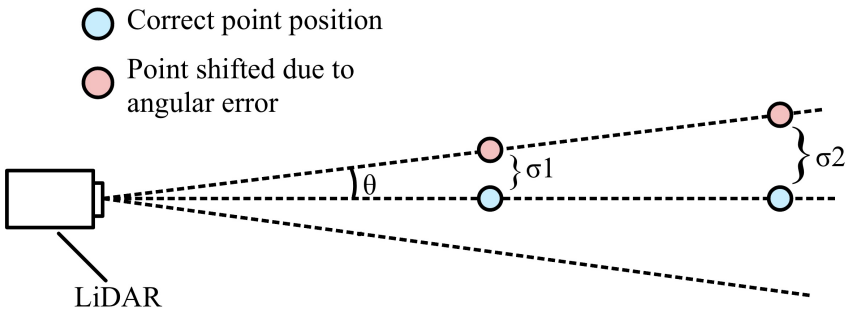


Figure 8.2: Figure demonstrating how random angular error affects points further for the LiDAR more significantly than closer points

As demonstrated in figure 8.2, even if the random angular error (θ) is small, the displacement of points due to random angular error (σ) can be significant if the point is sufficiently far enough from the LiDAR.

Beyond the ground points escaping the RANSAC algorithm, the Euclidean clustering algorithm satisfactorily segmented the scene into its various objects throughout each of the conducted experiments.

The phenomenon highlighted in figure 8.2 may also explain the difference in the behaviour of the algorithm on Box A versus Box B . The length of the Box B is greater than that of Box A , and thus the displacement due to random angular error is more significant as it is further from the LiDAR. Similarly, this may explain why the far side of Box C in figure 7.21a is higher than the point located closer to the LiDAR.

8.2 Motion compensation

When qualitatively analysed, the ICP motion compensation algorithm appears to correctly align the shifted point clouds in both the simulated and physical data as seen in figures 7.10 and 7.14 respectively. This seems to be reflected in the data of the motion simulation, which closely resembles that of the stationary simulation. However, this is not true for the physical data, which is noticeably more sporadic and inaccurate than the data captured during the physical stationary experiments. This suggests that some artefacts due to the motion of the LiDAR are escaping the ICP algorithm and are being reflected in the data.

To obtain a better understanding of what these artefacts may be, figure 7.18 is analysed. It can be seen that above the top plane of the box, there are what appear to be densely packed noise points that have not been removed by the filter. It is speculated that these are not noise points but rather points from point cloud frames that are not fully aligned. Consequently, these points are dense enough to avoid filtration, thus leading the algorithm to produce incorrect height measurements. This explains why the height measurements of Box A become less accurate and more sporadic from the stationary physical experiments.

The question as to why the ICP algorithm worked in simulation but not in reality remains. This is attributed to two primary factors. Firstly, each of the shifted frames in the simulation has perfectly corresponding points, meaning that they can be perfectly aligned. This is not the case in reality. Two consecutive LiDAR scans taken from a moving reference frame will not only have differing distributions of points along the objects shared in both frames but will also contain points corresponding to new objects that are not in view in both frames. These discrepancies between two consecutive frames may impede the ICP algorithm's ability to align them, thus resulting in the error observed in 7.18.

8.3 Filtering and number of frames

The number of neighbouring points used in the Point cloud denoising algorithm, along with the number of concatenated point cloud frames, were the two primary hyperparameters which were altered throughout the course of the project.

Generally, throughout the data, it was observed that the results were jagged, showing spikes and troughs between each increase in the number of frames used. This behaviour may be attributed to the fact that each additional frame may completely change what is considered noise by the Point cloud denoising algorithm. Figure 8.3 elaborates on this point by showcasing how the addition of points from another point cloud may bridge the gap between the object and what was previously considered noise points by the Point cloud denoising algorithm. This results in these noise points now being considered part of the object, which could drastically impact the model estimates. Conversely, points could be added that increase the density of points in certain areas on the object's surface, which could cause other points in other areas on the object's surface with less density to now be considered noise. In this fashion, adding certain frames may cause the model estimates to worsen, and adding more may make it improve, thus resulting in the jagged plots seen in the results.

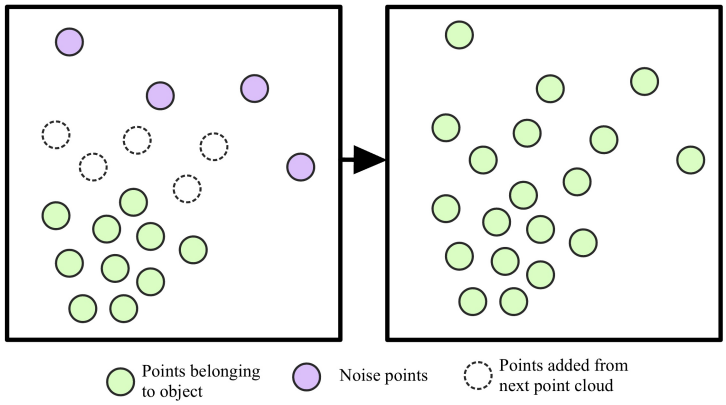


Figure 8.3: Figure showing how the addition of noise points from a subsequent point cloud frame impacts the parameter extraction process

Despite the sporadic data, a general trend was observed in the data; at lower numbers of frames, the mild filter would outperform the medium and aggressive filter, producing a smaller relative error in its estimates. However, as the number of concatenated point cloud frames increased, the mild filter's performance would worsen, whilst the other two filters would show improvement. In some cases, this improvement would halt, eventually leading to an increase in relative error again at an even higher number of frames. In other cases, the medium and aggressive filters would show downward trends in relative error up to 50 frames, suggesting that should the number of frames increase, so would their performance.

The proposed explanation behind this behaviour can be best explained in reference to figure 7.4, which compares the impact of the mild versus the aggressive filter on both low and high-density point clouds. Figure 7.4 shows that for low-density point clouds, the mild filter successfully removes the noise points, leaving the structure of the box intact. However, for high-density point clouds, some of these noise points are densely situated, thus escaping the filter and leading to an inaccurate predicted object model. Conversely, for low-density point clouds, the aggressive filters will remove the noise points along with some points belonging to the *object* as well. For higher-density point clouds, the points along the object's surface are densely packed enough to avoid filtration whilst the noise points are still removed. In this way, the discrepancy between the performance of the filters with increasing frames can be explained.

In the physical and stationary data, the medium filter produced the lowest relative errors across length, breadth and height at around 20 concatenated frames. It is, therefore, recommended that these hyperparameters be used for the algorithm when the target objects are at these ranges.

However, at more considerable distances, the mild filter outperformed the medium and aggressive filters, and at distances further than 40 m, it was the only feasible option out of those available. Even so, the estimates produced at these more considerable distances were very poor, with relative errors as high as 40%. This indicates an area that needs improvement in order for the algorithm to be deployed at the larger distances synonymous with ship-based data capture.

It was observed that when the filter did remove points belonging to the actual object, it was always points from the side of the object that was furthest from the LiDAR scanner. The explanation behind this phenomenon is best conducted in reference to figure 8.4. By modelling the LiDAR's visual field as a cone with rays leading to points that will form the point cloud, it can be seen that both the distance and angle between the LiDAR ray and the surface normal of the surface of the object impact the density of points found in the resulting point cloud. As seen

in figure 8.4, the density of points is reduced if the object is far from the LiDAR scanner and if its surface normal is almost perpendicular to the rays of the LiDAR scanner. Additionally, this effect will be further exacerbated by the nature of the Livox Avia scanning pattern observed in figure 5.1a which shows differing densities across the field of view of the Livox Avia.

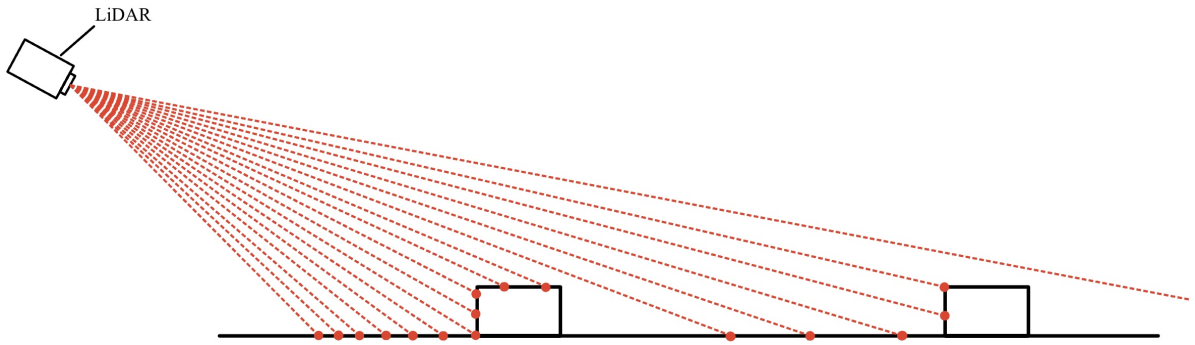


Figure 8.4: Figure demonstrating how objects further from the LiDAR have lesser point density than closer objects

In this diagram, it can be seen that objects towards the periphery of the Livox Avia's visual field will have a lower density than points located at its centre. This can clearly be seen in figure 7.20 as the regions corresponding to larger y-values in the x-y plane has lower density than those close to $y = 0$.

This explains why the points along the far edge of objects are removed by the filters first, as the density at these locations will be less. This also explains why the relative error plots differ from the length to breadth to height, as the object's orientation with respect to the LiDAR has a significant impact on its density of points.

Additionally, this phenomenon explains why, in figure 7.21, the top surface of the box is not present when it is further away from the LiDAR. This highlights a weakness in the feature extraction algorithm as it requires at least three of the surfaces to be present in the point cloud data to produce its estimates correctly.

8.4 Confidence

The confidence metric, built into the feature extraction component of the algorithm, was intended to give the user an idea of how confident the algorithm was that its estimates of an object's parameters were accurate. As such, one would expect to see an inverse correlation between the relative errors and confidence metric. This relationship was not evident in the data. Frequently, the algorithm would report confidence levels of 90% and upwards even if the relative error measured in the parameters of the object were as high as 20% or 30%. Additionally, filters with the largest relative error at a given number of frames did not produce the lowest levels of confidence.

Since the confidence metric in the algorithm is derived from the degree to which the estimated model of an object represents a rectangular prism, it would seem that the noise points present in the point clouds still preserve the rectangular prism geometry. Consequently, the model produces high levels of confidence even if plenty of noise points are included in the model estimate.

8.5 Processing time

The intention behind including processing time in the results of this report was to obtain an idea of how the computational intensity of the algorithm changed with hyperparameters. This data can aid in optimizing the processing pipeline discussed in this report for deployment on hardware with less processing power than the laptop used in the study.

To this end, a minimal distinction was found between the processing times of each of the filter types used throughout testing. This suggests that if need be, one could increase the number of neighbouring points considered by the Point cloud denoising algorithm without impacting the processing time of the algorithm dramatically.

Across all the data, the processing time increased linearly with the number of frames used. The ICP algorithm adds an additional 8 seconds to the processing times of the algorithm as well.

With these sorts of processing times, it is unlikely that this algorithm could run on real-time data in the field.

Chapter 9

Conclusions

The overarching goal of this project was to develop a crude first iteration of a processing pipeline capable of extracting the size and shape parameters of the sea ice floes in the MIZ of the Antarctic region. In taking the initial steps to accomplish this goal, this project's scope was limited to extracting parameters for rectangular prisms placed on a flat surface in a stationary reference frame. Only the LiDAR was subject to motion within this report. This project aims to create the skeleton of the pipeline such that each component can be improved upon at a later date.

Informed by the associated literature, a processing pipeline was established, which comprised two significant components: feature extraction and motion compensation. The feature extraction algorithm utilized mathematical tools such as RANSAC, Euclidean clustering, point cloud denoising and a technique of extracting parameters from a rectangular prism-shaped point cloud. A confidence metric was included in this algorithm to provide users insight into the suspect accuracy of a given measurement. These specific mathematical tools were chosen as they seemed to complement the simplified nature of the project scope.

The motion compensation component of the algorithm comprised an ICP algorithm that used surface normals in the point cloud to improve further its ability to register two point clouds. Additionally, IMU data was used to create an initial transformation to assist in the ICP algorithm.

To test the efficacy of this algorithm, three experiments were run, each designed to test a specific component of the processing pipeline. Each test was run in simulation before moving to physical experimentation with a Livox Avia LiDAR scanner. The first of these tests aimed at isolating the feature extraction component of the algorithm. This involved placing the LiDAR in a stationary reference frame in a scene containing multiple boxes of known dimensions, the parameters of which the feature extraction algorithm must obtain.

Secondly, to test the motion compensation algorithm, the LiDAR was attached to a Stewart platform set to perform sinusoidal motion with regard to pitch about its y-axis and a heave along its z-axis. Once again, the algorithm was tasked with extracting the parameters of the boxes with the added nuance of compensating for any artefacts introduced into the data as a result of the motion.

Lastly, the impact of range on the processing pipeline is assessed. To achieve this, an experiment similar to the first one was run, only it was repeated with the target box at increasing distances from the LiDAR scanner.

A mild, medium and aggressive filter was applied to the data over a range of concatenated frames. Each of these variations of the processing pipeline was then assessed concerning the relative error produced in each of the extracted parameters, as well as the confidence metric it produced and the pipeline's processing time.

In the results, it was found that in the stationary and movement experiments, a medium filter operating on approximately 20 concatenated point cloud frames offered the smallest relative error in its estimates.

However, at further distances, the algorithm performed significantly worse. This may be problematic given that characterizing sea ice will often be required at the sort of distances used in the range testing experiments. It was found that more frames improved the performance, as at a more considerable distance, the point clouds of objects were found to be sparsely populated. Additionally, it was found that at greater distances, the mild filter outperformed the medium and aggressive filters concerning relative error.

Additionally, it was found that at longer distances, angular error caused some ground plane points to escape the RANSAC algorithm.

Similarly, in evaluating the motion compensation algorithm, it was found that at high frame count, points of the ground plane were escaping the RANSAC algorithm as well. However, this is suspected to result from imperfect point cloud registration in this context.

The confidence metric was found to represent the suspected accuracy of the algorithm's accuracy poorly, and thus, a revised method is proposed in Further work.

Overall, this project accomplished its goal of establishing the first iteration of a processing pipeline capable of characterizing Antarctic sea ice. Although it failed in some areas, the findings of this report will prove valuable with regard to informing the potential future work that can be conducted, as discussed in the subsequent chapter.

Chapter 10

Further work

Throughout this project, ideas and opportunities for further development and improvement became apparent. This chapter discusses some of the logical progressions that can build upon this body of work. Additional work is needed to improve the algorithm in areas where it failed, as well as to take the next steps towards achieving the project's overarching goals of sea ice characterization in the Marginal Ice Zone surrounding the Antarctic continent. This discussion is divided into sections regarding the different areas of the processing pipeline.

10.1 Filtering

In the results of the experiments, it was found that at shorter distances, the pipeline equipped with a medium filter produced the lowest relative error. However, as the distance from the LiDAR to the target object grew, the mild filter outperformed the medium and aggressive filters. This behaviour suggests that a filter which becomes increasingly less aggressive the further the target object is away from the LiDAR would enable the algorithm to extract parameters from objects from a wide range of distances in a given LiDAR scene. Exactly how to implement a situational filter of this sort is certainly an area worth further investigation.

10.2 Confidence metric

The confidence metric developed in this report failed to provide the user with a gauge of how trustworthy a given measurement is. As discussed, this was because the metric was based on to what degree the object represented a rectangular prism. It was found that noise points did not substantially change the geometry of the object, resulting in high errors corresponding to increased confidence. Thus, it would be desirable to develop a confidence metric that produces high confidence with low error. It is suspected that a model that takes into account the distribution of points into account concerning outliers would be a potential option worth investigating.

10.3 Reduce simplifying assumptions

With the general sequence of steps defining the processing pipeline outlined in this report, the task of reducing the number of simplifying assumptions made in each submodule remains.

For example, the RANSAC algorithm used for ground plane removal will need to take into account that the ocean surface is not flat. Additionally, the motion compensation algorithm will need to account for the motion of the objects as well as the LiDAR sensor itself. Furthermore, the feature extraction algorithm developed in this report can only extract the parameters of box-shaped ice blocks, the kind which are very rarely found in nature. Consequently, a more sophisticated feature extraction algorithm is required, one capable of extracting the parameters of more organic shapes.

The talking points discussed above are intended as a guide to direct the research and future work conducted in furthering this project.

Appendix A

Full hardware and software specifications

Livox Avia	
Laser Wavelength	905 nm
Laser Safety	Class 1 (IEC 60825-1:2014) (Safe for eyes)
Detection Range (@ 100 klx)	190 m @ 10% reflectivity 230 m @ 20% reflectivity 320 m @ 80% reflectivity
Detection Range (@ 0 klx)	190 m @ 10% reflectivity 260 m @ 20% reflectivity 450 m @ 80% reflectivity
FOV	Non-repetitive scanning pattern: 70.4° (horizontal) × 77.2° (vertical) Repetitive scanning pattern: 70.4° (horizontal) × 4.5° (vertical)
Distance Random Error	1σ (@ 20 m) < 2 cm
Angular Random Error	1σ < 0.05 °
Beam Divergence	0.03° (horizontal) × 0.28° (vertical)
Point Rate	240,000 points/s (first or strongest return) 480,000 points/s (dual return) 720,000 points/s (triple return)
Data Latency	≤ 2 ms
Data Port	100Mbps Ethernet
Data Synchronization	IEEE 1588-2008 (PTP v2), PPS (Pulse Per Second), GPS (PPS+UTC)
False Alarm Ratio (@ 100 klx)	< 0.0003%
IP Rating	IP67
IMU	Built-in IMU model: BMI088
Operating Temperature Range	-20° to 65° C (-4° to 149° F)
Storage Temperature Range	-40° to 85° C (-40° to 185° F)
IP Rating	IP67
Power	Repetitive scanning pattern: 9 W (Startup: 16 W) Non-repetitive scanning pattern: 8 W (Startup: 16 W)
Power Supply Voltage Range	Livox Avia: 10-15 V DC (recommended 12 V DC and 30 W or higher) Livox Converter 2.0: 9-30 V DC
Noise	40cm omnidirectional <45 dBA
Dimensions	9161.264.8 mm
Weight	Approx. 498 g (without cables)
Power Supply Voltage Range	9-30 V DC
Dimensions	74×52×23 mm
Weight	88 g

Table A.1: Full specifications of Livox Avia [37]

Parameter	Symbol	Condition	Min	Typ	Max	Units
Acceleration Range	g_{FS3g}	Selectable via serial digital interface		±3		g
	g_{FS6g}			±6		g
	g_{FS12g}			±12		g
	g_{FS24g}			±24		g
Sensitivity	S_{3g}	g_{FS3g}, T_A=25°C		10920		LSB/g
	S_{6g}	g_{FS6g}, T_A=25°C		5460		LSB/g
	S_{12g}	g_{FS12g}, T_A=25°C		2730		LSB/g
	S_{24g}	g_{FS24g}, T_A=25°C		1365		LSB/g
Sensitivity Temperature Drift	TCS			0.002		%/K
Zero-g Offset	Off	Nominal VDD and VDDIO, 25°C, g_{FS6g}		20		mg
Zero-g Offset Temperature Drift	TCO			<0.2		mg/K
Output Data Rate	ODR		12.5		1600	Hz
Bandwidth range	BW	3dB cut-off frequency of the accelerometer depends on ODR and OSR	5		280 (245 for Z axis)	Hz
Nonlinearity	NL	best fit straight line, g_{FS3g}		0.5		%FS
Output Noise Density	n_{rms}	g_{FS3g}, T_A=25°C Nominal VDD supplies Normal mode		190 (Z-axis) 160 (X- & Y-axis)		µg/√Hz
Cross Axis Sensitivity	S	relative contribution between any two of the three axes		0.5		%
Alignment Error	E_A	relative to package outline		0.5		°

Table A.2: Full specifications of BMI088 accelerometer [36]

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Range	R_{FS125}	Selectable via serial digital interface		125		°/s
	R_{FS250}			250		°/s
	R_{FS500}			500		°/s
	R_{FS1000}			1000		°/s
	R_{FS2000}			2000		°/s
Sensitivity		Ta=25°C, R_{FS125}		262.144		LSB/°/s
		Ta=25°C, R_{FS250}		131.072		LSB/°/s
		Ta=25°C, R_{FS500}		65.536		LSB/°/s
		Ta=25°C, R_{FS1000}		32.768		LSB/°/s
		Ta=25°C, R_{FS2000}		16.384		LSB/°/s
Sensitivity tolerance		Ta=25°C, R_{FS2000}		±1		%
Sensitivity Change over Temperature	TCS	Nominal VDD supplies $-40°C \le T_A \le +85°C$ R_{FS2000}		±0.03		%/K
Sensitivity Supply Volt. Drift	S_{VDD}	T_A=25°C, $VDD_{min} \le VDD \le VDD_{max}$		<0.4		%/V
Nonlinearity	NL	best fit straight line R_{FS1000}, R_{FS2000}		±0.05		%FS
g-Sensitivity		Sensitivity to acceleration stimuli in all three axis (frequency <20kHz)			0.1	°/s/g
Zero-rate Offset	Off Ω_x Ω_y and Ω_z	Nominal VDD supplies T_A=25°C, slow and fast offset cancellation off		±1		°/s
Zero-rate Offset Change over Temperature	TCO	Nominal VDD supplies $-40°C \le T_A \le +85°C$ R_{FS2000}		±0.015		°/s per K
Zero-rate Offset Supply Volt. Drift	OffΩ_{VDD}	T_A=25°C, $VDD_{min} \le VDD \le VDD_{max}$		<0.1		°/s /V
Output Noise	n_{rms}	rms, BW=47Hz (@ 0.014°/s/√Hz)		0.1		°/s

Table A.3: Full specifications of BMI088 gyroscope [37]

Appendix B

Supplementary theory

B.1 Rotation matrices

Listed below are the 3-D rotation matrices which can be used to rotate a reference frame by (θ) degrees about the x, y or z axis [10]:

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix} \tag{B.1}$$

$$R_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix} \tag{B.2}$$

$$R_z(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{B.3}$$

These rotation matrices can be combined to create a unique rotation.

B.2 Obtaining reverse transformations from IMU data

This section briefly describes the steps taken to obtain a reverse transformation based on the data collected from the IMU in the Livox Avia. This reverse transformation is used as an initial estimate in the point cloud data.

1. Integrate the angular velocity values concerning time.
2. Integrate the linear acceleration values concerning time twice to obtain the linear position.
3. Obtain the time stamps of the two point clouds which are to be aligned and find the corresponding angular and linear position IMU readings.

4. Take the difference between the two selected IMU readings and use that value in equation 3.2

Appendix C

Source code

For all the source code developed throughout this project refer to the following GitHub repository:

GitHub repository link. Click here.

Appendix D

Ethics clearance

PRE-SCREENING QUESTIONNAIRE OUTCOME LETTER

STU-EBE-2023-PSQ000544

2023/08/06

Dear Daniel Jones,

Your Ethics pre-screening questionnaire (PSQ) has been evaluated by your departmental ethics representative. Based on the information supplied in your PSQ, it has been determined that you do not need to make a full ethics application for the research project in question.

You may proceed with your research project titled:

SHARC project: Ice foe size and shape characterisation from LiDAR scans

Please note that should aspect(s) of your current project change, you should submit a new PSQ in order to determine whether the changed aspects increase the ethical risks of your project. It may be the case that project changes could require a full ethics application and review process.

Regards,

Faculty Research Ethics Committee

Bibliography

[1] Janusz Bedkowski. Understanding 3d shapes being in motion. *Journal of Automation, Mobile Robotics and Intelligent Systems*, 1:42–46, 01 2013.

[2] Liam Clark. *The Stewart Platform: A Proposed Ocean Surface Simulation Platform.* University of Cape Town, 2018.

[3] J. Cohen. Point cloud registration: Beyond the iterative closest point algorithm, 2023. `https://www.thinkautonomous.ai/blog/point-cloud-registration/`.

[4] Charlie Cropp MRICS, 2017. `https://info.vercator.com/blog/feature-extraction-from-point-cloud-data`.

[5] Pierre Dellenbach, Jean-Emmanuel Deschaud, Bastien Jacquet, and François Goulette. Ct-icp: Real-time elastic lidar odometry with loop closure. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5580–5586. IEEE, 2022.

[6] Konstantinos G Derpanis. Overview of the ransac algorithm. *Image Rochester NY*, 4(1):2–3, 2010.

[7] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[8] R Forsberg, K Keller, and SM Jacobsen. Laser monitoring of ice elevations and sea-ice thickness in greenland. *International Archives OF Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(3/W4):163–168, 2001.

[9] Thor I. Fossen. *Guidance and control of Ocean Vehicles.* J. Wiley & Sons, 1999.

[10] Thor I. Fossen. *Handbook of Marine Craft Hydrodynamics and motion control.* John Wiley & Sons, 2021.

[11] Marco Gherardi and Marco Cosentino Lagomarsino. Characterizing the size and shape of sea ice floes. *Scientific reports*, 5(1):10226, 2015.

[12] Hristo Hristov. K-d trees in computer science, 2023. Available online: `https://www.baeldung.com/cs/k-d-trees`.

[13] Xiaoshui Huang, Guofeng Mei, Jian Zhang, and Rana Abbas. A comprehensive survey on point cloud registration. *arXiv preprint arXiv:2103.02690*, 2021.

[14] Mahlon C Kennicutt, David Bromwich, Daniela Liggett, Birgit Njåstad, Lloyd Peck, Stephen R Rintoul, Catherine Ritz, Martin J Siegert, Alan Aitken, Cassandra M Brooks, et al. Sustained antarctic research: a 21st century imperative. *One Earth*, 1(1):95–113, 2019.

[15] Jin-Woo Kim, Duk-jin Kim, and Byong Jun Hwang. Characterization of arctic sea ice thickness using high-resolution spaceborne polarimetric sar data. *IEEE Transactions on Geoscience and Remote Sensing*, 50(1):13–22, 2011.

[16] Peter M Kuhn. *Airborne infrared imagery of arctic sea ice thickness*. US Department of Commerce, National Oceanic and Atmospheric Administration ..., 1975.

[17] Rebecca Lindsey and Michon Scott.

[18] Ted Maksym, Sharon E Stammerjohn, Stephen Ackley, and Rob Massom. Antarctic sea ice— a polar opposite? *Oceanography*, 25(3):140–151, 2012.

[19] Miquel Massot-Campos and Gabriel Oliver-Codina. Optical sensors and methods for underwater 3d reconstruction. *Sensors*, 15(12):31525–31557, 2015.

[20] MathWorks. Image registration - matlab documentation, 2023. Available online: `https://www.mathworks.com/discovery/image-registration.html`.

[21] MathWorks. pcdenoise - matlab documentation, 2023. Available online: `https://www.mathworks.com/help/vision/ref/pcdenoise.html`.

[22] MathWorks. pcfitplane - matlab documentation, 2023. Available online: `https://www.mathworks.com/help/vision/ref/pcfitplane.html`.

[23] MathWorks. pcsegdist - matlab documentation, 2023. Available online: `https://www.mathworks.com/help/vision/ref/pcsegdist.html`.

[24] MathWorks. Rigid3d transform - matlab documentation, 2023. Available online: `https://www.mathworks.com/help/images/ref/rigidtform3d.html`.

[25] Paul McManamon. *Lidar Technologies and Systems*. SPIE Press, 2019.

[26] Tsukasa Niioka and C Kohei. Sea ice thickness measurement from an ice breaker using a stereo imaging system consisted of a pairs of high definition video cameras. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science, Kyoto Japan*, 38(8):1053–1056, 2010.

[27] Xiaojuan Ning, Fan Li, Yinghui Wang, and Wen Hao. Primitive shape extraction for objects in point cloud. In *2017 International Conference on Virtual Reality and Visualization (ICVRV)*, pages 144–149, 2017.

[28] Claire L Parkinson. Southern ocean sea ice and its wider linkages: insights revealed from models and observations. *Antarctic Science*, 16(4):387–400, 2004.

[29] AD Rogers, BAV Frinault, DKA Barnes, NL Bindoff, R Downie, HW Ducklow, AS Friedlaender, T Hart, SL Hill, EE Hofmann, et al. Antarctic futures: an assessment of climate-driven changes in ecosystem structure, function, and service provisioning in the southern ocean. *Annual Review of Marine Science*, 12:87–120, 2020.

[30] Sebastian Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. *Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling (3DIM 2001)*, pages 145–152, 2001.

[31] Andrei Sandru, Arto Visala, and Pentti Kujala. Shipborne sea-ice field mapping using a lidar. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4350–4357. IEEE, 2021.

[32] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009.

[33] Hayley H Shen, Stephen F Ackley, and Mark A Hopkins. A conceptual model for pancake-ice formation in a wave field. *Annals of Glaciology*, 33:361–367, 2001.

[34] Ian Simmonds. Comparing and contrasting the behaviour of arctic and antarctic sea ice over the 35 year period 1979-2013. *Annals of Glaciology*, 56(69):18–28, 2015.

[35] D. N. Thomas. *Sea ice*. Wiley-Blackwell, 2017.

[36] *BMI088 6-Axis Motion Tracking for High Performance Applications*. Available online: `https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmi088-ds001.pdf`.

[37] *Livox Avia User Manual v1.0*. Available online: `https://terra-1-g.djicdn.com/65c028cd298f4669a7f0e40e50ba1131/Download/Avia/Livox%20Avia%20User%20Manual%20202204.pdf`.

[38] Xianwei Wang, Hongjie Xie, Yanan Ke, Stephen F Ackley, and Lin Liu. A method to automatically determine sea level for referencing snow freeboards and computing sea ice thicknesses from nasa icebridge airborne lidar. *Remote sensing of environment*, 131:160–172, 2013.

[39] Yusheng Xu, Sebastian Tuttas, Ludwig Hoegner, and Uwe Stilla. Geometric primitive extraction from point clouds of construction sites using vgs. *IEEE Geoscience and Remote Sensing Letters*, 14(3):424–428, 2017.

[40] Wenan Yuan, Daeun Choi, and Dimitrios Bolkas. Gnss-imu-assisted colored icp for uav-lidar point cloud registration of peach trees. *Computers and Electronics in Agriculture*, 197:106966, 2022.