Machine Learning for the Harsh Environment: Applications in Sea Ice Classification and Satellite Magnetic Fault Recovery

by

© Benjamin R. Dowden, B. Eng.

A thesis submitted to the School of Graduate Studies

in partial fulfilment of the requirements for the degree of

Master of Engineering

Department of Mechanical Engineering

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

May 2022

St. John's Newfoundland

Abstract

This thesis presents the development of two machine learning navigation modules for harsh environment applications. The first application investigates semantic segmentation using neural networks for sea ice detection and classification in polar oceans. Two popular generic architectures, SegNet and PSPNet101 are used to segment images. Transfer learning is performed using two custom datasets, one with four classes: ice, ocean, vessel, and sky, i.e., sea ice detection dataset, and the second with eight classes: ocean, vessel, sky, lens artifacts, first-year ice, new ice, grey ice, and multiyear ice, i.e., sea ice classification dataset. The Nathaniel B. Palmer imagery, which captured 2-month footage of the icebreaker completing an Antarctic expedition was used in the creation of both datasets. A subset of the dataset was labeled to generate a 240-image training set for sea ice detection achieving an accuracy of 98% classification for the 26-image test set. The sea ice classification dataset consists of 1,090 labeled images achieving accuracies of 98.3% or greater for all ice types for the 104-image test set.

The second application investigates a new attitude error parameterization and a machine learning regression model for small satellite attitude fault recovery systems experiencing magnetometer bias faults. A simulation environment is developed to mimic an orbit of the international space station, and simulates both the magnetometer and the fine sun sensor on-board a small satellite. A right quaternion error parameterization is presented to ensure consistent error bound growth during the eclipse period of orbits where only a subset of sensor data is available. Using the improved error bounds a fault detection method using Mahalanobis distance is imple-

mented to flag any faults in the system. After the fault detection, the fault recovery uses a regression sliding window optimizer to determine the unknown magnetometer bias that the sensor encounters. The proposed method demonstrates improved root mean squared error and error bound consistency achievable using the right error formulation for magnetic bias fault detection and recovery applications of small satellites.

Acknowledgements

I wish to extend my deepest thanks to Dr. Oscar De Silva and Dr. Weimin Huang for their supervision, guidance, and encouragement throughout my Master of Engineering degree. Without their help, I would not have succeeded. I wish to thank Mr. Desmond Power and Mr. Dan Oldford for their assistance in my work with Killick-1 and sea ice classification, respectively. Thanks to the entirety of the ISLab for aiding me with any technical issues or questions I had. I would also like to thank James Hudson and Brody McKeown for helping with electrical issues, including board schematics and soldering.

In addition, I want to thank my friends for providing stimulating discussions and occasional necessary distractions from my research. Lastly, I would like to give my sincere appreciation to my family for the constant encouragement and support through all my trials and tribulations and for ensuring that I stay on track during difficult periods.

The author is grateful for the financial support provided by the Ocean Frontier Institute, through an award from the Canada First Research Excellence Fund, and in part by the Memorial University of Newfoundland, St. Johns, NL, to Dr. O De Silva, and the Canadian Space Agency CubeSat Grant (17CCPNFL11) to Dr. W. Huang and Mr. D. Power.

Contents

List of Tables

List of Figures

Table of Acronyms

Page number indicates the first significant reference.

ML : Machine Learning (p. 2).

SAR : Synthetic Aperture Radar (p. 5).

CNN : Convolutional Neural Network (p. 7).

MUN : Memorial University of Newfoundland (p. 8).

CCP : Canadian CubeSat Project (p. 8).

CSA : Canadian Space Agency (p. 9).

ADCS : Attitude Determination and Control Subsytem (p. 9).

IoU : Intersection over Union (p. 10).

HSV : Hue Saturation Value (p. 15).

SIFT : Scale-Invariant Feature Transform (p. 18).

SURF : Speeded Up Robust Features (p. 18).

HOG : Histogram of Oriented Gradients (p. 18).

GLCM : Grey-Level Co-occurence Matrix (p. 18).

ANN : Artificial Neural Network (p. 19).

FCN : Fully Connected Neural Network (p. 20).

IMO : International Maritime Organization (p. 26).

UAV : Unmanned Aerial Vehicle (p. 28).

DCM : Direction Cosine Matrix (p. 28).

ECI : Earth-Centered Inertial (p. 30).

ECEF : Earth-Centered Earth-Fixed (p. 30).

QUEST : Quaternion Estimation (p. 32).

LS : Least Squares (p. 32).

EKF : Externed Kalman Filter (p. 33).

UKF : Unscented Kalman Filter (p. 33).

MEKF : Multiplicative Extended Kalman Filter (p. 33).

ℓ_1-HBKF : Laplace Huber Based Kalman Filter (p. 34).

NEES : Normalized Estimation Error Squared (p. 35).

SLAM : Simultaneous Localization and Mapping (P. 35).

FTC : Fault Tolerance Control (p. 35).

FDI : Fault Detection and Isolation (p. 36).

POLARIS : Polar Operational Limit Assessment Risk Indexing System (p. 45).

FP : False Positive (p. 50).

FN : False Negative (p. 50).

FPR : False Positive Rate (p. 51).

FNR : False Negative Rate (p. 51).

TP : True Positive (p. 51).

TN : True Negative (p. 51).

WMM : World Magnetic Model (p. 68).

ISS : international Space Station (p. 69).

NIS : Normalized Innovation Squared (p. 79).

IGRF : International Geomagnetic Reference Field (p. 83).

RMSE : Root Mean Squared Error (p. 84).

SPI : Serial Peripheral Interface (p. 90).

I^2C : Inter-Integrated Circuit (p. 90).

Chapter 1

Introduction

In this chapter, the overall objective and motivation of this thesis are presented. An overview of two navigation challenges in harsh environments along with two different machine learning solution strategies for these applications will be discussed. The problem statement of the thesis is then formulated. Finally, objectives and expected contributions will be highlighted along with the organization of the thesis.

1.1 Research Rationale

Machine learning has emerged as a promising new solution to meet the increasing need for higher levels of environment interaction and autonomy in system design. Machine learning can be broadly identified as development of algorithms which can automatically improve using past experience and data gathered during operation. While machine learning was first proposed in 1949 as a model of brain cell interaction [8], it quickly gained traction in real-world computing applications such as checkers [9]. Currently, machine learning is one of the cutting edge technologies spanning

many application areas such as image detection [10, 11], fraud detection in banking [12], control algorithms [13, 14] and space robotics [15] to name a few.

A harsh environment is defined as an environment with conditions that are very difficult for people, animals, and plants to live in [16]. While there isn't a specific classification of a harsh environment, some of the common examples of these extreme environments include the geographic poles, deep oceans, volcanoes, arid deserts, and of course, space. Each of these environments poses unique challenges for system design.

There are several harsh environment effects that can be mathematically modelled using first principles. As example general rigid body kinematics, orbital mechanics, and a small subset of hydrodynamic forces, ice collision forces are phenomena where reasonable parametric models are available for system design and navigational control purposes. However, there are phenomena with high degree of variability such as sea ice thicknesses and types, feedback sensor lighting conditions, magnetic anomalies, and system degradation which require data driven adaptation of the mathematical description during operation. With the capacity to learn from past observations and patterns, machine learning pairs well with harsh environment applications. Although traditional computing methods exist for harsh environments in place of machine learning techniques, these methods lack robustness and fail to incrementally learn from new observations or patterns. They require detailed knowledge of the data and correlations to generate a proper output from the input. Furthermore, a change in operation conditions requires re-programming as there is no learning or adapting to better fit the environment. Data type is also a limiting factor for traditional methods where the methods often require a rigid well structured form of data which is heavily

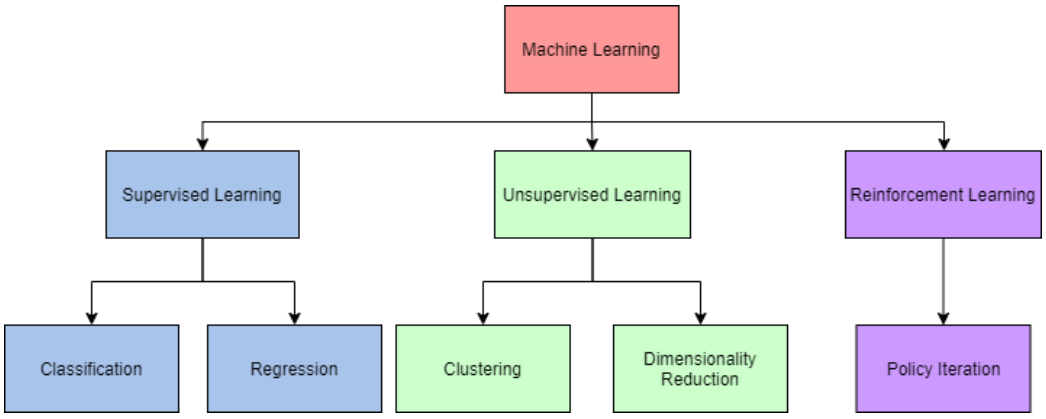


Figure 1.1: The main branches of machine learning algorithms [1].

pre-processed from its original form. Machine learning (ML) overcomes the aforementioned issues through presenting a different data driven paradigm to handle the same tasks [17]. As a result, complex tasks and scenarios can be addressed that would be challenging for humans making them appealing for harsh environment applications [18].

The field of machine learning can be primarily divided into three categories: Supervised learning, Unsupervised learning, and Reinforcement learning [19]. The learning methods are divided based on how the technique utilizes the data. The hierarchy of machine learning techniques is presented in Figure 1.1. In supervised learning, the data is given with the correct corresponding outputs or labeled. Unsupervised learning uses unlabeled data to identify inherent structures, and semi-supervised uses a combination of both labeled and unlabeled data. Lastly, reinforcement learning uses a different method of training where the system interacts with the environment/process for the algorithm to strategically capture new data and learn optimal policies driven by a scalar reward function [20]. Each method has its applications and challenges. Supervised learning is the most broadly used ML technique with applications rang-

ing from classification for camera driven autonomous vehicle control to regression modeling for sensor system calibration [21–23]. Supervised learning requires a sufficiently large labelled data-set for training purposes and as a result could be quite time consuming. Unsupervised learning is typically used for applications where inherent patterns in data needs to be identified without the use of labelled data. However, the method struggles with accuracy of the results compared to supervised methods depending on the applications [24, 25]. Reinforcement learning can be used for a robot (or agent) to learn a new task such as playing table tennis, catching a ball, etc., but requires a safe way for the system to interact with the environment when gathering new data for training purposes [26].

Supervised learning can be further broken into two main subsections: classification and regression. Classification is the discrete form of supervised machine learning, where the algorithm estimates the mapping function of the input variables to discrete or categorical output variables. Classification algorithms are used heavily for computer vision tasks [27] and are extremely useful in harsh environments where object detection and recognition are critical tasks such as in the navigation of ice-infested waters. Regression is the continuous form of supervised learning, it aims to estimate a mapping function to best fit input variables to continuous output variables. Some common regression techniques include linear regression, support vector machines, and regression trees [28]. This work will focus on the applications of supervised learning, since the method can use labelled data as reference and produce accurate results for harsh environment applications [29–31]. Two harsh environment application areas are considered.

The first application of supervised machine learning considered in this thesis is

semantic segmentation, a form of classification that caters well to polar navigation. With the increased traffic and demand in polar waters, safe navigation is critical. While a human eye can easily detect sea ice in the water, the challenge arises in the classification of the ice. Misclassification of ice can result in potentially severe consequences, including extensive denting, and compromising the structural strength [32]. Current methods for sea ice classification and detection include on-board trained ice navigators, the usage of ice maps, aerial flyovers, and radar. The usage of ice navigators is common practice however increases the cost of a given trip and is susceptible to human fatigue. Moreover, since it requires a heavily trained individual, certain vessels may not have the capacity for a dedicated specialist such as arctic explorers. The governing body of the region creates ice maps through the usage of synthetic aperture radar (SAR) imagery [33]. While SAR imagery has high-resolution mapping capability, it relies on public information and is typically created once or twice a day, depending on the time of the season and the location. In periods of high currents or winds, it is a common occurrence for the maps to have errors in the ice as the conditions can drastically change over the course of the day. Additionally, these maps contain limited data regarding specific ice parameters and only provide a high-level classification of the ice [34]. Aerial flyovers are a less common approach for navigating the ice due to the high costs and limitations. Similar to ice pilots, this requires a dedicated specialist to fly a vehicle to track the sea ice ahead of the vessel. Furthermore, not all vessels are capable of housing a helicopter or large capable drone on deck, and the weather in the area limits the capability. Radars have seen success in navigating specific aspects of icy conditions. Since all ice going vessels are equipped with ice radar, there exists technology to expand the radar capability for iceberg

detection [35]. This has proven advantages in rough seas and icy waters but only provides limited information to the captain. The detection of bergy bits and icebergs is critical for collision avoidance, but icy waters pose other threats that the radar fails to detect, such as multi-year ice. These methods have limitations and are typically used in conjunction with each other to inform the user of the ice environment. For this application, machine learning can potentially aid in classifying ice types on polar voyages and provide a labelled near real-time result to captains and crew to ensure safe navigation through icy waters.

The second application of supervised machine learning presented in this thesis investigates machine learning regression for satellite magnetic anomaly detection and recovery. Given the harsh environment of space, including the lack of atmosphere, reduced gravity, temperature fluctuations, and radiation, faults of components are a common occurrence. Any failure in the control system or components can lead to complete loss of the system [36]. One such fault is anomalies within the magnetometers which can occur temporarily or have a permanent impact. These anomalies can occur for various reasons. On-board components can generate temporary disturbances for the delicate sensor or external effects such as solar flares and radiation can result in an anomaly [37]. Depending on the anomaly, different recovery methods are required. In the case of internal temporary interference such as magnetorquer usage, simple anomaly detection of the disturbance and omitting the data for attitude determination is required. In order to accurately detect any anomaly within the sensors, an accurate measure of confidence in the expected sensor values is necessary. The more accurate the confidence bounds of the filter, the better any deviation from the confidence bounds will be flagged indicating an actual anomaly. Existing methods do

not effectively track the confidence bounds in eclipse conditions due to a theoretical limitation of the methods (identified later in this thesis). Recent developments in nonlinear attitude error parameterizations have shown improved consistency results in similar application areas [38] indicating a high potential to remedy the inconsistencies of state-of-the-art satellite attitude determination methods.

In the case of a permanent disturbance, a recalibration of the sensor is required where the current calibration is no longer valid. Regression modelling helps in this case as it allows to fit a new model to the observed long term trend of data. Regression calibration has been applied to address low-cost sensor degradation [23] or diagnostics of faults of the spacecraft [39] which require sufficient orbital data corresponding to a large portion of the satellites' expected poses. However, recaliberation based on limited sensor information feeds during orbit remains a challenge for small satellite operations.

1.2 Problem Statement

The following section describes research gaps and problems that will be addressed throughout this thesis.

1.2.1 Problem 1: Sea Ice Detection and Classification

Recent developments in machine learning algorithms have the potential to automate the process of sea ice detection and classification for navigation purposes. Key work in this field includes [40], where authors present a semantic segmentation CNN to classify river ice from aerial drone footage of rivers. Additionally, [41] uses a CNN

to classify various ice objects. However, instead of classifying ice types, this work focuses on ice objects such as icebergs and large ice floes. Work in [42] proposes the usage of sentinel-1 synthetic aperture radar data to classify sea ice using neural networks, where satellite imagery is used instead of on-board camera imagery for sea ice classification. Furthermore, the work in [43] proposes an improved ice navigation system with the combination of ship-based lidar, satellite imagery, and onboard visual classifications from operators. Work at Memorial University of Newfoundland (MUN) has previously investigated traditional image processing-based methods [44, 45], which have resulted in systems that are sensitive to lighting conditions while not having a structured means of improving the system based on expert feedback. While these solutions aim to improve ice detection, none focus directly on image-based polar sea ice classification, especially classifying the type of ice which drives navigation decisions. This makes the methods unsuitable for implementation as a navigational aid for polar vessels encountering ice. Furthermore, an extensive data set is required to train a machine learning system to accurately detect and classify sea ice conditions. However, there does not exist any public labelled data set for the system training task at hand. Therefore, a custom dataset and a training approach need to be investigated for polar sea ice classification using onboard image feeds.

1.2.2 Problem 2: Fault Detection and Mitigation for Small Satellites

There has been a significant increase in the number of nano-class satellites launched into space each year. The development of the CubeSat standard has drastically

simplified and streamlined the process for nano-class satellites to be launched. In Canada, the Canadian CubeSat Project (CCP) developed by the Canadian Space Agency (CSA) has allowed 15 universities across Canada to build their own respective CubeSat and provide a launch window. MUN is one of these universities to receive this opportunity building the Killick-1 satellite. With any CubeSat, there exists a variety of limitations and tradeoffs between subsystems. One critical subsystem is the Attitude Determination and Control Subsystem (ADCS). Due to the restrictive nature of CubeSat design, the satellite is restricted by the sensors available to determine the attitude often lacking redundant sensory information to directly cross validate readings from sensors. This limitation poses a challenge in fault detection and mitigation, as if a component encounters an error, it can lead to catastrophic failure of the entire satellite.

For fault detection and recovery of small satellites, a suitable error parameterization is required to get a good estimate of the expected sensor readings. Due to the limited sensor availability, standard parameterizations in conventional algorithms experience significant drifts in the estimates when the satellite enters the eclipse period of the orbit. There is a recent development in attitude error parameterization to ensure consistent estimates of confidence bounds during drifting periods, i.e., right Lie group error paremeterization[38]. Thus far, work to date in the area of small satellites has not considered this novel non-linear error paremeterization which is expected to enhance fault detection and recovery pipelines of cubesat ADCS systems.

1.3 Objectives and Contributions

The overall objective of the thesis is to evaluate and propose new machine learning methods for two harsh environment navigation problems. First is the sea ice detection and classification problem encountered by ice going vessels. Second is the magnetic anomaly detection and recovery problem faced by low power cube satellites.

1. **Objective 1** Evaluate state-of-the-art semantic segmentation networks for sea ice detection and sea ice classification in polar oceans using custom labelled sea ice navigation dataset.

   First, a four-class sea ice detection dataset will be developed and the state-of-the-art neural networks will be trained on the dataset. The accuracy and Intersection over Union (IOU) performance will be evaluated on the trained networks. A new expanded eight-class dataset with increased ice classes for ice classification will then be developed and the networks will be retrained. The accuracy and IOU will be evaluated again of the new trained neural networks and a conclusion on the capability of the networks will be drawn.

2. **Objective 2** Evaluate the recently developed right $\mathbb{S}3$ [46] error parameterization for attitude determination of small satellites.

   First, a mathematical model of a three sensor satellite will be presented and the right $\mathbb{S}3$ observability consistency will be evaluated. A simulation environment will be developed to evaluate the error parameterization of the satellite and compared versus the standard error parameterization. Lastly, physical testing with the error parameterization will be conducted to validate the simulation

results.

3. **Objective 3** Design and validate a magnetic fault detection and recovery method based on nonlinear rotation error parameterization and machine learning data regression for small satellites.

   A fault detection algorithm will be presented using the improved confidence bounds from the error parmeterization. The fault detection algorithm will be evaluated in a simulation environment to detect constant magnetic bias. Finally, a machine learning data regression algorithm will be presented and evaluated using a simulation environment for fixed magnetic anomalies.

1.4 Organization of the Thesis

- **Chapter 1** presents an overview of the research area, highlights the research statement, and outlines the objectives and contributions of this work.

- **Chapter 2** presents the literature review in the areas of image processing and semantic segmentation related to sea ice detection and classification and highlights the research gaps. Additionally, it presents a literature review of the state-of-the-art attitude determination and control subsystems of small satellites.

- **Chapter 3** presents the machine learning work carried out to develop and implement the sea ice detection and sea ice classification neural networks and the results.

- **Chapter 4** presents the work carried out to design the fault detection of the magnetometer drift and faults, including the improvement of fault bounds in unobservable states to aid in the fault detection. The results of the proposed solutions are presented and discussed.

- **Chapter 5** presents the conclusion and future directives of this study.

Chapter 2

Related Work

This chapter is broken down to two sections.

The first section presents a brief overview of image processing. A background on classical image detection and classification is discussed. Two different machine learning algorithms for semantic segmentation that are investigated for sea ice detection and classification in this study are introduced.

The second section presents a background of attitude determination and control. A review of fault detection algorithms is outlined for satellites, and an overview of control algorithms is investigated. Lastly, machine learning techniques are investigated for fault detection, fault isolation, and fault recovery. Additionally, a brief summary of the right quaternion parameterization for improved error-bound estimates is discussed.

2.1 Machine Learning Methods for Sea Ice Classification

2.1.1 Image Processing Methods

Image processing is a subset of computer vision that has existed since its proposal in the 1960s with the goal of extracting or improving a digital image in some facet [47]. The basis of image processing is to take the entire digital image or a subset of the image and apply various algorithms to isolate or determine specific features. Some common algorithms used in image processing include methods like Otsu's method, where a histogram is generated from a grayscale image and using the algorithm, an optimal threshold is determined to maximize the separation between the gray levels [48]. Another histogram-based technique is image equalization. Using the histogram of a grayscale image, the image can be lightened or darkened by adjusting the histogram locally or globally. This technique is commonly used in data augmentation or photo editing software, and the complexity of the algorithm developed depends on the overall desired application [49]. Spatial filtering can be applied to the digital image to reduce noise or smooth the image, or conversely for edge detection. In each case, a neighborhood operation is applied to the values of the image pixels of a subsection of the image that has the same dimensions as the filter. One of the most common image pre-processing techniques is to apply a Gaussian blur, filtering out the noise of an image while also reducing image detail.

Edges are a critical feature commonly extracted from images. There are numerous edge detection filters, including Roberts, Prewitt, and Sobel [50]. However, edge

detection algorithms operate on the same common principles, first smooth the image to reduce noise, locate the potential edge points, and finally, localize the exact edge using the potential edge points. Edge detection can be expanded further using more advanced algorithms, including Laplacian of Gaussian, difference of Gaussian, or canny edge detection.

Morphological operations are one of the last common generic image processing tasks. The four common morphological operations are erosion, dilation, opening, and closing, where opening and closing are combinations of erosion and dilation. Opening being erosion followed by dilation, while closing is dilation followed by erosion. Each operation uses a structuring element which is a small set or sub-image, to filter across the image.

2.1.2 Classical Semantic Segmentation and Classification

The sea ice detection problem is a classification problem with four classes at the most rudimentary level (vessel, ocean, sea ice, sky). Some of the classical semantic segmentation methods that will be researched include: thresholding, k-means clustering, watershed, and feature detection.

2.1.2.1 Thresholding

While thresholding serves as a basic image processing technique, it has excellent capability in separating images into two classes. Even though the research focuses on four classes for detection, the main output is sea ice from the remainder of the image, creating a foreground-background problem. From a grayscale image, thresholding can be globally optimized through Otsu's method, as mentioned above. The technique is

a dynamic threshold selection aiming to maximize the weighted sum between foreground and background pixels [48]. To address the lighting variation across images, adaptive thresholding improves on Otsu's method. Adaptive thresholding was developed initially by White and Rohrer [51] with the application for optical character recognition and character image extraction. This technique has seen multiple different algorithms over the years [49, 52, 53] with a common approach to compare each pixel to an average of the pixels in the surrounding neighborhood. This thresholding technique has proven to function well in specific color channels or color spaces such as the Hue, Saturation, Value (HSV) to threshold the background from the foreground [54].

2.1.2.2 K-Means Clustering

K-means clustering, is useful in image classification to sort the image pixels into k distinct groups sorting or grouping the pixels. In the case of only two groups this performs similarly to thresholding algorithms however it is more robust in that the grouping can be increased. The standard k-means algorithm is illustrated in Figure Fig. 2.1:

The algorithm iterates to minimize the mean squared distance from each pixel or data point to its nearest center [55]. This segmentation method is heavily used in medical image processing for detection of organs or other necessary boundaries [56]. The size of h heavily limits this method; as the value increases, coarser segmentation results occur with multiple clusters due to the discrete nature of images [57]. While this method is effective in the medical imagery field, in the sea ice detection and classification field, the method is unsupervised and generally would produce many

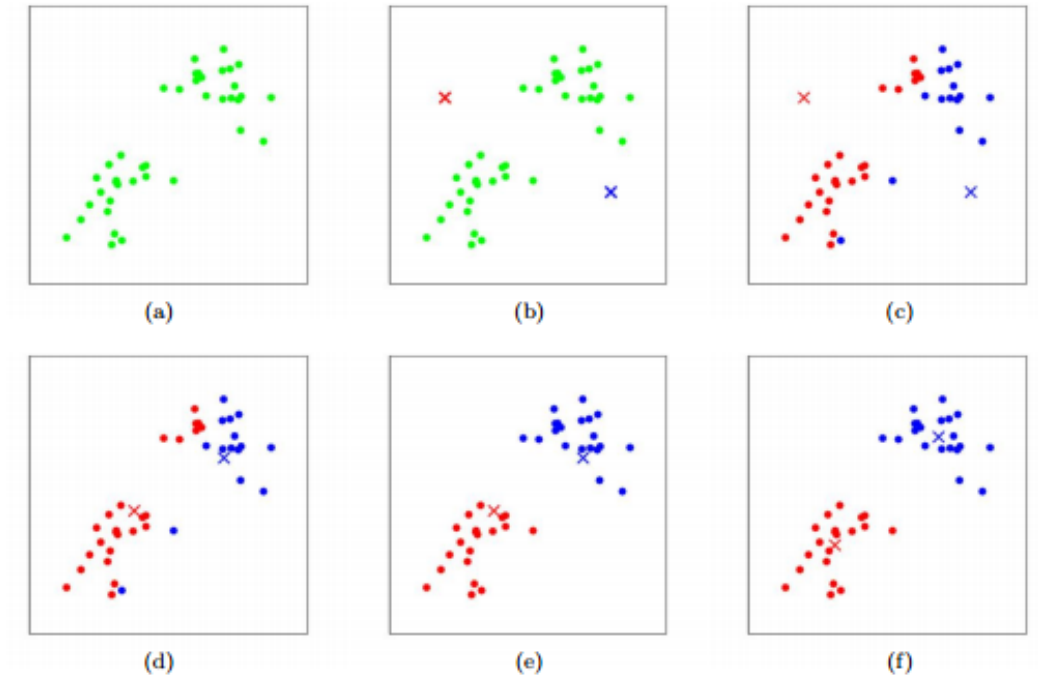


Figure 2.1: Example of how k-means clustering works, the dots represent the training examples, and centroids are represented by the crosses. (a) the original dataset.(b) random initialization of the clusters. (c-f) the k-means clustering algorithm is demonstrated as the points are reassigned and new centroids are defined [2].

irrelevant clusters in complex lighting conditions.

2.1.2.3 Watershed Method

The watershed transformation is a mathematical methodology aimed at segmenting images through edge detection. The original proposal of the technique uses a flooding process on a grayscale image; this results in a geodesic skeleton remaining in the image defining the boundaries. This technique has two methods, an unsupervised and a supervised approach. In the unsupervised watershed, no sections are labeled with markers which can commonly lead to over-segmentation. The supervised methodology uses user-defined markers to indicate the segmentation regions. The supervised approach is an excellent method for dataset labeling as used in multiple labeling software [58, 59] however is not useful for implementation to address the ice detection and classification issue as it requires user input and fails to automate the process. Both methods apply morphological operations to detect local minima and maxima in the images and define boundaries between the sections. This technique is popular in the field of medical imagery processing [60, 61] for both 2D and 3D image processing and in earth surveillance [62]. This method can be useful in boundary detection of ice pans; however, in non-full ice fields, the segmentation of the ice from the waves becomes ambiguous as over-segmentation is a common occurrence with watershed segmentation [63, 64]. Additionally, this technique for ice detection would only serve to over-segment the ocean surface and ice without room for classification and require specific masking to remove the sky and vessel from the imagery.

2.1.2.4 Feature Detection

Feature recognition is a fundamental image processing technique for problems including object recognition, motion tracking, image stitching, and 3D reconstruction. Features range from curvature features like corners and sharp changes, short lines and line ending, to textures. The features selected are designed to be both geometric invariant and photometric invariant [65]. Feature description is a critical step for feature detection, especially for object detection. One of the most common feature descriptors is the scale-invariant feature transform (SIFT) which uses multiple scale spaces to determine extrema or interest points. A model is fit to each interest point to describe both the location, scale, and orientation [66]. Other feature descriptors include speeded up robust features (SURF) [67] and histogram of oriented gradients (HOG) [68] which offer various advantages and disadvantages compared to each other. These techniques are excellent at detecting and locating known objects in various images, localization for robots, and image stitching [69–71]. This technique is not useful in unknown environments or where the object to detect varies heavily, i.e. sea ice. The texture is another feature descriptor that can be used for object recognition and detection. The texture in an image can be evaluated and determine the entropy for regions of the image. The greater the entropy, the more information content [72]. Another approach to texture segmentation is the determination of both grey-level co-occurrence matrix (GLCM) and grey-level difference vector, which measure the distance and angular spatial relationships over subsections of an image [73]. These techniques are widely applicable to known regions and commonly used in the medical and aerial surveillance fields [73–75]. This method can be seen to struggle to provide

full segmentation in some cases, where properly segmented regions occur; however, segmentation of regions within the object is seen, and a large variation of the tuning parameters are required to get optimal segmentation [76].

2.1.3 Machine Learning for Semantic Segmentation

While classical semantic segmentation or image labeling techniques present a high degree of success in a particular idealized set of operating conditions, purely visual classification methods struggle with various challenges. In any real-world environment that an image recognition method is developed for, small changes in the environment can lead to degraded performance or complete failure of the algorithm. This is definitely the case when it comes to the harsh environment of the polar seas. Due to the ever-changing lighting during the cycle of a day and glare off the ice and ocean combined, it creates a challenge to properly segment the image [77]. While there exist solutions to deal with varying lighting environments, they typically require adjustments to the colorspace to increase the separation between perceived foreground and background pixels. They require a parameter to be set, making a robust solution difficult [78]. The variance in the ice type, size, color, texture, and shape presents an additional hurdle [79].

Neural networks have become an established approach for modern semantic segmentation problems. In contrast to classical techniques highlighted above, artificial neural networks (ANN) provide a more robust solution to the segmentation problem with its ability to train for varying conditions while capturing both the global and local context of an image, resulting in significantly improved results for changing envi-

ronmental factions. The incremental training capability of ANNs is heavily utilized to improve machine vision front ends of popular industrial systems such as autonomous driving [80]. ANNs are designed to handle problems similar to how a human brain processes problems. They are comprised of units that represent neurons, with each neuron receiving inputs, processing the input, and generating an output. A layer can be defined as a collection of nodes or neurons mapping the inputs of a network to its output. Typical neural networks are a network with several layers that handle increasingly complex input-output relationships. The combination and arrangement of the units define the architecture of the network, which can vary heavily. Unlike classical methods, ANN's are not programmed for a specific task; they are trained on datasets to learn the desired task [81]. Through the network's training, bias and weights are learned for each unit within the model, making the model cater to the desired problem. Deep learning is an expansion of ANN, which refers to the complex multi-layers of network and how the neurons within the network interact with each other [82].

A wide array of neural network architectures for semantic segmentation exist, each having varying accuracy and speed trade-offs intended for different application domains. One of the widely popular neural networks for image classification was AlexNet. The neural network consists of eight layers with weights, five convolutional layers, and three fully-connected layers. The last layer of the network classified the input image into a distribution of 1000 class labels as per the dataset to an end test accuracy of 84.6% [83]. The basic network architecture consists of convolutional layers connected such that a final pixel-wise prediction to segment the image is produced, e.g., fully connected convolutional neural network (FCN) [84]. Depending on the task

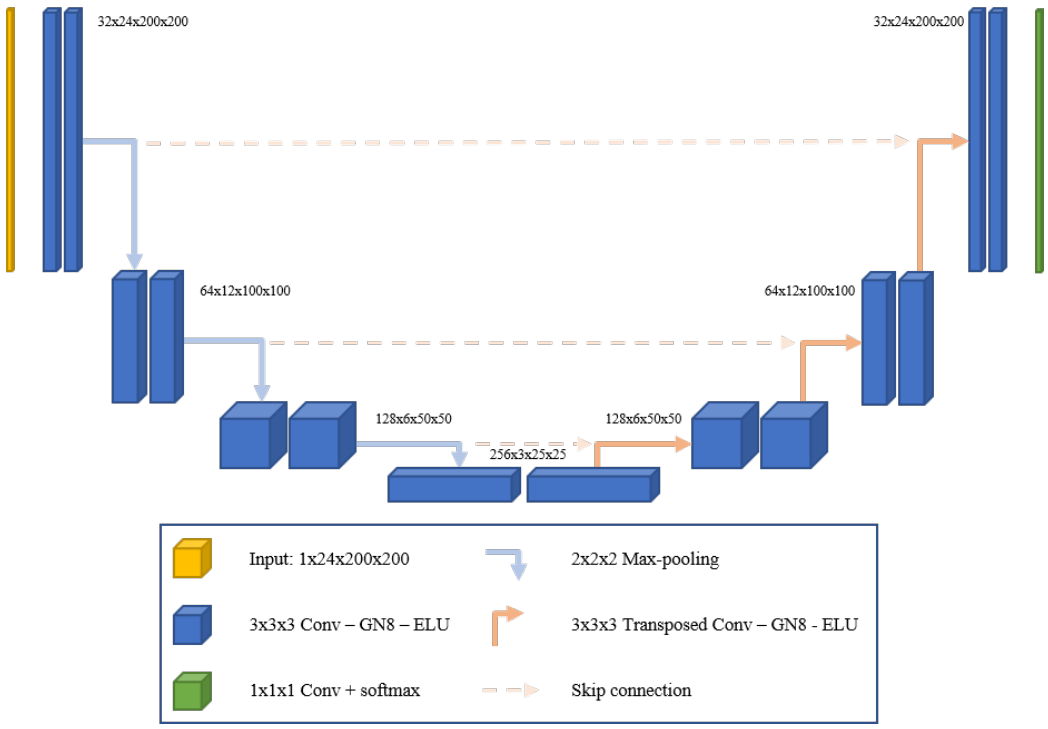


Figure 2.2: The U-net architecture. Each blue box corresponds to a multi-channel feature map. The arrows denote the type of operation. The dashed lines represent the skip connection between each corresponding encoder decoder layer [3].

that the neural networks are built for, these networks become increasingly complex. U-Net [3] presents an expansion of the FCN, adding skip connections and symmetry between the encoder and decoder layers to yield better accuracies. The visualization of the U-Net architecture is presented in Figure 2.2, where the encoder, decoder, and skip connections are seen. Additional layers in architectures and different activation functions further increase the effectiveness of the network for segmentation problems.

For this work two networks were selected to analyse their capabilities for sea ice detection and classification based on their popularity in semantic segmentation applications [85]. The first network chosen was a lightweight network known as SegNet

which offers a quick training time and low processing time. This network was chosen to examine if a lightweight solution is capable to achieve a high success rate in the environment without adding unnecessary complexity to the model. The second model further examined is the PSPNet101 network. Contrary to the SegNet model, this is a heavy neural network known for its capability for semantic segmentation in urban applications. This model was chosen to provide a highly accurate model for the task with the trade-offs of both increased training time and increased processing time of the input imagery.

2.1.3.1 SegNet

Segmentation Network (SegNet) [4] is a deep encoder-decoder architecture developed by the University of Cambridge. This model features a quick and relatively simple neural network to perform semantic segmentation. It is offering a significantly less intensive training time and processing time compared to more complex network architectures. The neural network functions through using a sequence of non-linear processing layers known as encoders and a corresponding set of decoders with a pixel-wise classifier to label the images. Additionally, each encoder consists of one or more convolutional layers, followed by non-overlapping max-pooling and sub-sampling layers. Through the max-pooling indices in the decoders, it allows the network to retain high-frequency information such as shapes and contextual relationships while reducing the overall number of trainable parameters in the decoders [4]. The structure of the neural network is seen in Figure 2.3.

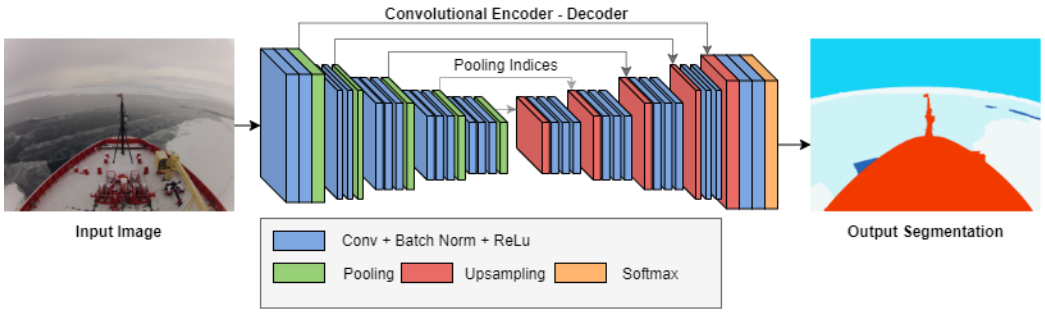


Figure 2.3: The structure of the SegNet encoder-decoder architecture [4].

2.1.3.2 PSPNet101

Pyramid Scene Parsing Network (PSPNet) was introduced for semantic segmentation to provide the capability of global context information to increase the accuracy of the classifications. This model offers a high level of accuracy compared to all other models seen in the field; however, it requires an intensive and lengthy training process due to a large number of parameters. PSPNet101 functions through the combination of the ResNet101 convolutional neural network and a four-layer pyramid pooling module to create the output results [5]. The four-layer pyramid pooling allows the network to develop different region-based context aggregation with division into different sub-regions in the images. The structure of the neural network is seen in Figure 2.4. The ResNet101 CNN is implemented at the start of the pyramid scene parsing to extract features. ResNet101 contains 101 layers that use skip connections in the neural network that serves to prevent significant visual feature loss across the layers. This skip connection implemented is responsible for the models' ability to learn the classification identity without losing context or residual learning in which it obtained its name [86].

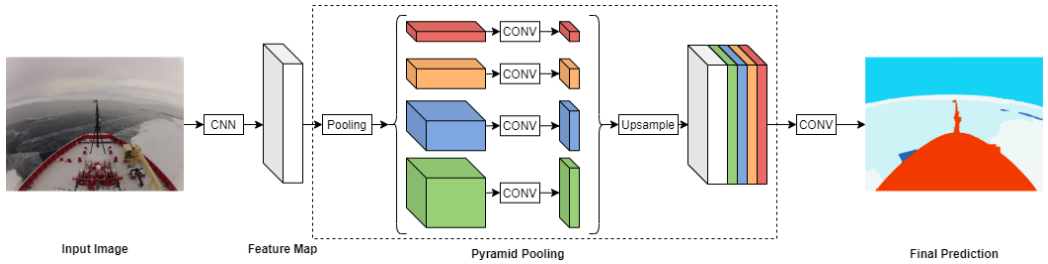


Figure 2.4: The architecture of the PSPNet model, in the case of the PSPNet101 model implemented, the CNN input seen in the figure is the ResNet101 model [5].

2.1.3.3 Transfer Learning

For established deep learning applications there are readily available datasets such as for autonomous driving [87], pedestrian tracking [88], and object recognition [89]. However, to the best of the authors' knowledge, there are no labeled datasets available in the literature to classify ice types in polar environments. In such scenarios, custom labeled datasets are created to meet the study's needs, which is adopted in this work. Automotive applications reported in [90] create a custom dataset for vehicle classification based on camera position, and [91] develop a large dataset of 14,144 images for vehicle classification. Marine applications reported in [92] design its own custom dataset for coral reef segmentation, and work in [93] design a custom dataset for oceanic Eddies. The coral reef dataset has 413 images, where the authors perform a nine-class classification of the data. Typically dataset sizes of the order of 100's are utilized in these studies, while larger, more representative datasets would offer an enhanced network performance. To remedy the smaller datasets that occur as a result of a custom dataset, there are two common approaches, data augmentation and transfer learning. Data augmentation artificially expands the dataset through various image

transformations or adjustments to create new images for training. Commonly used effective data augmentation methods are simple geometric transformations, horizontal flipping, color space modifications, and random cropping [94]. Transfer learning [95] is a technique used to improve the learning of a new neural network through the transfer of pre-trained model weights. In transferring the weights, the new model can start the training process using the previous knowledge and adapt it to the new dataset. This leads to reduced training times and potential higher accuracies in the model quicker.

2.1.4 Related Ice Classification Work

The work presented in this thesis is the first work in its field using in-situ onboard camera imagery to classify sea ice using neural networks to the best of the authors' knowledge. Closely related applications to this work include [40], where authors present a semantic segmentation CNN to classify river ice from aerial drone footage of rivers. Additionally, [41] uses a CNN to classify various ice objects. However, instead of classifying ice types, this work focuses on ice objects such as icebergs and large ice floes. Work in [42] proposes the usage of sentinel-1 synthetic aperture radar data to classify sea ice using neural networks, where satellite imagery is used instead of onboard camera imagery for sea ice classification. Lastly, [43] proposes an improved ice navigation system with the combination of ship-based lidar, satellite imagery, and onboard visual classifications from operators. Sea ice detection and classification in operational settings are carried out by an ice services specialist considering several information sources, including ice radar, any available satellite, aerial imagery, and

in-situ visual identification. The concentration of different ice types in a given zone directly correlates to the operational risk of the vessel governed by International Maritime Organization (IMO) standard specifications. Furthermore, the details of the ice concentration of different zones serve as additional inputs to the daily ice charting activities of the international ice patrol, and the Canadian ice service [34]. Previous research at Memorial University of Newfoundland has previously investigated traditional image processing-based methods for sea ice classification [44, 45], which have resulted in systems that are sensitive to lighting conditions, while not having a structured means of improving the system based on expert feedback. This thesis targets automating the in-situ visual detection and classification of ice to support operational risk assessment and ice charting activities onboard vessels while having a means of incrementally retraining the system as new data becomes available.

2.2 Machine Learning for Attitude Determination and Fault Detection

2.2.1 Attitude Determination

Attitude is the three-dimensional orientation of a body or vehicle with respect to a specific reference frame. Attitude determination is the use of various sensors and mathematical models to gather necessary information related to attitude components. These multiple components are used to determine the attitude, typically parameterized by quaternions, Euler angles, or rotation matrices [96]. In any navigation system, the attitude is critical, especially for aircraft, spacecraft, automobiles, and robotic

platforms. A wide variety of sensors are employed to measure specific vectors to determine the attitude in a system. Attitude determination systems onboard satellites use a combination of both relative and absolute attitude sensors to determine the attitude. The primary sensor used in satellite attitude control is the gyroscope, providing only the angular inertial measurements, an additional attitude measurement is required to fully understand the satellite's full state (e.g., position, velocity, and attitude). These aiding sensors include coarse and fine sun sensors, horizon sensors, star sensors, and magnetometers [97]. Gyroscopes can provide attitude by integrating angular velocity, but this is susceptible to drift, so measurements of know reference directions or absolute attitude sensors (magnetic north, gravity, sun vector, star vector) are needed to periodically correct the drift [98]. There exist multiple methods to address this drift, including Kalman filtering, additional redundant sensor readings [99], or incorporating camera data [98] to list a few.

2.2.1.1 Attitude Representation

The attitude representation method is critical for performance of the attitude estimator. Euler angles are one of the most common attitude representations obtaining their name from Leonhard Euler [100]. Euler angles for attitude representation consist of three angles used to describe the rotation about the axes. There exist twelve possible rotation sequences about the axes that are commonly divided into two groups:

- Proper Euler angles $(x-y-x, x-z-x, y-x-y, y-z-y, z-x-z, z-y-z)$
- Tait-Bryan angles $(x-y-z, y-z-x, z-x-y, x-z-y, z-y-x, y-x-z)$

The proper Euler angles are known as the classical Euler angles initially proposed.

In contrast, the Tait-Bryan angles consist of a rotation about each axis as opposed to the symmetric set of proper Euler angles [101]. Both representations are still referred to as Euler angles. In any Euler angle attitude representation, the rotation sequence would be specified. One of the most common representations is the roll (γ), pitch (β), yaw (α) used for aircraft, unmanned aerial vehicles (UAVs), robotics, and spacecraft. The main issue with Euler angle representation is the gimbal lock. Gimbal lock occurs when a rotation about an axis results in the rotation matrix aligning, making it indistinguishable to identify individual rotations within the matrix due to the axis alignment [102]. While this is not an issue in sea vessels or cruising aircraft; in satellites or spacecraft, gimbal lock is quite possible; thus, a different attitude representation is used.

The rotation matrix or direction cosine matrix (DCM) is another attitude representation to define the orientation of a body frame relative to a reference frame. As the name suggests, the representation is a 3x3 matrix with the columns representing unit vectors in the body with respect to the reference axes [103]. While DCM overcomes the gimbal lock issue seen in Euler angles, it is not overly used in attitude control due to its number of parameters, requiring nine values to be stored and updated [104, 105].

The quaternion is another representation of attitude, proposed by William Hamilton in 1843. Instead of representing an attitude as a combination of rotations about three axes, the quaternion represents a single rotation angle about a unit vector [97]. A quaternion has four parameters that a four-component complex number can define:

$$\mathbf{q} = q_1 + q_2\mathbf{i} + q_3\mathbf{j} + q_4\mathbf{k} \tag{2.1}$$

where 1, **i**, **j**, **k** are the basis of the quaternion. i,j, & k represent imaginary numbers with the following properties:

$$\begin{aligned} \mathbf{i} \circ \mathbf{i} = -1, \quad \mathbf{i} \circ \mathbf{j} = \mathbf{k}, \quad \mathbf{i} \circ \mathbf{k} = -\mathbf{j}, \\ \mathbf{j} \circ \mathbf{j} = -1, \quad \mathbf{j} \circ \mathbf{k} = \mathbf{i}, \quad \mathbf{j} \circ \mathbf{i} = -\mathbf{k}, \\ \mathbf{k} \circ \mathbf{k} = -1, \quad \mathbf{k} \circ \mathbf{i} = \mathbf{j}, \quad \mathbf{k} \circ \mathbf{j} = -\mathbf{i} \end{aligned}$$

A unit quaternion is a special quaternion with the norm of 1, all attitude representations use a unit quaternion. that is:

$$|\mathbf{q}| = \sqrt{q_1^2 + q_2^2 + q_3^2 + q_4^2} \tag{2.2}$$

To normalize any quaternion to make it a unit quaternion, it is divided by its norm:

$$||\mathbf{q}|| = \frac{\mathbf{q}}{|\mathbf{q}|} \tag{2.3}$$

A unit quaternion can be written in terms of the its angle θ and the unit vector $\boldsymbol{u} = \mathbf{q}/||\mathbf{q}||$:

$$\mathbf{q} = cos(\theta) + \boldsymbol{u}\, sin(\theta) \tag{2.4}$$

Quaternions also have unique multiplicative properties, the product of two quaternions is defined by:

$$\mathbf{q}_1 \bigotimes \mathbf{q}_2 = \mathbf{q} = \begin{bmatrix} s \\ \vec{\nu} \end{bmatrix} = \begin{bmatrix} s_1 \cdot s_2 - \vec{\nu_1} \circ \vec{\nu_2} \\ s_1 \cdot \vec{\nu_2} + s_2 \cdot \vec{\nu_1} + \vec{\nu_1} \times \vec{\nu_2} \end{bmatrix} \tag{2.5}$$

where s_1 is the scalar portion of the first quaternion (q_1) and ν_1 is the vector portion of the first quaternion (q_2, q_3, q_4), s_2 is the scalar portion of the second quaternion (q_2) and ν_2 is the vector portion of the second quaternion quaternion [106]. Quaternions

have a wide heritage in space controls due to eliminating the gimbal lock and low parameters compared to the alternative attitude methods. The quaternion product is used to combine two rotations together to find the resultant rotation. The DCM equivalent of this is matrix multiplication between two rotation matrices. Quaternions are not a faultless representation, and they struggle from the issue of double wrapping the attitude. The issue arises when the quaternion is represented as the opposite vector and rotation, resulting in the same rotation in space but differing quaternion value [107]. To overcome this issue, the sign of $\mathbf{q}$ is monitored and adjusted to ensure that $q_0 \geq 0$ resulting in rotation angles $\theta \in [0, \pi]$ about the axis of rotation [108].

2.2.1.2 Reference Frames

In any attitude determination, there exist variables measured in various reference frames. In this thesis, any reference frame will be mentioned in curly brackets {}. Any vectors will be in bold to indicate a vector, and any matrices will be represented with a capital letter.

The first necessary reference frame is the Earth-centered inertial {ECI} frame. This frame is the inertial frame assumed for satellite navigation where Newton's laws of motion are applied. Hence it is assumed to be a non accelerating frame. For {ECI} the origin is defined as the center of Earth, the x-axis points toward the vernal equinox, z-axis points towards the Earth spin axis and y completes the right-hand coordinate system. The second reference frame for attitude determination is the Earth Centered Earth Fixed {ECEF} frame. This frame has its origin fixed to the Earth's center and rotates with the same rotation of Earth about the z-axis. The {ECI} frame initial time is defined based on the standard used. The most common {ECI}

definition is J2000 with the Earth's Mean Equator and Equinox a 12:00 Terrestrial Time on January 1st, 2000. With this definition, the x-axis is aligned with the mean equinox, the z-axis with Earth spin axis, and y-axis is 90°East of the celestial equator [109]. The body frame {B} is a frame rigidly defined on the vehicle, usually at a convenient fixed location like the center of mass to simplify the derivations of other matrices [97]. The axes have no rigid definition and are defined freely; however, in small satellites, the axes are typically defined to be perpendicular to the satellite faces. The last important frame is the sensor frame {S}. This frame is defined as per the sensor manufacturer. Transformation matrices are commonly used to describe the sensor frame in the body frame before any additional transformations. Other reference frames include platform frames where sensors are located on either a non-rigid attachment or not aligned with the body frame.

2.2.2 Attitude Filtering Algorithms

Attitude filtering algorithms serve to calculate the attitude of vehicles, more specifically in the case of this work to determine the attitude of a satellite. Various measurement data is collected from sensors to determine the attitude, namely, gyroscopes, magnetometers, sun sensors, star trackers, and earth sensors. The sensors used vary depending on the mission and requirements for accuracy. There exist multiple algorithms for attitude determination. However, they can be split into two categories: attitude determination algorithms, and attitude estimation algorithms [110].

2.2.2.1 Attitude Determination

Attitude determination algorithms are widely used however, they are limited to estimating the attitude and no other states, including noise, biases, or angular rates. Standard determination algorithms include TRIAD, Quaternion Estimation (QUEST), and Least Squares. TRIAD is a straightforward algorithm that uses two measurement vectors to obtain the attitude of the satellite [111]. QUEST is an improvement on the TRIAD algorithm through the usage of multiple references and observed measurements. The algorithm estimates the optimal eigenvalue and eigenvector to solve the Wahba problem, thus providing the attitude [111, 112]. The Wahba problem consists of finding the matrix A that minimizes the following:

$$L(A) = \frac{1}{2}\sum_{i=1}^{n} \boldsymbol{a_i}|\boldsymbol{b_i} - A\boldsymbol{r_i}|^2 \tag{2.6}$$

where $\boldsymbol{a_i}$ are non-negative weights, $\boldsymbol{b_i}$ is a set of unit vectors measured from the satellite body frame, and $\boldsymbol{r_i}$ are the corresponding unit vectors in a reference frame.

Lastly, Least Squares (LS) is a regression model that aims at minimizing the cost function of the measurement residual between the sensor readings, that is:

$$\boldsymbol{\epsilon} = \boldsymbol{y} - H\hat{\boldsymbol{x}} \tag{2.7}$$

$$J = \boldsymbol{\epsilon}^T\boldsymbol{\epsilon} \tag{2.8}$$

where $\boldsymbol{\epsilon}$ is the measurement residual, $\boldsymbol{y}$ in the measurement, H is the measurement matrix, $\hat{\boldsymbol{x}}$ is the state estimate, and J is the cost function. The LS method has been used to improve QUEST while still maintaining the lightweight computations desired for attitude determination algorithms [113].

2.2.2.2 Attitude Estimation

Attitude estimation algorithms expand upon attitude determination algorithms by estimating the full state vector of the satellite. The algorithms use both sensor measurements, and dynamic or kinematic models to propagate the estimated state. The most common attitude estimation algorithm is the Kalman Filter [114]. The primary function of a Kalman filter is to perform linear Gaussian state estimation using noisy input and measurement signals. There exists a wide variety of Kalman filter types. Extended Kalman Filter (EKF) is an extension of the conventional Kalman filter to work for both discrete-time and continuous non-linear systems [115]. The Unscented Kalman Filter (UKF) is an improvement to the EKF. The UKF approximates the probability density from the non-linear transformation of a random variable instead of the approximation with a Taylor series expansion [116]. Multiplicative Extended Kalman Filter [117] (MEKF) introduces multiplicative error state definitions when computing rotational errors, outperforming EKF when having rotational dynamics. More recent developments include the Laplace ℓ_1 Huber Based Kalman Filter (ℓ_1-HBKF), which employs Laplace distribution and Huber based method to update the measurement covariance. This improves filtering performance with different model errors and heavy-tail noise [118]. Fuzzy logic has seen implementation in EKFs to improve the noise covariance matrices, using IF-THEN rules to update the noise based on the detected distortions, external accelerations, and vibrations [119]. Adaptive Kalman filtering serves to improve the filtering results through adaptively tuning the covariance matrix based on the disturbances and noise encountered [120]. Alternatively, cost function minimization (or data fitting) using optimization libraries for the

purpose of estimation can be used for attitude estimation [121, 122]. The main drawback of cost function minimization is that they require high computational demand for on-board implementation.

2.2.3 Attitude error parameterization

Both quaternions and rotation matrices are not vector spaces, i.e., addition and subtraction operations of euclidean geometry is not valid. Hence these are identified within a different mathematical construct termed Lie groups. The group $\mathbb{SO}3$ represents the attitude of the rigid body using rotation matrix parameterization. The group represents unit quaternions [123]. To better map the non-linear errors of attitude, these Lie groups are used to improve the error mapping in filtering algorithms. The error of any parameterization must use a unique action defined for each lie group to determine the error between two attitudes which is exploited in this thesis for accurate attitude estimation.

Generic filters and optimization libraries use linear error parameterization, i.e. the attitude error $\tilde{\boldsymbol{q}}$ is defined using vector subtraction $\boldsymbol{q} - \hat{\boldsymbol{q}}$. However this is inaccurate when it comes to unit quaternions as the resulting quantity is no longer a unit quaternion. In mathematical sense we say that quaternions belong to the group $\mathbb{S}^3$ and the corresponding error parameterization for group $\mathbb{S}^3$ is defined using quaternion multiplication. This error can be defined in two ways. The left error parameterization assumes that $\tilde{\boldsymbol{q}} = \hat{\boldsymbol{q}}^{-1} * \boldsymbol{q}$. Similarly the right error parameterization assumes that $\tilde{\boldsymbol{q}} = \boldsymbol{q} * \hat{\boldsymbol{q}}^{-1}$. Typically these attitude errors are represented as rotation vectors $\tilde{\boldsymbol{\theta}}$ and the mapping from the rotation vector space to the quaternion group space defined

using a mathematical operation called the exponential map for $\mathbb{S}^3$. This mapping results in the following identities for left and right error parameterization: For left error: $\boldsymbol{q} = \hat{\boldsymbol{q}} * \text{Exp}_{S3}(\tilde{\boldsymbol{\theta}})$ $\tilde{\boldsymbol{\theta}} = \text{Log}_{S3}(\hat{\boldsymbol{q}}^{-1} * \boldsymbol{q})$ For right error: $\boldsymbol{q} = \text{Exp}_{S3}(\tilde{\boldsymbol{\theta}}) * \hat{\boldsymbol{q}}$ $\tilde{\boldsymbol{\theta}} = \text{Log}_{S3}(\boldsymbol{q} * \hat{\boldsymbol{q}}^{-1})$. Right error is shown to maintain the accurate confidence levels of its estimated quantities than the generic versions used by satellites. This is also termed as consistent estimation of attitude. With the right $\mathbb{S}^3$ error parameterization, better Normalized Error Estimation Squared (NEES) is observed. Better NEES performance signifies an accurate estimation of the error bounds allowing for improved anomaly detection. Work in [38] shows consistent state estimation for visual inertial navigation systems where the right error parameterization significantly outperforms the standard parameterization and maintains a consistent NEES. Work in [124] reports improved performance using the right error parameterization for Simultaneous Localization and Mapping (SLAM) of robots. Work in [125] shows similar improved estimation performance for multi robot localization using the right error parameterization. Furthermore, [46] proves theoretical convergence guarantees available for right error parameterized filters. Given the accurate tracking of the confidence limits, estimators designed using the right error parameterization are expected to have better performance for attitude estimation, fault detection, and calibration.

2.2.4 Fault Detection

Fault tolerant control (FTC) systems are necessary for any harsh environment system with limited or no physical access to the system after deployment. In some systems, fault prediction ML algorithms have been developed to address faults prior to failure.

These systems use detailed models and historical data to predict faults and determine preventative actions to mitigate the failure. Work in [126] implements SVM and Multilayer perceptron models to determine maintenance times of a centrifugal pump. Long short-term memory algorithms have also shown to be successful for predictive faults in complex manufacturing processes [127]. While fault prediction is capable of completely negating faults, it requires a large amount of data and an understanding of the faults that may occur in the system. Due to the unpredictable nature of space and limited access to data and the satellite itself, fault prediction is not capable of meeting the needs of FTC for satellites. In the case of satellites, fault-tolerant control falls into two categories: passive FTC and active FTC. In passive FTC, a fixed controller is used that has limited fault tolerance capability that relies on multiple sensors and actuators to mitigate predictive failure. Active FTC uses highly re-configurable controllers such that the FTC can be accounted for with built-in algorithms or near real-time with communication with ground stations [128]. There exist multiple algorithms for fault detection and isolation (FDI) used in active FTC. For actuator failures, non-linear observers and sliding mode control have had excellent success in mitigation of the faults [129, 130]. Various machine learning techniques have also shown potential for actuator fault detection. Work in [131] implemented a incremental locally linear embedding method to evaluate if the residual error exceeds a fault threshold. Fault trees and decision trees have been used in conjunction with neural networks to vote on whether a fault has occurred during the orbit [39]. For sensor failure in the attitude determination algorithms, extensions of the sensor filtering algorithms are added to detect and compensate for faults. Work in [132] detects faults through the innovation sequences within the EKF filter, then further identifies the fault through multiple

hypotheses testing. Work in [133] applies an adaptive threshold method using UKFs in conjunction with residual and innovation sequences to flag faults. Lastly, neural network observer strategies can be applied for FDI, using nonlinear-in-parameters neural networks to identify the faults from unknown states [134]. With all these fault detection algorithms, an accurate confidence of the expected sensor readings is required. High confidence can be achieved by redundant sensor sources where the reference signals always have a comparison or better algorithms which keep track of the error bounds.

2.2.4.1 Fault Isolation and Recovery

Once a fault have been detected, it is important in any FTC to isolate the fault and further attempt recovery of the fault if possible. Typical approaches for FDI include the use of redundant sensors and actuators to address the fault. Work in [133] presents a fault isolation and recovery method for satellite reaction wheels using a redundant reaction wheel. In some cases redundant sensors are not available to address the fault, [135] presents a the use of a redundant-free UAV sensor FDI and recovery using a state-estimator-based FDI and recovers faults using a geometrical method. [136] proposes a squared prediction error to detect faulty signals then uses data recovery to reconstruct fault free signals for the faulty sensors. Depending on the type of fault that has occurred, the recovery method varies. In the case of sensor failure a recalibration can be conducted to address the fault. Work in [137] presents the application of regression to recalibrate low-cost sensors for better performance of the lifetime operation. Work in [138] demonstrates the application of regression error compensation to address sensor faults within prosthetics. Sliding window filters prove

to be an excellent means of recalibration for magnetometers and have been cited to estimate unknown parameters within gun launched projectiles [139].

Chapter 3

Sea Ice Detection and Classification

This chapter presents a sea ice detection and classification method for icebreakers in polar seas. Sea ice monitoring plays a critical role in any icebreaker's journey. While the onboard process typically relies on dedicated ice specialists to identify and report sea ice conditions, recent advances in machine vision semantic segmentation methods can offer enhanced automated workflows for sea ice monitoring activities. The standard methodology requires an increased manpower and is prone to human error due to fatigue or overburden of work. With the usage of onboard cameras in conjunction with machine learning algorithms, the sea ice detection and classification problem can be addressed with semantic segmentation solutions to provide near real-time classification of ice.

The state of the art semantic segmentation algorithms require large datasets to provide an accurate classification work. To the best of the author's knowledge, there

does not exist any datasets of labelled sea ice for classification applications publicly available. Additionally, the research field of sea ice classification has primarily been addressed via SAR imagery which is only available periodically and lacks both resolution and real-time classification for the captains of the vessels. Therefore, two custom datasets are required to be developed to provide onboard in-situ sea ice classification and train novel semantic segmentation algorithm applications in the field.

The contributions of this work is as follows:

- We produce two novel datasets of labeled images from aboard an icebreaker in polar seas for deep learning sea ice detection and classification studies. First is the sea ice detection dataset, which includes 240 labeled images and four classes. Second is the sea ice classification dataset, which includes 1,090 labeled images and four additional classes capturing the type of ice seen in the images.

- Establish the detailed benchmark performance for the datasets using the SegNet and PSPNet semantic segmentation architectures, with transfer learning using the Citscapes dataset [140]. The study was capable of achieving an accuracy of 97.8% for sea ice detection and IoUs of 76%, 82%, and 93% for classification of new, grey, and first-year ice types respectively.

This chapter is outlined as follows: Section 3.1 introduces the two custom data sets created and presents the training process for all four models used. In Section 3.2, the sea ice detection results are presented and analyzed. In addition, the results of the sea ice classification trained networks are examined. A general chapter summary is provided in Section 3.3.

3.1 Data Sets

This study first investigates the feasibility of deep semantic segmentation for sea ice detection using onboard in-situ images. This is performed using the "*sea ice detection dataset*" presented in Section 3.1.1. Upon successful feasibility assessment, we perform a detailed evaluation of deep neural networks for different sea ice type classification using images. This is performed using the "*sea ice classification dataset*" presented in Section 3.1.2. Both datasets were created using footage from an actual polar icebreaker journey, i.e., Nathaniel B. Palmer. The properties of the imagery used to develop both datasets are outlined in Table 3.1. The exact route of the Nathaniel B. Palmer on imagery journey is presented in Figure 3.1, the footage took place from the period of February 17$^{\text{th}}$, 2013 to March 20$^{\text{th}}$, 2013 in the Ross Sea during a research cruise investigating the role of dissolved organic carbon in the Ross Sea ecosystem.

Number of images	42,947
Image type	jpg
Image size	1920x 1440
Days	26
Frequency of capture	30 photos per second
Camera info	GoPro HD2
Camera height	17m from the water line

Table 3.1: The properties of the Nathaniel B. Palmer images.

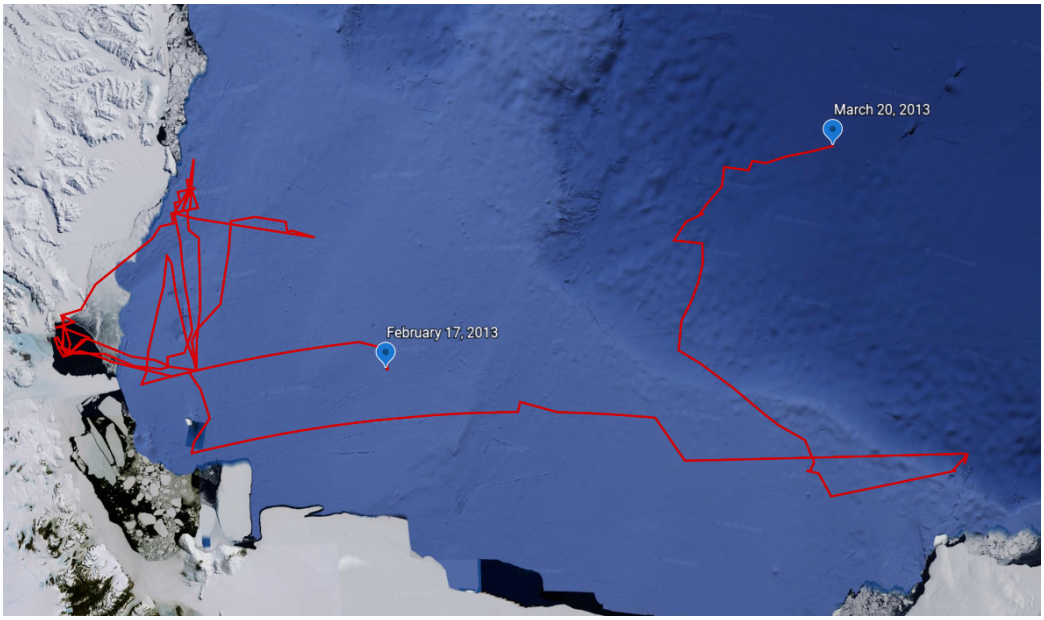


Figure 3.1: The route of the Nathaniel B. Palmer from February 17th, 2013 to March 20th in the Ross Sea.

3.1.1 Sea Ice Detection

The first dataset is a multi-class dataset constructed from Go-pro images taken from the Nathaniel B. Palmer ice breaker on its two-month expedition through the Ross Sea, Antarctica. Video footage of the dataset can be seen in https://youtu.be/BNZu1uxNvlo. This dataset is comprised of 720p and 4K high definition images taken from a fixed location on the icebreaker. The images were periodically captured throughout each of the days, with upwards of 1600 photos captured each day. Each of the days presents an array of different conditions encountered in the voyage ranging from midday sun to gray skies and the setting sun. In addition, some days present precipitation on the lenses, which obscure the overall image. The purpose of this dataset is to evaluate if semantic segmentation is feasible for ice detection aboard icebreakers. This set is

designed to be simplistic in nature, with only four classes being: Ice, Vessel, Ocean, and Sky. This dataset is drawn from four unique days of the journey containing different ice coverage and types, lighting, and environmental conditions commonly encountered in polar voyages. Approximately 60 photos were chosen from each day and labeled into the four classes using the freely available watershed labeling tool PixelAnnotationTool [58]. The dataset was then broken up into three sets, being training, validation, and testing. The data was split into each of the categories with 80% for training, 10% for validation, and 10% for testing. Due to the limited size of the training set, data augmentation was applied to the images via horizontal mirroring to give a final training dataset of 382 images.

3.1.1.1 Sea Ice Detection Labels

The semantic Segmentation ice detection model has four classes, defined as follows:

- Ocean: all open water in the image.
- Ice: all ice visible in the image, including ice pans, thin first-year ice, and icebergs.
- Vessel: all sections of the boat, including the mast and flag visible on the front of the boat.
- Sky: all visible sky in the image.

The model is designed to find and label all pixels to each of the classes, focusing on separating the ocean from the ice. The proper classification of the ice is crucial as mislabeled ice would defeat the entire labeling purpose and provide little relevant

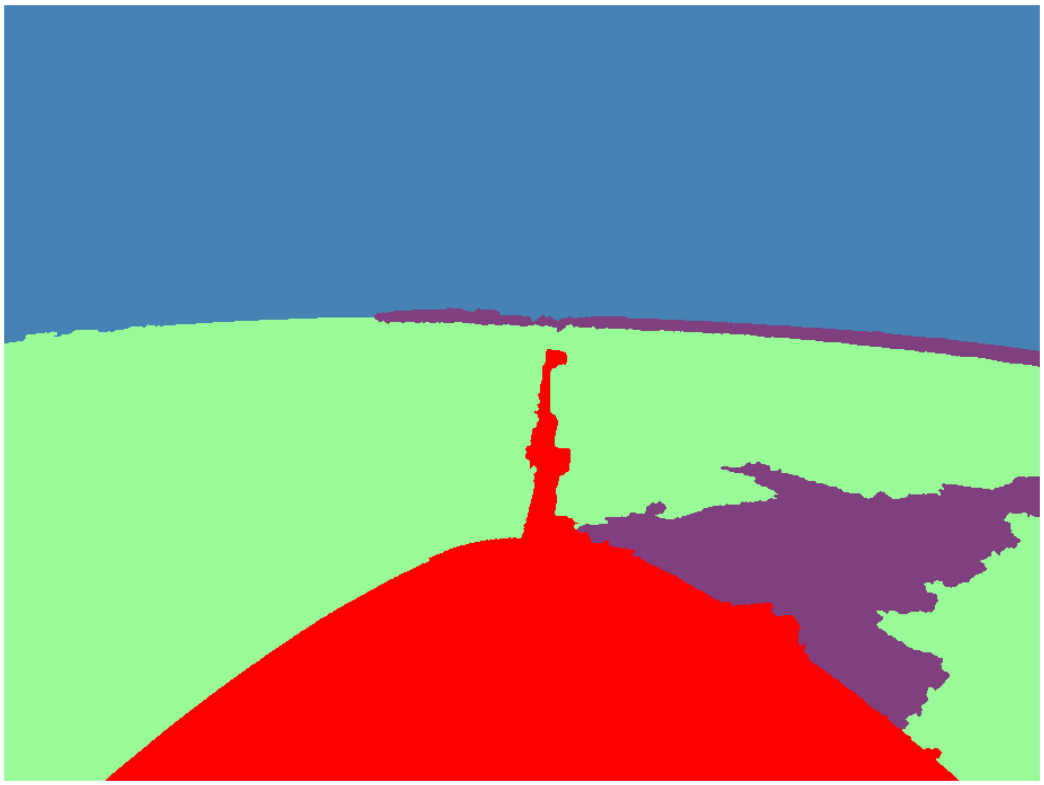


Figure 3.2: An example of a labelled training image showing the four classes: the boat in red, ocean in purple, sky in blue and ice in green.

data to any navigation or mapping system. An example of a labeled image from this data model can be seen in Figure 3.2. It is important to note that the labeled image is colorized for visualization purposes, and the actual input labeled images are grayscale ranging from 0-3.

3.1.2 Sea Ice Classification

The second dataset used in this study is also extracted from the same Nathaniel B. Palmer go-pro imagery, as described above. Unlike the last dataset, there is an increased focus on ice types and a variety of images used in the set. Of the entire

available images, every day was used in the creation, with the exception of days that only contained night-time footage. This decision to exclude the night-time images from the set is due to the small sample size and limited visibility of the actual ocean state from the camera location. This creates a comprehensive dataset capturing many environmental conditions and ice types encountered on the icebreaker journey, resulting in a significant variance of ice types in the dataset. For each of the days in the dataset, approximately 60 images were chosen at random to be labeled. Each of the images has a corresponding pixel-wise label, breaking the image down into eight distinct classes. The pixel-wise labels were created using careful and detailed labeling with the same watershed labeling tool used above [58].

The training dataset consists of 896 images, with approximately 53 images from each of the days. Each image has a corresponding pixel-wise label, breaking each image down into eight distinct classes. For validation, approximately six images were randomly selected from each of the 17 days of footage to give 99 images for validating the model. The test set is comprised of 102 images chosen from the 15 days used for training and an additional two days that were not used for the training and validation sets.

3.1.2.1 Sea Ice Classification Labels

The semantic segmentation ice classification dataset considers eight classes, four classes to describe the non-ice objects, and four classes for ice types. The non-ice objects are defined as follows:

- Ocean: all open water in the image.

- Vessel: all sections of the boat, including the mast, people, and flag visible on the front of the boat.
- Sky: all visible sky in the image.
- Lens Artifacts: any particles or objects on the camera lens obstructing the image, an example is seen in Fig. 3.3f.

The model is designed to find and label all pixels to each of the classes with a focus on separating the ocean and different ice classes. The proper classification of the ice is crucial as certain mislabeled ice would defeat the purpose of the entire labeling and would provide little relevant data to any navigation or mapping system. A labeled image from this data model showcasing the majority of the classes can be seen in Fig. 3.2. It is important to note that the labeled image is colorized for visualization purposes, and the actual input labeled images are grayscale ranging from 0-7. There are several characteristics related to sea ice considered when creating "egg codes" for ice charts [141]. The egg code of an ice chart reports the types of ice (based on thickness), concentration (using a scale of 10), and form of ice (based on the size of ice floes). This information is also used in the IMO Polar Operational Limit Assessment Risk Indexing System (POLARIS), where concentrations related to nine different ice types are used for creating the ice numeral risk rating as given in Table 1 of [142]. Accurate classification of these types requires an ice service specialist and other onboard sea ice observation aids [143]. As this work is basing ice type classification only on sea ice imagery and labeling is performed by a knowledgeable non-expert user, only four visually distinguishable ice types are used in this work. The non-expert labeler consulted an expert for classification validation and used reference

material [142] and [6] to follow established visual ice classification procedures when labeling the images. The considered sea ice types are as follows.

3.1.2.2 New Ice

New ice is one of the difficult ice types to classify. It is used as a general term for recently formed ice, including frazil ice, grease ice, and shuga. While these ice types vary in looks, they all are thin and weak formations of ice, loosely frozen together. In addition to the classical new ice, nilas is also included as new ice in this dataset. Nilas is a thin elastic crust of ice with a matte crust and up to 10 cm thick. While this is not new ice, the similarity in appearance and equivalent weakness of new ice make it a valid inclusion as New ice for classification. New ice is predominantly seen as smooth patches of water in the dataset with very little color. New ice is seen in Fig. 3.3c.

3.1.2.3 Grey Ice

Grey ice, in the classification, represents two classes of ice being both grey ice and grey-white ice. The main difference between these two types of ice is that the thickness is 10-15 cm thick for grey ice, whereas grey-white ice is young ice 15-30 cm thick. These classes have been combined to simplify the classification and the inability to calculate the thickness of the ice based on the current dataset. The grey color is the primary visual indicator for this ice; however, the ice shape and overall texture of the ice are additional clues to identify the ice. An example of grey ice is seen in Fig. 3.3b.

3.1.2.4 First-year Ice

First-year ice is described as floating ice of up to one year of growth and developed from young ice. The ice thickness ranges from 0.3 to 2 meters in thickness and is commonly seen as pack ice and sheets in the Arctic. In standards, this ice classification is broken down into additional categories based upon the thickness being:

- Thin First-year Ice/White Ice - first stage: 30-50 cm thick,
- Thin First-year Ice/White Ice - second stage: 50-70 cm thick,
- Medium First-year Ice: 70-120 cm thick,
- Thick First-year Ice: Greater than 120 cm thick.

However, due to the difficulty of determining the ice thickness based purely on a fixed camera front-facing view, the ice has been grouped into a single class. The main visual features used to detect and classify the ice for labeling are the solid white color and typical flat surface. The broken pieces of pans in rubble fields have also been defined as First-year ice to simplify the labeling process. In the dataset, it appears as both ice pans and sheets of ice. An example of first-year ice is seen in Fig. 3.3a.

3.1.2.5 Multiyear Ice

The last classification of ice for labeling is multiyear ice. This type of ice is old ice that has seen at least two summers melt. The visual distinction of this ice is the smooth hummocks with large interconnecting, irregular puddles on top of the ice. It has a blue color to it where bare. Unfortunately, in the Nathaniel B. Palmer imagery, there does not exist any multiyear ice due to it being an Antarctic voyage. Any additional

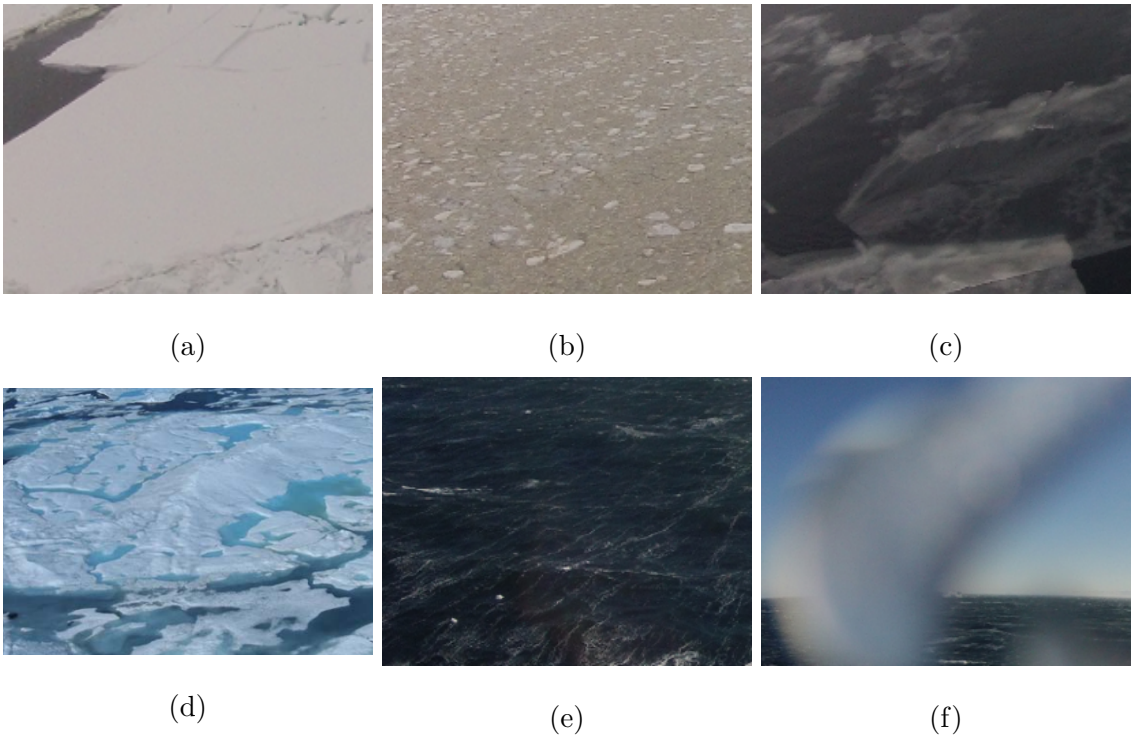


Figure 3.3: Examples of (a) first-year ice, (b) grey ice, (c) new ice, (d) multiyear ice, [6], (e) ocean, (f) lens artifact from the dataset.

expansions to the dataset will aim to include this ice classification as it is a critical type of ice to distinguish between. An example of multiyear ice is seen in Fig. 3.3d.

3.1.3 Training

The SegNet model was trained from scratch using a windows machine with an Intel i5-4670K CPU and 16 GB of RAM to train the detection networks. There were 382 images in our training set, resized to 918x688. The model was trained across five epochs, with the training performance evaluated after each iteration to determine if additional training was required. The PSPNet101 network was trained using transfer learning for the cityscapes 713x713 dataset. Due to the models' heavy training pro-

cess, a virtual machine with 16 vCPU and 60 GB of RAM was created and trained for nine epochs across the 382 training images resized to 713x713. For both of these models, due to the small dataset size, there was no batch size, and all images were used in each epoch. Additionally, the smaller epoch numbers were used to give the overall performance of both architectures to evaluate each model's performance.

For the ice classification dataset, to train both neural networks, a virtual machine on google cloud compute was created to optimize training. Both models were trained using a Linux-based, 16 vCPU with 60 GB of RAM. Although CPU was used to train the model due to ease of access, GPU will be used to expedite the training process for large datasets in future work. For the SegNet model, it was trained for 100 epochs with a batch size of 512 images. The network was trained from scratch as no pre-trained models were readily available in the model implementation. For the PSPNet model, the network was trained using transfer learning from the cityscapes dataset. Due to the transfer learning, the model was trained with 713x713 images. For both models, the intersection over union (IoU) and pixel-wise accuracy were computed after each epoch. The final weights used for both models were the models with the best performance across the validation set. In Fig. 3.4, the pixel-wise accuracies of validation sets for both architectures on both datasets are seen. It is noted that the PSPNet101 model has fewer epochs due to the intensive training time (2 hours per epoch). For both models, it is observed that the training set accuracy is higher than validation as expected; however, it is noted that neither model started to overfit at their max epoch. For the SegNet model, the best performance was determined to be at epoch 72, while the best performance for PSPNet101 is at the 24^{th} epoch.

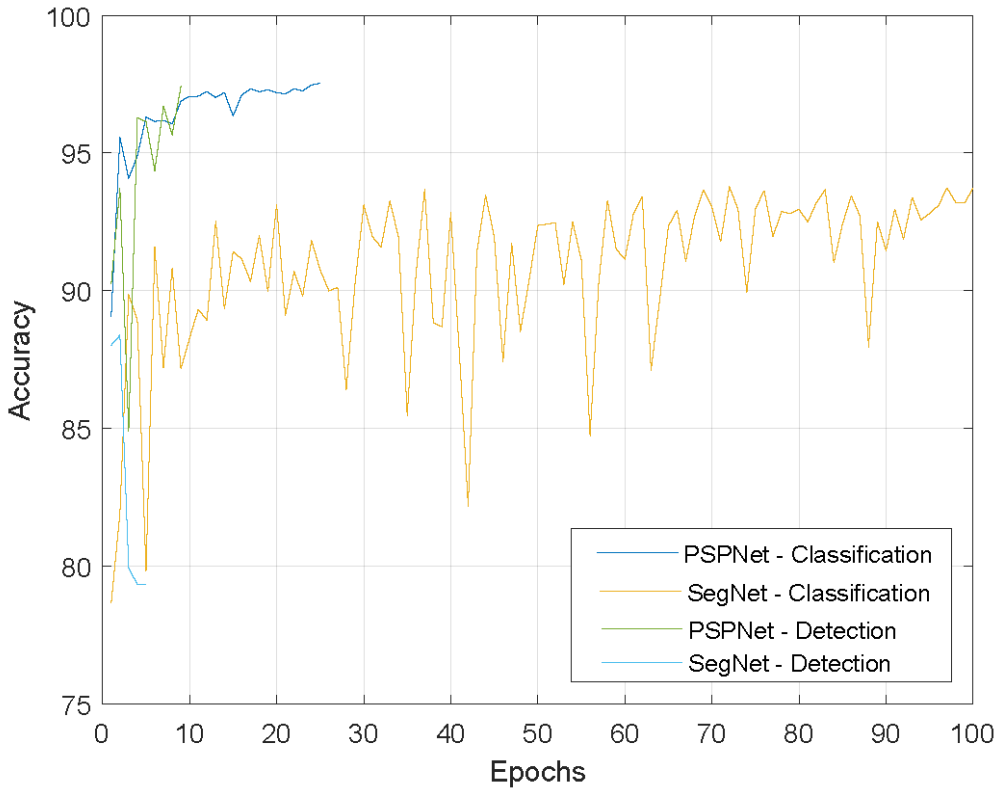


Figure 3.4: pixel-wise accuracy for the validation sets of both neural networks for the ice classification dataset and the ice detection dataset

3.2 Results

3.2.1 Evaluation

For sea ice detection, both the SegNet and PSPNet101 networks are evaluated on two performance metrics: pixel-wise accuracy and Intersection over Union (IoU). For semantic segmentation of ice classification, both networks were evaluated using two additional metrics, being false positive (FP) and false negative (FN). All four metrics

are described below:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.1}$$

$$FP\ rate(FPR) = \frac{FP}{FP + TN} \tag{3.2}$$

$$FN\ rate(FNR) = \frac{FN}{FN + TP} \tag{3.3}$$

$$IoU = \frac{area\ of\ intersection}{area\ of\ union} \tag{3.4}$$

where given a class (A), positives are the pixels that are predicted as belonging to that class. Negatives are the pixels that are predicted as not belonging to that class. True positive (TP): correct positive prediction; False positive (FP): incorrect positive prediction; True negative (TN): correct negative prediction; False negative (FN): incorrect negative prediction. IoU (Intersection Over Union) is a measure of how well the bounding boxes fit the actual location of an object. Where intersection and union are the intersection and union of the true and predicted pixel bounding boxes.

The results below were obtained on the respective test datasets, which were created to provide a range of polar icebreaker operational conditions. These images are sourced from the same days in which the neural network was trained; however, they differ from the images used for the training data. Where the images are sourced from an actual image feed of an icebreaker, they provide an excellent metric on how the network would perform in the field. While this network only provides the results for daytime images using one vessel, it showcases the potential and overall performance of semantic segmentation for ice detection and classification.

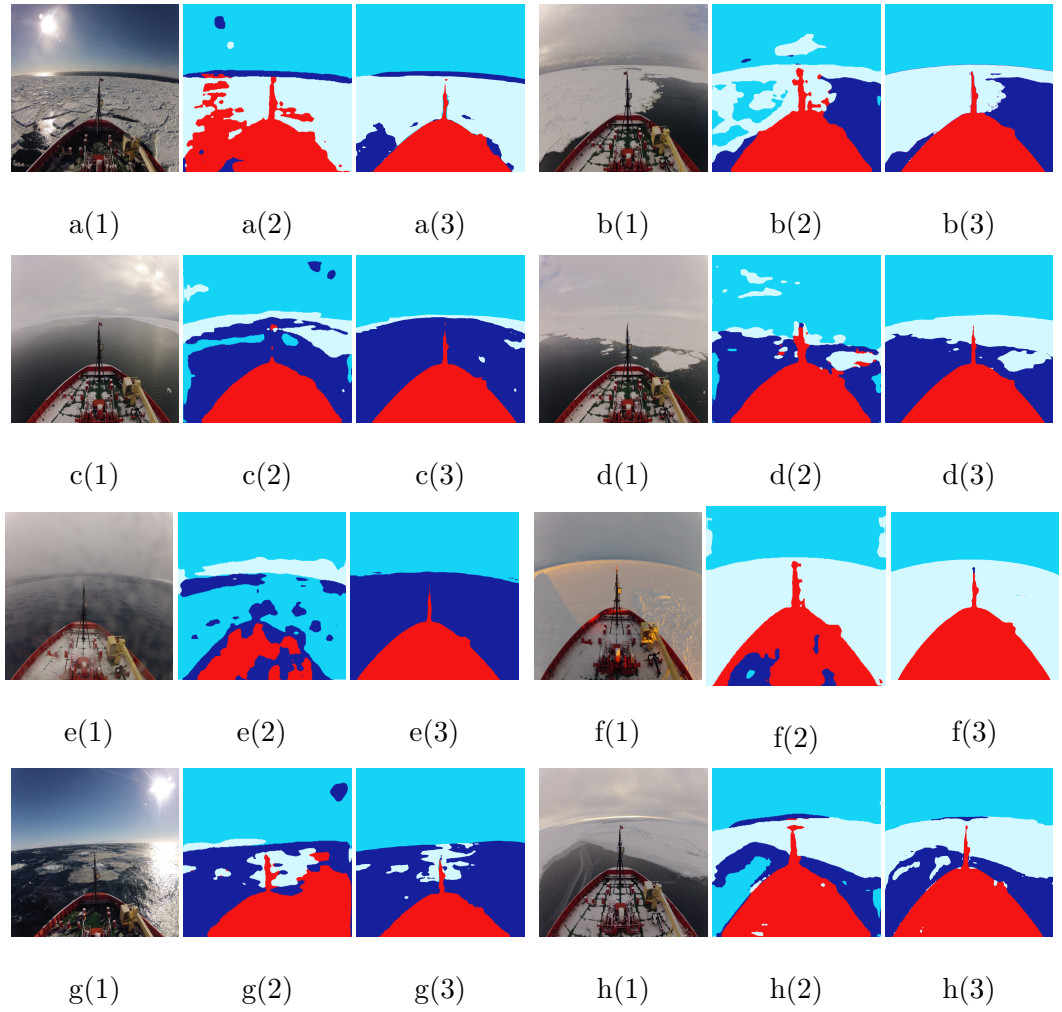


Figure 3.5: Comparison of segmentation results of both models for test images. In images (a) - (f), the overall segmentation performances of both neural networks are observed on images from the test dataset. (g) - (h) demonstrate the performance of architectures on images from days of the [7] dataset that was unused for training. (1), (2), (3) are used to indicate the image, SegNet prediction of classes, and PSPnet prediction of classes, respectively, with the boat in red, the ocean in dark blue, the sky in cyan, and ice in light blue.

3.2.2 Sea Ice Detection Results

The performance of both models for sea ice detection is presented in Table 3.2. PSPNet101 outperformed SegNet in every metric with an average IoU of 90.1% compared to 69.8% for SegNet, demonstrating a much greater contextual awareness. The pixel-wise accuracies of the models present that the SegNet architecture has significantly more classification errors as no class is over 95% compared to PSPNet, which boasts all classes at least 96% accurate. When analyzing the actual classification results in Fig. 3.5, the SegNet results have large patches of misclassifications and struggle with differentiation between classes with similar coloring. Additionally, any considerable variation, including water droplets on the lens and midday glare off the ice, causes errors. For PSPNet101, there are few misclassifications visible in the results, with the model performing well across the variety of polar ocean environments presented. In the case of untrained imagery as seen in 3.5 g & h, PSPNet101 has a more robust performance in the unfamiliar conditions with some misclassification occurring; however, no large errors. SegNet, however struggles more with significant misclassification in the sky and vessel pixels. The PSPNet101 neural network's actual performance as it is run across a section of day's journey can be found in the following video link https://youtu.be/Otx2Pu0XUdM.

3.2.3 Sea Ice Classification Results

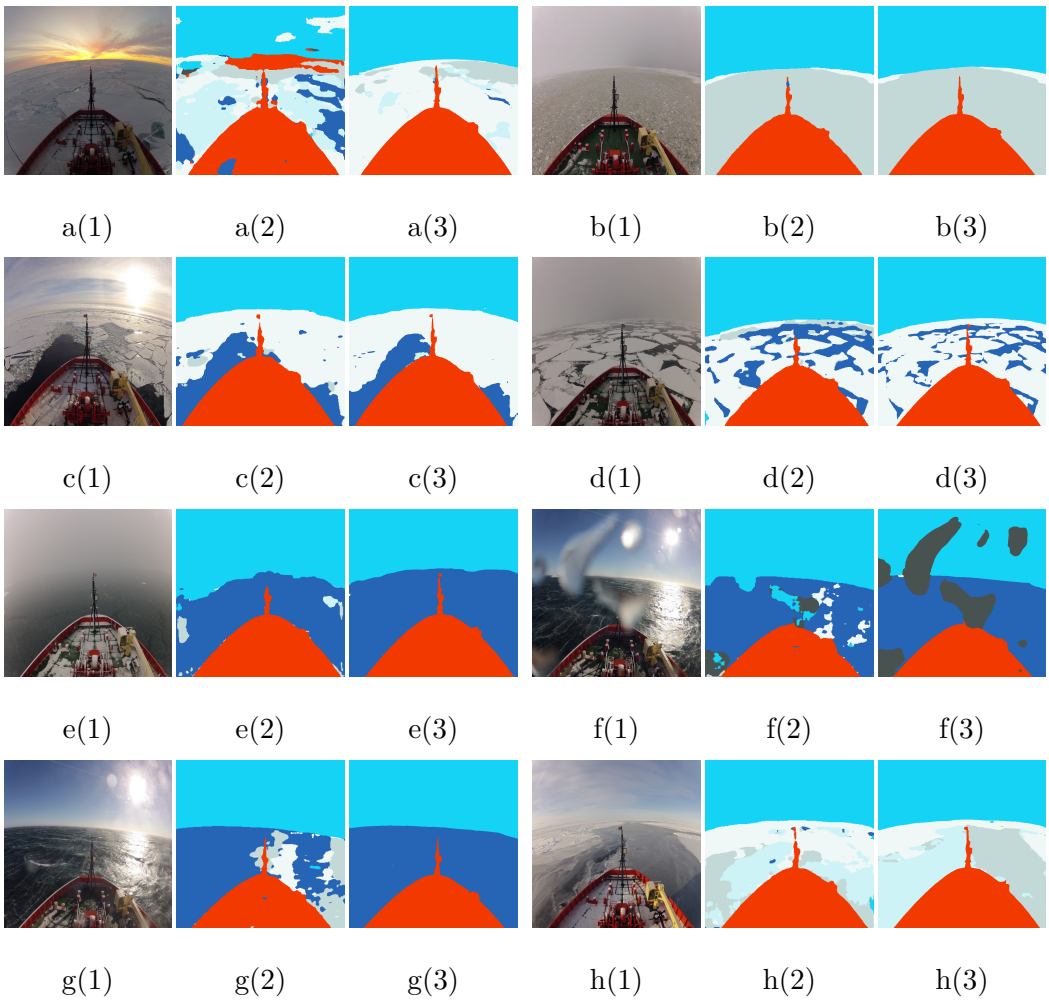


Figure 3.6: Comparison of segmentation results of both ice classification models using test images. (1), (2), (3) are used to indicate the image, SegNet prediction of classes, and PSPnet prediction of classes, respectively, with the boat in red, the ocean in dark blue, the sky in cyan, first-year ice in white, grey ice in grey-blue, and new ice in pale blue. In images, both neural networks' overall segmentation performances are observed on images from the test set.

Metric	Network	Average	Ocean	Vessel	Sky	Ice
IoU	SegNet (Detection)	69.8	44.1	80.4	74.7	80.0
IoU	PSPNet101 (Detection)	90.1	80.8	89.3	95.7	94.7
Pixel-wise	SegNet (Detection)	92.0	89.5	95.5	89.3	93.6
Pixel-wise	PSPNet101 (Detection)	97.8	96.6	97.7	98.4	98.5

Table 3.2: IOU & pixel-wise accuracy of the PSPNet101 and SegNet deep neural networks on the ice detection datasets.

Metric	Network	Average	Ocean	Vessel	Sky	New Ice	First-Year Ice	Grey Ice	Lens Artifacts	Ice
IoU	SegNet (Classification)	53.6	66.9	96.9	91.2	31.9	73.5	59.1	9.0	54.8
IoU	**PSPNet101 (Classification)**	**75.5**	**89.9**	**98.9**	**99.0**	**76.1**	**93.4**	**81.9**	**65.0**	**83.8**
Pixel-wise	SegNet (Classification)	96.7	96.9	99.3	96.7	95.3	92.9	96.4	99.7	94.9
Pixel-wise	PSPNet101 (Classification)	99.2	99.2	99.7	99.7	98.5	98.3	98.8	99.9	98.6
FN Rate	SegNet (Classification)	28.0	16.6	1.5	3.2	59.3	20.8	4.1	90.4	28.1
FN Rate	PSPNet101 (Classification)	7.7	8.4	0.5	0.5	12.4	4.6	2.6	25.3	6.5
FP Rate	SegNet (Classification)	2.0	2.1	0.5	3.4	1.6	2.7	3.6	0.0	2.6
FP Rate	PSPNet101 (Classification)	0.5	0.1	0.2	0.3	0.9	0.7	1.1	0.0	0.9

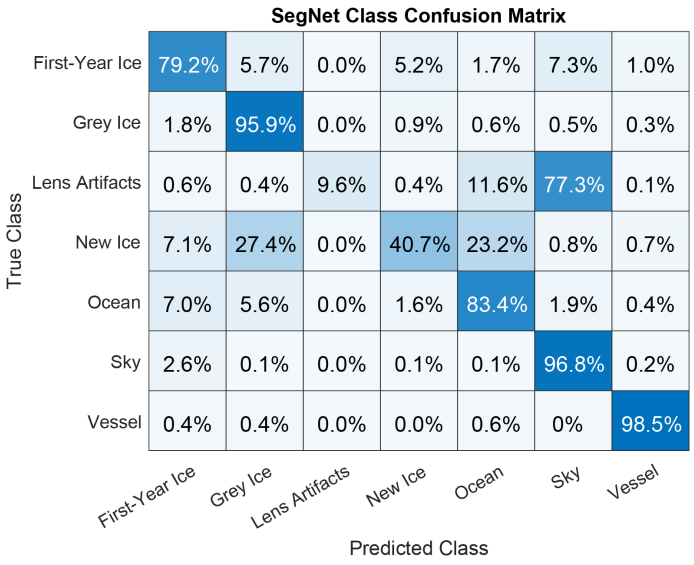
Table 3.3: IOU, pixel-wise accuracy, false positive rate, false negative rate performance of the PSPNet101 and SegNet deep neural networks on the ice classification dataset. The ice class represents a combination of the accuracies for each ice type in the classification dataset.

Table 3.3 presents the results for the SegNet and PSPNet architectures for the sea ice detection dataset. SegNet was only able to achieve an average IoU of 53.6%, while the model performed well for the vessel and sky classifications; it only reached an IoU of 54.8% for the combined ice types. PSPNet101 had a higher average IoU at 75.5% with a combined ice classification IOU of 83.8%. For every classification IoU, PSPNet presents a higher accuracy. The pixel-wise accuracies between both models suggest a more promising result for SegNet. However, in each class, it is marginally outperformed by PSPNet101. In Fig. 3.6, PSPNet101 demonstrates its strength across the varying conditions with no apparent errors in any of the results. SegNet, on the other hand, presents difficulties with differing environments. The classification of the vessel and sky is consistent. However, there are clear errors in the results, including sky as ice due to the ship shadows and errors from the sun's reflection off the ocean. While the SegNet architecture has poor overall accuracy for segmentation, the PSPNet101 demonstrated excellent potential in the field. Overall, PSPNet101 had a higher accuracy than the two models. PSPNet101 has a good average IoU performance, along with excellent IOU for all classes, except for new ice and lens artifacts. SegNet IoU demonstrates that while it had good pixel accuracy, the errors significantly impacted the IoU, only resulting in an average IoU of generating 53.6% and having two classes perform poorly with individual IoUs of 31.9% and 9.0%. This disparity in network performance is not unexpected, as PSPNet101 is a significantly more complex and deep neural network than SegNet. The PSPnet101 architecture's performance across an entire day of images can be found at the following link https://youtu.be/LisIE47Kz28.

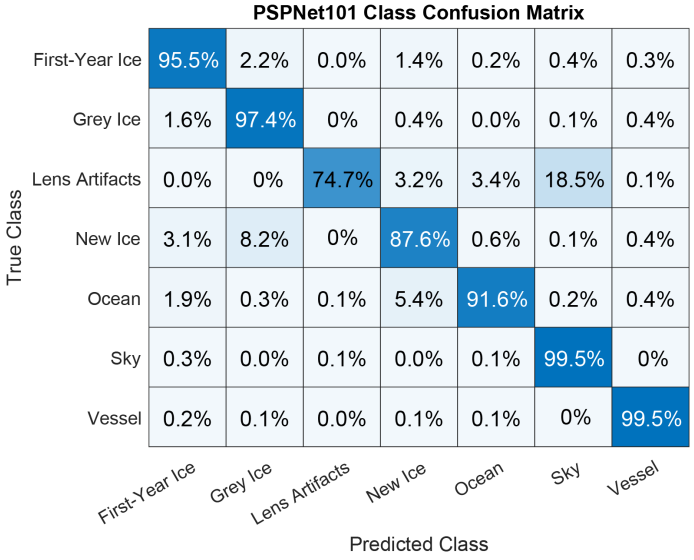
For SegNet, the rates of the FN and FP are typically higher than that of PSP-

Net101. There is an extremely high rate of false negative detection for new ice and lens artifacts. The ocean and first-year ice classes also are classified wrong, with only the vessel, grey ice, and sky classes having a low false negative rate. The false positive rate average for SegNet is only 2.0%, with grey ice, sky, and ocean commonly mistaken. The lens artifacts class has a low FP rate due to the limited classification of the class in the model. For PSPNet, new ice and lens artifacts have the highest FN rate, with new ice and grey ice having the highest FP rate. This is as expected because both new ice and lens artifacts have the most confusion between other classes. The vessel, sky, and first-year ice have minimal FN and FP rates.

In both networks, they appear to struggle with the proper classification of new ice and lens artifacts. For the lens artifacts, these appear as water droplets on the lens and have little representation in the dataset. Due to the labeling method, the overall region labeled as lens artifacts is challenging to define correctly and overlaps with the more common classifications. For the new ice, it confused most with first-year ice and the ocean class, as seen in Fig. 3.7. From the confusion matrix given in Figure 3.7a, it is evident that the SegNet has a poor performance on the small classes, with a widespread of false positives. The new ice classification is commonly confused for first-year ice, grey ice, and ocean, and the lens artifacts are more often confused as both sky and ocean classes than its class. The first-year ice is also occasionally confused with the sky, which is the case in Figure 3.6 h(2) and k(2), where there are sections of the ice labeled as the sky. Lastly, the ocean class is seen to be mistaken as ice classes. For PSPNet, Figure 3.7b has the same issue with new ice confused with grey ice and first-year ice; however, it is to a much smaller degree. There is little confusion observed between non-ice classes and ice classes. Lastly, the first-year ice



(a)



(b)

Figure 3.7: (a) The confusion matrix for the SegNet architecture on the ice classification dataset. (b) The confusion matrix for the PSPNet101 architecture on the ice classification dataset.

and grey ice classes are occasionally confused with each other around 2.2% and 1.6%, respectively. However, this confusion, as seen in Fig. 3.6 is mainly caused by these classes commonly bordering on each other.

Although there is no directly comparable work to benchmark the performance, we provide a comparison of our results with the following closely related applications. Work in [42] applies a custom lightweight CNN architecture using three convolutional layers, which achieves an overall pixel-wise accuracy of 91.6% for a 77-image validation dataset. Similarly, [40] achieves a mean IOU of 88.1% and pixel accuracy of 95.9% using a custom CNN for 3-class segmentation. Finally, [41] uses the ResNet architecture to achieve an accuracy of 90% for a 39 image test dataset for ice object classification. The results achieved for detection is presented in Table 3.2 is comparable with the performance of these related application domains which use CNNs. It is important to note that a higher complexity architecture than the networks used in [40–42] is needed for the proposed work due to the challenging application domain. The lightweight CNN architecture considered in this work, SegNet, significantly underperformed for ice class classification as shown in the comparison of the results given in Table 3.3.

Fig. 3.8 illustrates the performance processing time tradeoff between the two networks for both sea ice detection and classification. The tradeoff between the models is apparent, where PSPNet presents a more accurate segmentation at the cost of speed. There is little change in the frames per second of the PSPNet model between both datasets, indicating that the number of classes in the classification dataset does not significantly impact the model speed.

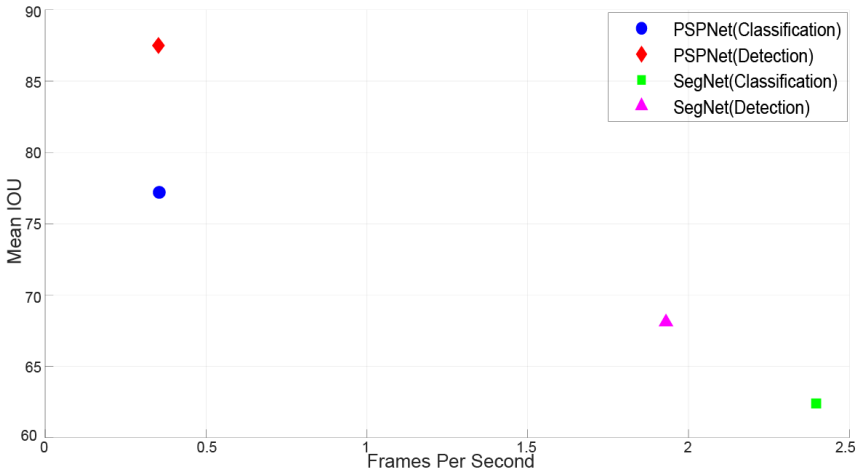


Figure 3.8: Speed vs. mean IOU for both architectures for both datasets. A clear divide between accuracy and speed is presented between both models. The running time is measured on an AMD Ryzen 7 3800X with 32 GB of RAM.

3.3 General Chapter Summary

In this chapter, two custom semantic segmentation datasets are created for training and evaluating neural network performance of sea ice detection and sea ice classification onboard polar vessels. Two semantic segmentation neural networks, SegNet and PSPNet101, are implemented for sea ice detection into four distinct classes in polar waters and evaluated by their pixel-wise and IoU performance on the test set. The two neural networks are investigated on the sea ice classification dataset, which expands on the number of classes in the images, including three ice types and lens artifacts. The performance of both algorithms is again compared versus each other with increased metrics including false positive, false negative, and confusion matrices. Lastly, the speed tradeoff of both neural networks is evaluated to determine the

overall capability for in-situ ice classification for polar vessels.

The results show that the PSPNet101 architecture can attain 97% accuracy for sea ice detection and achieves false negative rates of 12.4%, 2.6%, and 4.6% for new, gray, and first-year ice types classification, respectively. In the future, this work should be expanded further increasing the training dataset size and incorporate data from different polar voyages. Increased segmentation of the detected first-year ice is desired to further meet the desired classifications. Additionally, the detection of the freeboard of ice and rolling faces of ice is of crucial importance to realize automated navigation and mapping aids for arctic environments.

Chapter 4

Fault Detection in Small Satellite Attitude Determination

This chapter presents a magnetometer fault detection and recovery method for small satellites. Magnetometers are prone to pick up any short-term anomalies of the local magnetic field arising from data telemetry, magnetorquer, or reaction wheel activation of the satellites. Furthermore, the attitude estimation solution will be affected by persistent changes in the local magnetic field, such as sensor degradation, deployable panels, and thermal cycling. For small satellite attitude determination, an accurate magnetometer signal is necessary to keep precise attitude estimation. Therefore a magnetic fault detection method is required to disable the magnetometer updates in the short term. In case of persistent anomalies, it becomes necessary to recalibrate the magnetometer parameters using orbital data to account for the changes in the mathematical model. For this purpose, this chapter proposes a fault detection and recovery method using advances in estimation theory and machine learning.

State-of-the-art attitude estimation and fault detection methods rely on the confidence bounds of the expected sensor readings as a means of detecting possible anomalies [132]. However, these confidence bounds are not correctly tracked, especially in the case of small satellites, because fewer sensors support the attitude estimation solution. Furthermore, the traditional methods [144] do not adhere to theoretical constraints that exist in the estimation problem, i.e., the techniques are not observability consistent [145]. Therefore, there is a need to develop observability consistent filters for small satellite attitude estimation, which can keep accurate track of confidence bounds leading to correct identification of anomalies in sensor readings.

The state-of-the-art magnetic calibration methods [146] require custom calibration runs for the magnetometer that can ideally be performed in a lab setting. The magnetic readings from these calibration trajectories (at a minimum 360-degree rotation along two orthogonal axes) are used to identify any bias, and scaling parameters of the magnetometer [147]. These values are hardcoded in the estimation filter to assist with the estimation process. However, during orbit, these calibration runs cannot be performed. It would require a long duration to complete rotations with minimal magnetorquer and reaction wheel activation corrupting the data. More importantly, the execution will need the attitude estimator to be already functioning to control the trajectory. However, by considering all available data (gyroscope, sun sensor, biased magnetometer), it can be shown that the unknown bias of the sensor is observable for an orbit [148] given an accurate regression procedure is performed on the data. Therefore, a machine learning regression procedure needs to identify these unknown magnetometer parameters using satellite data during an orbit and execute it in a suitable way to account for the nonlinear attitude error parameterization applicable

to the estimation problem.

The contributions of this work are as follows:

- A novel filter with quaternion right attitude error parameterization for small satellites. This attitude estimation method allows the filter to maintain accurate confidence levels of the estimation for a prolonged duration compared to state-of-the-art methods and, as a result, accurately detect anomalies of sensor readings.

- A machine learning regression method using nonlinear attitude parameterization to recalibrate the magnetometer parameters using captured orbital data. This machine learning regression approach considers all data in the calibration process including, the sun sensor, gyroscope, and any other sensors of the small satellite, making it capable of optimizing the parameters without the need for full 360° 3D calibration rotations of the magnetometer.

This chapter is outlined as follows: first, the mathematical model used for attitude determination is presented in Section 4.1 leading to a new attitude estimation algorithm; then, the observability analysis and constraints are analyzed in Section 4.2. Section 4.3 presents the simulation environment and simulation results for the proposed parameterization. Section 4.4 presents the physical test environment and physical test results for the proposed parameterization.Section 4.5 discusses the proposed fault detection strategy. **??** presents the machine learning regression algorithm that uses the orbital data for failure recovery of the magnetometer. The final section, Section 4.7 presents the simulation environment and simulation results for the fault

detection algorithm, fault isolation algorithm, and the machine learning algorithm. Lastly, a general summary of the chapter is provided in Section 4.8.

4.1 Satellite Attitude Dynamics

The position and orientation of a satellite are defined using the body frame {B} attached to the geometric center of the satellite, and the Earth-centered inertial frame {ECI} is used as the reference frame for navigation. To simplify the model, we assume that the gyroscope and the magnetometer sensor coordinate systems are aligned and located at the body frame {B}. Each sun sensor i is defined by a sensor frame $\{S_i\}$ with axes aligned with the body frame. The magnetometer requires a magnetic field model defined with respect to the Earth-centered earth-fixed {ECEF} frame of reference. The overall coordinate systems related to the attitude determination and control subsystem (ADCS) are seen in Figure 4.1.

The nonlinear state-space model of the system can be represented using the following model:

$$\begin{aligned} \dot{\boldsymbol{x}} &= f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w}) \\ \boldsymbol{y} &= h(\boldsymbol{x}, \boldsymbol{v}) \end{aligned} \tag{4.1}$$

where $\boldsymbol{x}$ is the system state vector, $\dot{\boldsymbol{x}}$ is the first order derivative of the system state vector, $\boldsymbol{u}$ is the input vector, and $\boldsymbol{w}$ is the process noise. Vector $\boldsymbol{y}$ is the measurement vector, and $\boldsymbol{v}$ is the measurement noise vector. The state vector has a dimension of 7 and is defined as:

$$\boldsymbol{x} = \left[{}^{ECI}\boldsymbol{q}_B^T \; \boldsymbol{b}_w^T \right]^T \tag{4.2}$$

the state includes the unit quaternion of the satellite, ${}^{ECI}\boldsymbol{q}_B$ corresponding to rotation

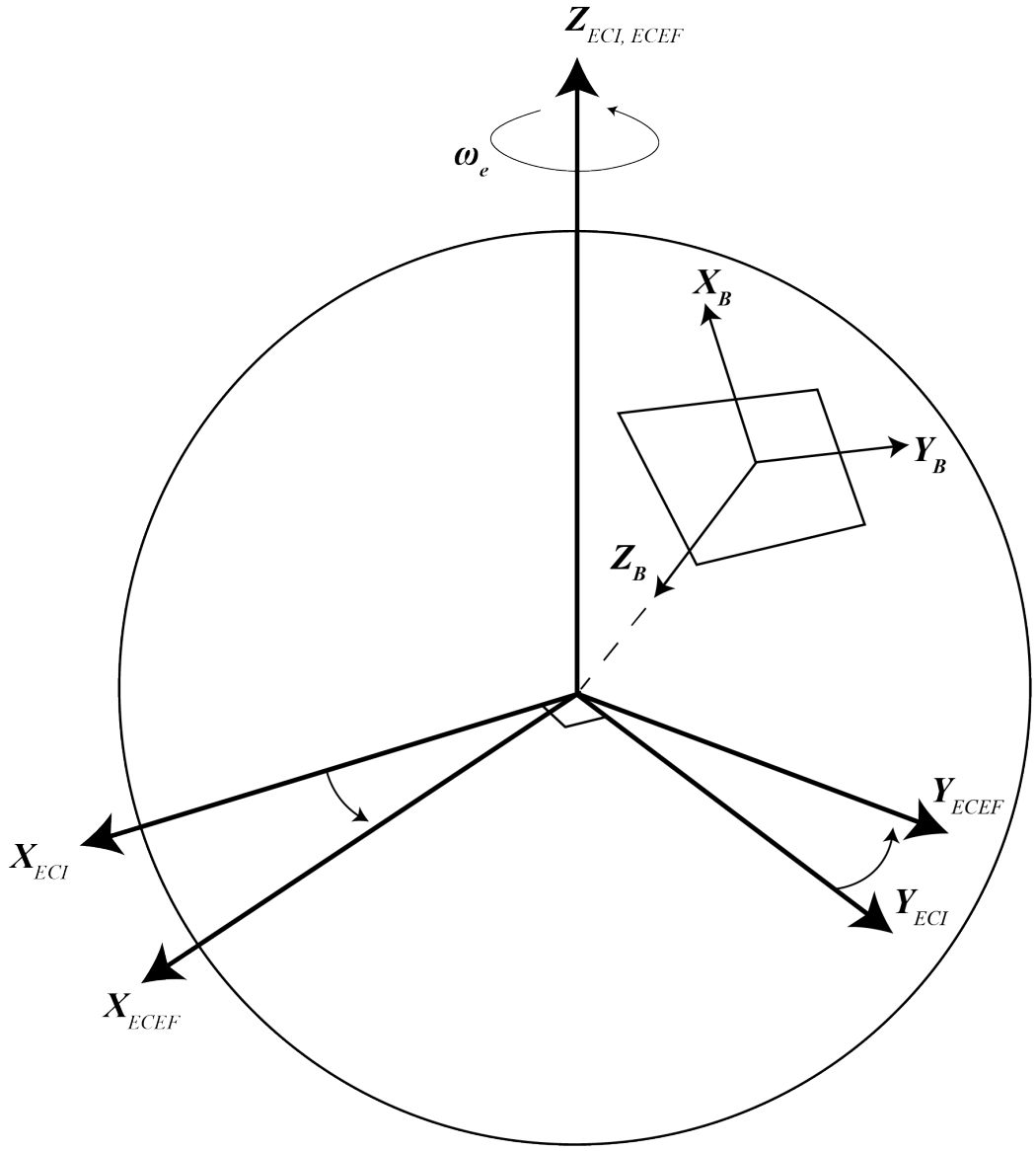


Figure 4.1: Coordination systems related to attitude on satellites. {B} is the body frame, {ECI} is the Earth-centered inertial frame located at the center of Earth and fixed, {ECEF} is the Earth-centered, Earth-fixed frame situated in the center of Earth and rotates.

from the body-fixed frame {B} to the ECI frame {ECI} and the gyroscope bias, $\boldsymbol{b}_w$ expressed in frame {B}. To simplify the equations, $\boldsymbol{q}$ will be used to denote the quaternion orientation of the satellite.

The ADCS rotation dynamics corresponds to the following equation:

$$^{ECI}\dot{\boldsymbol{q}}_B = 0.5\boldsymbol{q} * {}^{B}\boldsymbol{\omega} \tag{4.3}$$

where $^{B}\boldsymbol{\omega}$ is the angular velocity of the satellite corresponding to the rotation relative to the {ECI} frame expressed in the {B} frame. This is driven using measurements from the gyroscope given by:

$$\begin{aligned} \boldsymbol{\omega}_m &= {}^{B}\boldsymbol{\omega} + \boldsymbol{b}_\omega + \boldsymbol{\nu}_{\omega_m}, \\ \dot{\boldsymbol{b}}_\omega &= \boldsymbol{\nu}_{b_\omega} \end{aligned} \tag{4.4}$$

where $\boldsymbol{\omega}_m$ is the gyroscope measurements vector, $\boldsymbol{\nu}_{\omega_m}$ and $\boldsymbol{\nu}_{b_\omega}$ are stochastic Gaussian noise variables for gyroscope measurement and gyroscope bias random walk process respectively.

The satellite model in (4.1) only has one input vector being the gyroscope measurement, giving:

$$\boldsymbol{u} = \boldsymbol{\omega}_m$$

The system noise vector in (4.1) is defined as $\boldsymbol{w} = [\boldsymbol{\nu}_{\omega_m}^T \ \boldsymbol{\nu}_{b_\omega}^T]^T$.

The measurement vector of the satellite has a dimension of 9 and is defined as:

$$\boldsymbol{y} = [\boldsymbol{y}_b^T \ \boldsymbol{s}_{fine}^T \ \boldsymbol{s}_{coarse}^T]^T \tag{4.5}$$

the vector includes the magnetometer measurement $\boldsymbol{y}_b$ expressed in {B}, the fine sun sensor measurement $\boldsymbol{s}_{fine}$, and coarse sun sensor measurement $\boldsymbol{s}_{coarse}$. The mathematical models of these measurements are given in the following sections.

4.1.1 Magnetometer Model

A magnetometer sensor outputs the magnetic field strength at any point during an orbit corresponding to the latitude and longitude of the point. To properly be able to simulate the magnetometer model during the orbit of the satellite, the exact position of the satellite in {ECEF} frame is required. The magnetic field reference vector can be obtained by using the orbital position of the satellite and the world magnetic model. That is:

$$\boldsymbol{b_e} = {}^{ECEF}R_{ECI}^{T}\, WMM({}^{ECEF}R_{ECI}{}^{ECI}\boldsymbol{p})$$

where $\boldsymbol{b_e}$ is the magnetic field reference vector for the satellite position, ${}^{ECI}\boldsymbol{p}$ is the satellite position expressed in the {ECI} frame, and ${}^{ECEF}R_{ECI}$ is the rotation matrix from {ECI} to the {ECEF} frame. The world magnetic model (WMM) [149] gives the magnetic field strength using the current magnetic database. Using the obtained magnetic field strength vector, the satellite magnetic field strength measurement is given as:

$$\boldsymbol{y}_b = R_q^T \boldsymbol{b_e} + \boldsymbol{\nu}_b \tag{4.6}$$

where $R_q := {}^{ECI}R_B$ is the quaternion rotation matrix from the {B} to body frame {ECI}, and $\boldsymbol{\nu}_b$ is the measurement error in the magnetometer. This equation gives the magnetic field strength measured at the satellite expressed in the body frame {B}. The noise of the sensor is modelled as stochastic Gaussian noise.

4.1.2 Sun Sensor Models

In principle, a sun sensor measures the relative direction of the sun with respect to the satellite body frame. This output measurement is a vector called the sun vector

measurement and is modelled as:

$$^{B}\boldsymbol{s} = R_q^T \, ^{ECI}\boldsymbol{s}_{ref} + \boldsymbol{\nu}_s \tag{4.7}$$

where $\boldsymbol{\nu}_s$ is the measurement noise of the sensor, and $^{ECI}\boldsymbol{s}_{ref}$ is the sun reference vector. The sun reference vector is obtained by using the sun's relative position to the satellite's orbital position with respect to the {ECI}. To simplify the sun's model for the simulation, the Earth is assumed to be in a fixed position relative to the sun in the {ECI} frame. This assumption is valid for any small number of full orbit simulations, where the Earth would not rotate drastically around the sun in a period of a standard 90-minute orbit. Given that the Earth has an orbital period of 365 days around the sun, the change of the Earth's position for any given day, which equates to 16 orbits of the international space station (ISS), is less than 1 degree around the sun. The reference sun vector is obtained by:

$$^{ECI}\boldsymbol{s}_{ref} = \frac{\boldsymbol{p}_{sat} - \boldsymbol{p}_{sun}}{||\boldsymbol{p}_{sat} - \boldsymbol{p}_{sun}||}$$

where $\boldsymbol{p}_{sun}$ is the sun position expressed in the {ECI} frame, and $\boldsymbol{p}_{sat}$ is the satellite orbital position expressed in the {ECI} frame.

The satellite has two types of sun sensors, which both produce slightly different results. The first sun sensor, a fine sun sensor is a high precision sensor that gives the sun vector as outlined in (4.7). Small satellites such as the Killick-1 satellite of MUN, are typically only equipped with one fine sun sensor. As a result, the field of view of the sensor should be modelled to see if the sun sensor measurement is available. The

field of view model is implemented as follows:

$$\begin{aligned} \boldsymbol{p}_{fs} &= R_q^T \boldsymbol{p}_{fs_B} \\ \theta &= \text{atan2}(\frac{\boldsymbol{p}_{fs} \times {}^{ECI}\boldsymbol{s}_{ref}}{\boldsymbol{p}_{fs} \cdot {}^{ECI}\boldsymbol{s}_{ref}}) \end{aligned} \tag{4.8}$$

where $\boldsymbol{p}_{fs}$ is the position of the fine sun sensor with respect to the sensor {S} frame, and $\boldsymbol{p}_{fs_B}$ is the position of the fine sun sensor expressed in the body frame {B}. Variable θ is the four-quadrant angle between the fine sun sensor and sun reference vector used to determine if the sun is in the sensor's field of view. If the sun is not in the field of view selected as 45°, the sensor outputs a zero vector.

The second type of sun sensor, a coarse sun sensor, gives the sun reference vector a more extensive uniformly distributed noise. Coarse sun sensors exist as an array of sensors with at least one coarse sun sensor on every face of the satellite. Each sensor determines the angle between the sun and sensor using the same equations in (4.8). The minimum angle between each of the sensors and the sun is taken, and the sensor position with respect to the body frame is taken as the output sun reference angle. This is a simplification of the sun position determination that assumes one sensor is in full view instead of a combination of each of the visible sensors.

Lastly, sun sensors only operate when the sun is in the field of view of the satellite. To simplify the equations to determine if the Earth occults the sun, the satellite position, Earth and sun are projected to 2D as seen in Figure 4.2. To determine if the satellite has visibility of the sun, its position is checked to see if it falls within the area of no visibility behind earth. To accomplish this process, the barycentric weights of the satellite are calculated and compared to the three points of the gray triangle [150]. If any of the weights are outside of the range of 0-1, the satellite is outside of

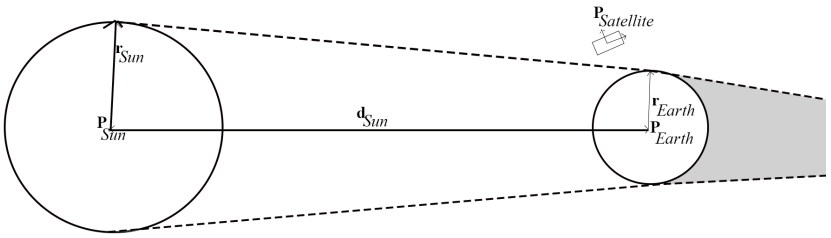


Figure 4.2: A 2D approximation of the sun, Earth and satellite position for the sun reference angle. The area shaded in gray is the area where the satellite has no visibility of the sun.

the polygon.

4.1.3 Filtering algorithm

In this section, we present the state-of-the-art attitude estimation method used by satellites, i.e., the multiplicative extended Kalman filter (MEKF) [117] using an error state Kalman filter design workflow [38]. The error state Kalman filter uses the system's input and sensor noise parameters to find an optimum solution for the unknown state vector. The algorithm is optimized for linear systems with Gaussian noise parameters. However, in practice, the Kalman filter is often used for nonlinear systems [151, 152] such as the ones considered here. The drawback is that the method has a smaller domain of attraction where the estimated state vector $\hat{\boldsymbol{x}}$ approaches the actual state vector $\boldsymbol{x}$. The algorithm effectively attempts to drive the error state (difference between $\hat{\boldsymbol{x}}$ and $\boldsymbol{x}$) to zero.

The error state vector corresponds to the geometric distance between the actual state vector $\boldsymbol{x}$ and the estimated state vector $\hat{\boldsymbol{x}}$. The error state is defined as $\tilde{\boldsymbol{x}} = \boldsymbol{x} \ominus \hat{\boldsymbol{x}}$, where the inverse mapping operation $\ominus$ is used to represent the geometric difference

using lie groups as seen in [123]. For the bias, this mapping is standard subtraction however this operation changes for the quaternion. The error for the quaternion is represented using a rotation vector $\tilde{\boldsymbol{\theta}}$ which is defined as, $\tilde{\boldsymbol{\theta}} = \text{Log}_{S3}(\hat{\boldsymbol{q}}^{-1} * \boldsymbol{q})$ where $*$ represents quaternion multiplication, and $\text{Log}_{S3}(\cdot)$ is the mapping from the quaternion representation to the rotation vector representation. Similarly, the actual state $\boldsymbol{x}$ can be defined as $\boldsymbol{x} = \hat{\boldsymbol{x}} \oplus \tilde{\boldsymbol{x}}$, where $\oplus$ is the retraction operation used to represent the error perturbation of the estimated state. For bias it is vector addition $\boldsymbol{b} = \hat{\boldsymbol{b}} + \tilde{\boldsymbol{b}}$ but for quaternions it is defined as $\boldsymbol{q} = \hat{\boldsymbol{q}} * \text{Exp}_{S3}(\tilde{\boldsymbol{\theta}})$ where $\text{Exp}_{S3}(\cdot)$ is the mapping from rotation vector representation to the quaternion representation.

The definitions for $\text{Exp}_{S3}(\cdot)$ and $\text{Log}_{S3}(\cdot)$ are as follows:

$$\boldsymbol{q} = \text{Exp}_{S3}(\theta \boldsymbol{u}) \triangleq \cos\left(\frac{\theta}{2}\right) + \boldsymbol{u}\sin\left(\frac{\theta}{2}\right) \tag{4.9}$$

$$\theta \boldsymbol{u} = \text{Log}_{S3}(\boldsymbol{q}) \triangleq 2\boldsymbol{v}\frac{\arctan(||\boldsymbol{v}||, w)}{||\boldsymbol{v}||} \tag{4.10}$$

where $w = q_1,\ \boldsymbol{v} = (q_2, q_3, q_4)$. Vectors $\boldsymbol{u} = \frac{\boldsymbol{\theta}}{||\boldsymbol{\theta}||}$ and $\theta = ||\boldsymbol{\theta}||$ are the axis and the angle corresponding to the rotation vector $\boldsymbol{\theta}$ respectively.

The resulting error state vector can be summarized as:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} \ominus \hat{\boldsymbol{x}} = \begin{pmatrix} \tilde{\boldsymbol{\theta}} \\ \tilde{\boldsymbol{b}}_{\boldsymbol{w}} \end{pmatrix} = \begin{pmatrix} \text{Log}_{S3}(\hat{\boldsymbol{q}}^{-1} * \boldsymbol{q}) \\ \boldsymbol{b}_{\boldsymbol{w}} - \hat{\boldsymbol{b}}_{\boldsymbol{w}} \end{pmatrix}. \tag{4.11}$$

Similarly, the error perturbations of the estimated state can be summarized in (4.12). The first-order linearizations of the errors can be defined assuming small angles:

$$\hat{\boldsymbol{x}} \oplus \tilde{\boldsymbol{x}} = \begin{pmatrix} \tilde{\boldsymbol{\theta}} \\ \tilde{\boldsymbol{b}}_{\boldsymbol{w}} \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{q}} * \mathrm{Exp}_{S3}(\tilde{\boldsymbol{\theta}}) \\ \hat{\boldsymbol{b}}_{\boldsymbol{w}} + \tilde{\boldsymbol{b}_{\boldsymbol{w}}} \end{pmatrix} \xrightarrow{linearize} \begin{pmatrix} \hat{\boldsymbol{q}} * (1, 1/2\ \delta\boldsymbol{\theta}^T)^T \\ \hat{\boldsymbol{b}}_{\boldsymbol{w}} + \delta\boldsymbol{b}_{\boldsymbol{w}} \end{pmatrix} \tag{4.12}$$

where $\tilde{\boldsymbol{\theta}}$ is the linearized attitude error and $\tilde{\boldsymbol{b}}_{\boldsymbol{w}}$ is the linearized bias error.

Defining the continuous error state model $\dot{\tilde{\boldsymbol{x}}}$ in 4.3 using (4.12) and the first order linearization gives:

$$\dot{\tilde{\boldsymbol{x}}} \xrightarrow{\text{Linearize}} \boldsymbol{\delta\dot{x}} = F\boldsymbol{\delta x} + G_w\delta\boldsymbol{n}_w \tag{4.13}$$

where F and G_w are the Jacobian matrices of the process model with respect to the error state and the process noise. The linearized measurement noise vector $\delta n_w = [\boldsymbol{\nu}_{\omega_m}\ \boldsymbol{\nu}_{b_\omega}]^T$ consists of the first order linearizations of the process noises.

The F and G_w matrices corresponding to this model are given as:

$$F = \begin{pmatrix} [-\boldsymbol{\omega}_m - \hat{\boldsymbol{b}}_\omega]_\times & I_3 \\ 0_3 & 0_3 \end{pmatrix}$$

$$G_w = \begin{pmatrix} I_3 & 0_3 \\ 0_3 & I_3 \end{pmatrix}$$

To simplify the matrices, I_i is a $i \times i$ identity matrix, 0_i is a zero matrix with size $i \times i$ and $[\cdot]_\times$ denotes a skew symmetric matrix operator.

The linearized error state measurement model is formulated as:

$$\tilde{\boldsymbol{y}} \xrightarrow{\text{Linearize}} \delta\boldsymbol{y} = \boldsymbol{H}\delta\boldsymbol{x} + \boldsymbol{G}_\nu\delta\boldsymbol{n}_\nu \tag{4.14}$$

where H and G_ν are the Jacobian of the measurement model with respect to the error state and the measurement noise. The linearized measurement noise vector

$\delta \boldsymbol{n_\nu} = [\boldsymbol{\nu_b}\ \boldsymbol{\nu_{fine}}\ \boldsymbol{\nu_{coarse}}]^T$ is the first order linearizations of the measurement noises. The H and G_ν are defined as:

$$H = \begin{pmatrix} [\hat{R_q}^T \boldsymbol{b}_e]_\times & 0_3 \\ [\hat{R_q}^T \boldsymbol{s}_{ref}]_\times & 0_3 \\ [\hat{R_q}^T \boldsymbol{s}_{ref}]_\times & 0_3 \end{pmatrix}$$

$$G_\nu = \begin{pmatrix} I_3 & 0_3 & 0_3 \\ 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & I_3 \end{pmatrix}$$

where $\hat{R_q}^T$ is the rotation matrix parameterized using the estimated quaternion $\hat{\boldsymbol{q}}$ which expresses the rotation matrix from the {ECI} frame to the body frame {B}.

4.2 Observability

Observability in the sense of control systems, refers to the ability to observe your state variables under specific conditions using the inputs $\boldsymbol{u}$ and outputs $\boldsymbol{y}$ of the system. Given the mathematical model and set of inputs and measurements, we solve all the current trajectory states using the given sensor readings. If the observability matrix is not full rank, it indicates that certain state variables are unobservable and cannot be properly updated in the system. For a non-linear system such as the satellite attitude determination system, the non-linear observability matrix $\mathcal{O}$ can be expressed as:

$$\mathcal{O} = \left\{ \nabla \mathcal{L}^l{}_{fa,\dots fb}\ h(x) | a, b = 0 \dots k; l \in \mathcal{N} \right\}$$

4.2.1 Observability Analysis

To determine the observability of the satellite during the eclipse, the gradient of the measurement model can be expressed as:

$$h\left(x\right) = \begin{bmatrix} \left[\hat{R}_q^T \boldsymbol{b_e}\right]_\times & 0_3 \end{bmatrix}$$

During the eclipse, both sun sensors will produce null signals. They are not accounted for in the measurement model leaving only the magnetometer data as the sensor readings available for attitude determination. With this model, a null space matrix N is computed as:

$$N_1 = \begin{bmatrix} R_q^T \boldsymbol{b_e} \\ 0 \end{bmatrix}$$

with our states as:

$$\widetilde{\boldsymbol{x}} = \begin{bmatrix} \widetilde{\boldsymbol{\theta}} \\ \widetilde{\boldsymbol{b}}_{\boldsymbol{w}} \end{bmatrix}$$

Any rotation around the magnetic field vector will result in an unobservable state. To address this, observability constraints can be implemented. To ensure the observability, the following conditions must be met:

$$N_{(k+1)} = \Phi_{(k+1|k)} N_k \quad \& \quad HN_k = 0$$

where Φ denotes the state transition matrix from k to $k+1$. With the current model, the $HN = 0$ condition is only met when:

$$\left[\hat{R}_q^T \boldsymbol{b_e}\right]_\times R_q^T \boldsymbol{b_e} = \hat{R}_q^T \left[\boldsymbol{b_e}\right]_\times \hat{R}_q^T R_q^T \boldsymbol{b_e} = 0$$

but,

$$\hat{R}_q^T R_q^T \;=\; \delta R$$

This means that the condition is only satisfied if the error between our estimated and actual quaternion is 0. This is not possible to achieve with the current model, where we rely on our estimated quaternion to be perfect. The left quaternion error parameterization used for the model is switched to the right error model to address this issue. This expresses our attitude error variables in the {ECI} frame as opposed to the body frame. The error definition update is shown below:

$$Left : \boldsymbol{q} = \hat{\boldsymbol{q}} * Exp_{S3}\left(\widetilde{\boldsymbol{\theta}}\right)$$

$$\rightarrow Right : \boldsymbol{q} = Exp_{S3}\left(\widetilde{\boldsymbol{\theta}}\right) * \hat{\boldsymbol{q}}$$

where exp represents the exponential mapping of the estimated rotation error from the lie group *SO(3)* to S^3 [123], which now gives the H and null matrix as:

$$H = [[\boldsymbol{b}'_{\boldsymbol{e}}]_x \quad 0] \ \& \ N \ = \begin{bmatrix} \boldsymbol{b_e} \\ \\ 0 \end{bmatrix}$$

Now to satisfy the $HN = 0$ condition, we only rely on our measurement magnetic field to be equal to the actual magnetic field, $\boldsymbol{b}'_{\boldsymbol{e}} \approx \boldsymbol{b_e}$. This can reasonably be achieved using the orbit data and does not depend on the attitude state of the filter. To implement the right error parameterization change in the attitude estimator, all F, G & H matrices are required to be modified to express in the {ECI} frame. This gives:

$$F = \begin{pmatrix} 0_3 & -R_q \\ \\ 0_3 & 0_3 \end{pmatrix}$$

$$G = \begin{pmatrix} -R_q & 0_3 \\ \\ 0_3 & I_3 \end{pmatrix}$$

$$H = \begin{pmatrix} [\boldsymbol{b}_e]_\times & 0_3 \\ [\boldsymbol{s}_{ref}]_\times & 0_3 \\ [\boldsymbol{s}_{ref}]_\times & 0_3 \end{pmatrix}$$

In summary the algorithm for the proposed attitude estimator is as follows:

Right $\mathbb{S}_3$ EKF Attitude Estimator

Model :

$$\begin{aligned}\dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w})\\ \begin{pmatrix}\dot{\boldsymbol{q}}\\ \dot{\boldsymbol{b}}_\omega\end{pmatrix} &= \begin{pmatrix}\frac{1}{2}\boldsymbol{q}(\boldsymbol{\omega}_m - \boldsymbol{b}_\omega + \boldsymbol{\nu}_{\omega_m})\\ \boldsymbol{\nu}_{b_\omega}\end{pmatrix}\\ \boldsymbol{y} &= h(\boldsymbol{x},\boldsymbol{\nu})\\ \begin{pmatrix}\boldsymbol{y}_b\\ \boldsymbol{y}_s\end{pmatrix} &= \begin{pmatrix}R^T\boldsymbol{b}_e + \boldsymbol{\nu}_b\\ R^T\boldsymbol{s}_{ref} + \boldsymbol{\nu}_s\end{pmatrix}\end{aligned}$$

Error model :

$$\tilde{\boldsymbol{x}} = \boldsymbol{x}\ominus\hat{\boldsymbol{x}} = (\tilde{\boldsymbol{\theta}},\tilde{\boldsymbol{b}}_\omega) \qquad where,$$

$$\begin{pmatrix}\tilde{\boldsymbol{\theta}}\\ \tilde{\boldsymbol{b}}_\omega\end{pmatrix} = \begin{pmatrix}Log_{S3}(\boldsymbol{q}*\hat{\boldsymbol{q}}^{-1})\\ \boldsymbol{b}_\omega - \hat{\boldsymbol{b}}_\omega\end{pmatrix} \approx \begin{pmatrix}\delta\boldsymbol{\theta}\\ \delta\boldsymbol{b}_\omega\end{pmatrix} \rightarrow \delta\boldsymbol{x} = \begin{pmatrix}\delta\boldsymbol{\theta}\\ \delta\boldsymbol{b}_\omega\end{pmatrix} \rightarrow \dot{\boldsymbol{\delta x}} = F\boldsymbol{\delta x} + G_w\boldsymbol{w}$$

$$\tilde{\boldsymbol{y}} = \boldsymbol{y}\ominus\hat{\boldsymbol{y}} = \begin{pmatrix}\boldsymbol{y}_b - \hat{R}^T\boldsymbol{b}_e\\ \boldsymbol{y}_s - \hat{R}^T\boldsymbol{s}_{ref}\end{pmatrix} \qquad \rightarrow \delta\boldsymbol{y} = \begin{pmatrix}\delta\boldsymbol{y}_b\\ \delta\boldsymbol{y}_s\end{pmatrix} \rightarrow \boldsymbol{\delta y} = H\boldsymbol{\delta x} + G_\nu\boldsymbol{\nu}$$

Initialization :

$$\hat{\boldsymbol{x}} = \begin{pmatrix}\boldsymbol{q}_0, \boldsymbol{0}\end{pmatrix} \quad P = 0.1I_6 \quad Q_w = \mathrm{E}(\boldsymbol{w}\boldsymbol{w}^\mathsf{T}) \quad R_\nu = \mathrm{E}(\boldsymbol{\nu}\boldsymbol{\nu}^\mathsf{T})$$

Filter Equations :

$$\hat{\boldsymbol{x}}_{k+1|k} = \int_{kT}^{(k+1)T} f(\hat{\boldsymbol{x}}_k,\boldsymbol{u}_k)dt$$

$$\begin{pmatrix}\hat{\boldsymbol{q}}_{k+1|k}\\ \hat{\boldsymbol{b}}_{\omega_{k+1|k}}\end{pmatrix} = \int_{kT}^{(k+1)T} \begin{pmatrix}0.5\hat{\boldsymbol{q}}_k(\boldsymbol{\omega}_{m_k} - \hat{\boldsymbol{b}}_{\omega_k})\\ \boldsymbol{0}\end{pmatrix} dt$$

$$P = \Phi P\Phi^T + Q_d$$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1|k}\oplus K\tilde{\boldsymbol{y}}$$

$$\begin{pmatrix}\hat{\boldsymbol{q}}_{k+1}\\ \hat{\boldsymbol{b}}_{\omega_{k+1}}\end{pmatrix} = \begin{pmatrix}\mathrm{Exp}_{S3}(K_q\tilde{\boldsymbol{y}})*\hat{\boldsymbol{q}}_{k+1|k}\\ \hat{\boldsymbol{b}}_{\omega_{k+1|k}} + K_{b_\omega}\tilde{\boldsymbol{y}}\end{pmatrix} \leftarrow K = \begin{pmatrix}K_q\\ K_{b_\omega}\end{pmatrix}$$

$$S = HPH^T + R_d$$

$$K = PH^TS^{-1}$$

$$P = (I-KH)P(I-KH)^T + KR_dK^TP$$

Filtering Matrices :

$$F = \begin{pmatrix}\boldsymbol{0}_3 & -R\\ \boldsymbol{0}_3 & \boldsymbol{0}_3\end{pmatrix} \qquad H = \begin{pmatrix}[\boldsymbol{b}_e]_\times & \boldsymbol{0}_3\\ [\boldsymbol{s}_{ref}]_\times & \boldsymbol{0}_3\end{pmatrix}$$

$$G_w = \begin{pmatrix}-R & \boldsymbol{0}_3\\ \boldsymbol{0}_3 & I_3\end{pmatrix} \qquad G_\nu = \begin{pmatrix}I_3 & \boldsymbol{0}_3\\ \boldsymbol{0}_3 & I_3\end{pmatrix}$$

$$\Phi = e^{FT} \approx I + FT + \frac{1}{2}F^2T^2 \quad G_d = \int_0^T e^{F(T-\lambda)}G_w\, d\lambda \approx G_wT$$

$$Q_d = G_dQ_wG_d^T \qquad R_d = G_\nu R_\nu G_\nu^T$$

Note that this algorithm is different from the usual left error parameterization used by state-of-the-art attitude estimation algorithms (MEKF) for satellites [117], i.e., $\tilde{\boldsymbol{q}} = \hat{\boldsymbol{q}}^{-1} * \boldsymbol{q} = Exp_{S3}(\tilde{\boldsymbol{\theta}})$. This thesis evaluates the improved performance achievable by using the right error parameterization EKF algorithm presented above in terms of estimating the correct confidence bounds for satellite attitude estimation.

4.3 Simulation Testing of the Right Error Parameterization

4.3.1 Simulation Environment

The first step for testing the proposed right quaternion $\mathbb{S}_3$ error parameterization is to develop a simulation environment to simulate both the proper orbit and sensors for the satellite. MATLAB has been used to simulate the space environment for its ease of access for simulations and the authors' familiarity with the software. Both the sun and Earth are simulated in the model with their proper sizes and distances apart to generate sun sensor results. To simplify the environment, the sun and Earth are configured as fixed points in the simulation. The simulation only focuses on a small number of orbits of the satellite around Earth lasting less than an entire day, this simplification has no impact on the performance of the simulation. To simulate the satellite trajectory around Earth, a high precision orbit propagator is used to generate an orbital trajectory for the satellite orbiting earth [153]. The propagator parameters are configured to mimic the International Space Station's orbit around the planet, which is the standard orbit that any CubeSat satellite would follow, and

is the expected orbit of the Killick-1 CubeSat. The magnetometer is simulated using the International Geomagnetic Reference Field (IGRF) model. This model uses a specified time and orbital position in latitude, longitude, & altitude to return the magnetic field vector in the {ECEF} frame. A conversion function is used to convert the magnetic field vector to the {ECI} frame and further to the body frame of the satellite, resulting in the simulation of an actual magnetometer reading for a satellite. The sun sensors are simulated based on each of the sensors' orientation with respect to the sun. To determine if any of the sensors are out of the field of view, each sensor reference angle to the sun position is computed. Additionally, the sun and Earth's position are taken into account to determine if the satellite is in the eclipse or not. Each sensor reading is simulated with Gaussian noise and drift in the case of the gyroscope. To best analyze the performance of the new $\mathbb{S}_3$ error parameterization, a Monte Carlo simulation is used to evaluate the left versus right quaternion algorithm.

4.3.2 Performance of the Right Error EKF

The simulation is run with the same seeded random number generation and same noise figures for both the error parameterizations to evaluate the right quaternion error parameterization performance. The two primary evaluation criteria for the sensor fusion are the Root Mean Squared Error (RMSE) and Normalized Estimation Error Squared (NEES). RMSE is defined as:

$$\text{RMSE} = \sqrt{\sum_{i=1}^{n} \frac{(\boldsymbol{y_i} \ominus \hat{\boldsymbol{y}}_{\boldsymbol{i}})^2}{n}} \tag{4.15}$$

NEES is defined as:

$$\text{NEES}_k = (\boldsymbol{y}_k \ominus \hat{\boldsymbol{y}}_k) P_{k|k}^{-1} (\boldsymbol{y}_k \ominus \hat{\boldsymbol{y}}_k) \tag{4.16}$$

A lower RMSE value signifies accurate estimation while an ideal NEES value of the degrees of freedom of the variable signifies consistent estimation, i.e. the predicted confidence bounds correctly bounds the actual estimation errors. That is an ideal NEES value of 3 for both the $NEES\ ang$ and $NEES\ b_w$. To compute these metrics, a Monte Carlo simulation is conducted for ten iterations. The RMSE and NEES of both models are displayed in Figures 4.3 and 4.4. It is important to note that the simulator used the same orbit trajectory while a Monte Carlo simulation was conducted. In this orbit, the satellite enters the eclipse at 650 seconds and remains in the eclipse until 2500 seconds. The eclipse region is highlighted in grey. From the figure, a significant spike is observed in the left quaternion error parameterization during the eclipse in both plots. From the RMSE plot, at the start of the eclipse, both error parameterizations follow a similar expected increase in the error of the estimated orientation. The longer the satellite is in the eclipse, an increase in the error is seen from the left parameterization. In contrast, the right error parameterization remains below its counterpart. There is a slight unnoticeable variance between both parameterizations on the gyroscope bias error side, where this variable is unaffected by the reduction of sensors. In analyzing the NEES comparison, a significant impact is observed. While the right quaternion parameterization stays consistent throughout the entire orbit holding a value around 3, the left error parameterization NEES consistently grows during each eclipse. This is the result of spurious updates of the unobservable states during the eclipse period. Once the satellite regains vision of the sun, the NEES returns to its correct values. The simulation results are also expanded to simulate five orbits to demonstrate the consistency of the estimates achievable by the right error parameterization when compared with the left error parameterization

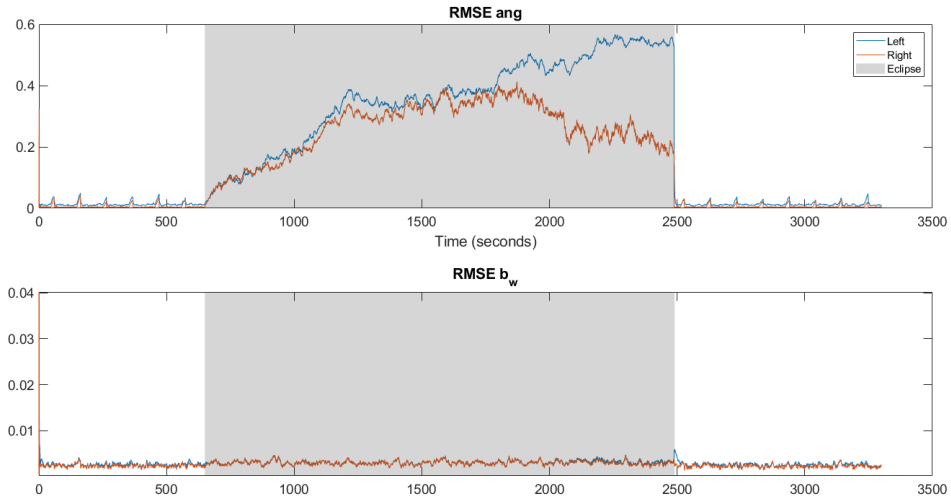


Figure 4.3: The left and right quaternion error parameterization RMSE performance for one full orbit simulation.

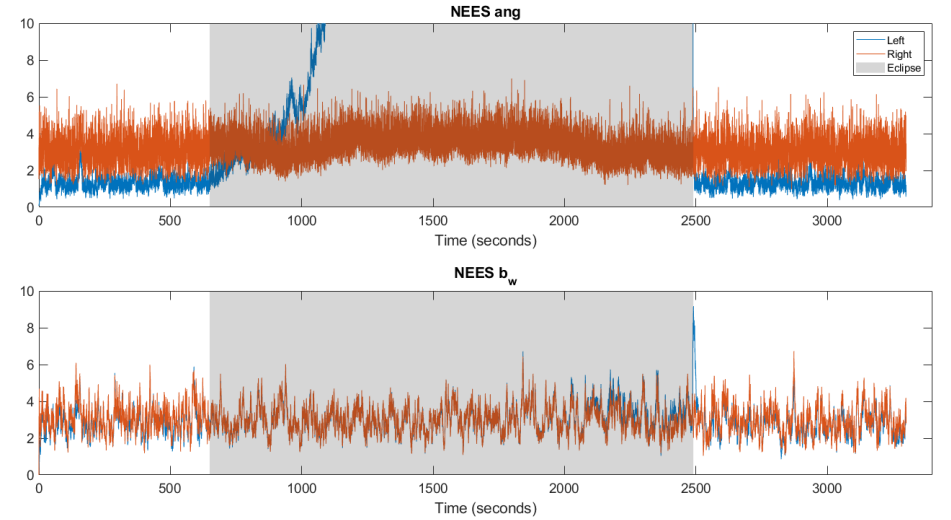


Figure 4.4: The left and right quaternion error parameterization NEES performance for one full orbit simulation.

for attitude estimation. The RMSE and NEES plots are seen in Fig. 4.5 and Fig. 4.6. The results are observed for the five orbits, with the left parameterization resulting in larger spikes during each eclipse period. The right error parameterization maintains a lower RMSE for angle estimation of on average 0.0981 lower than the conventional left parameterization during the eclipse period. In addition, the right parameterization maintains a NEES value closer to the ideal value of 3 due to its observability consistency property.

4.4 Physical Testing

4.4.1 Sensor Board

As part of the thesis, physical testing of the right error parameterization was performed on hardware to validate the filter for experimental implementation. To conduct physical testing, a sensor suite was built to incorporate all sensors in the same board. The original board design was built by Steven Zhang, an undergraduate student on the Killick-1 project. The sensor suite was designed to include a three-axis ADXRS453 gyroscope, a three-axis BMM150 magnetometer, and a three-axis ADXL354 accelerometer that all communicate to a raspberry pi zero via an SPI interface. The sensor suite has been modified to communicate with the magnetometer over I2C and use a raspberry pi zero W instead of the previous design. The usage of the raspberry pi zero W allows for wireless communication to the sensor suite where the previous board required an umbilical to communicate with any computer. In addition, a fine sun sensor is simulated in the sensor suite using a raspberry pi camera

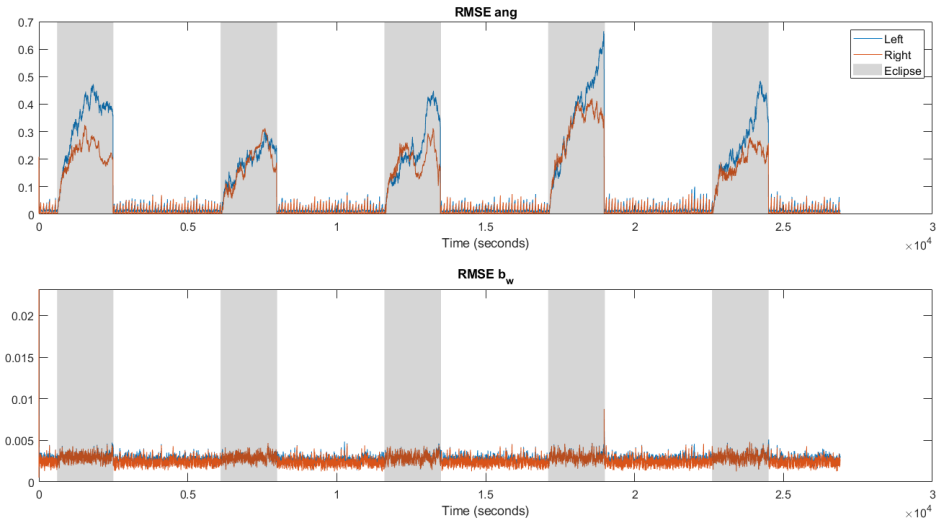


Figure 4.5: The left and right quaternion error parameterization RMSE performance for five full orbits simulation.

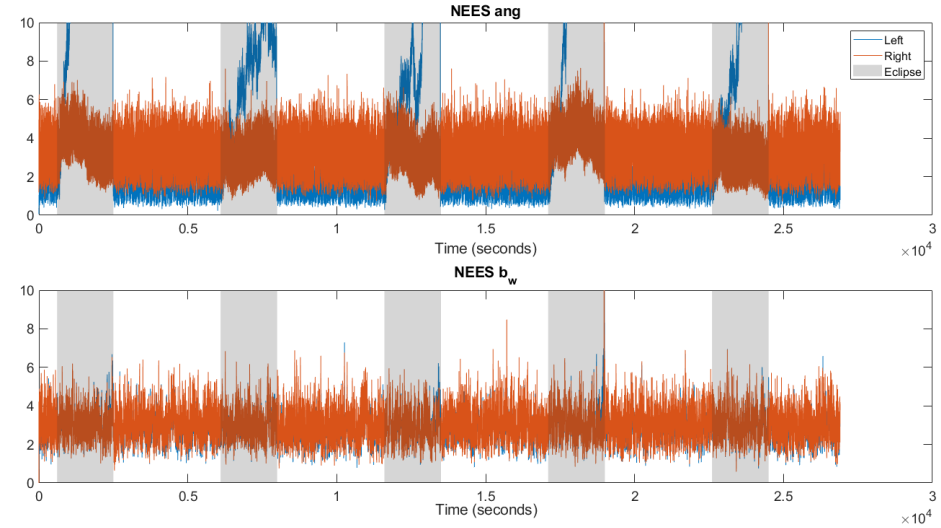


Figure 4.6: The left and right quaternion error parameterization NEES performance for five full orbits simulation.

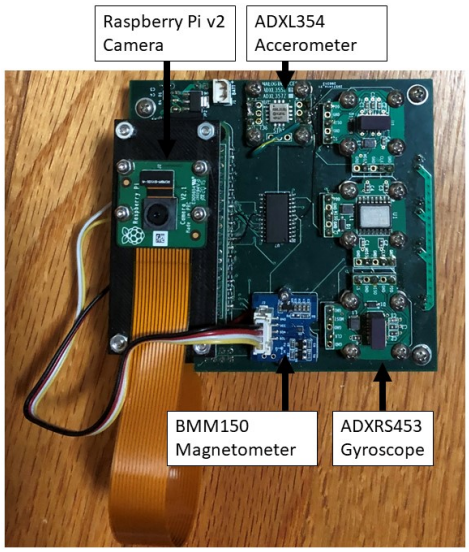


Figure 4.7: Sensor suite testing board

module V2 mounted on top of the raspberry pi with a 3D printed mounting board. The sensor suite is seen in Figure 4.7. The board can be powered remotely via a battery pack connected to the sensor board or a portable USB charger connected to the raspberry pi power micro USB header. The camera attached to the board is used to simulate a fine sun sensor used for ADCS on satellites. The camera algorithm is designed to detect the sun. The sun is simulated using a flashlight with a white cover over the lens to reduce the glare. The algorithm uses basic thresholding to filter out anything excluding the flashlight in a dark room and detects the centroid of the sun. The sun vector is then determined using the following algorithm in Figure 4.8: The magnetometer was calibrated within the lab environment using a figure-eight technique to record a dataset used to calibrate the magnetometer's coefficients. Since the sensors communicate with three different protocols, the sensors on the board are threaded to allow for the fastest sampling time possible for each sensor. The times-

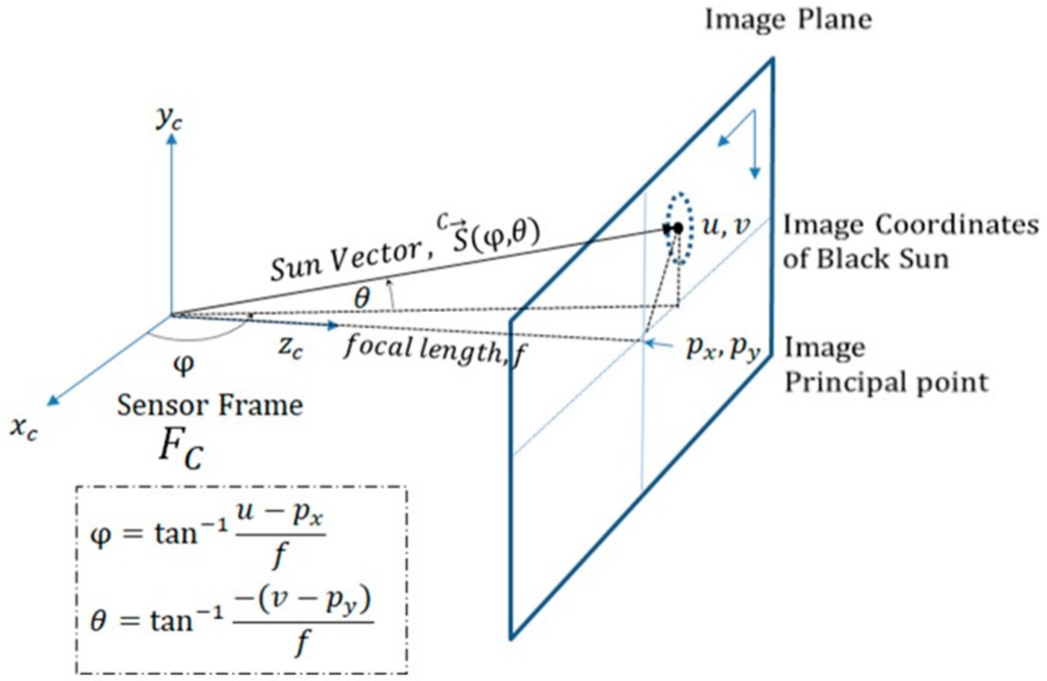


Figure 4.8: The sun sensor calculation technique

tamps are recorded for each iteration to allow for easy syncing of the data. With the threading, the magnetometer, accelerometer, and gyroscope achieve a maximum frequency of 30 Hz. The sun sensor achieves a maximum frequency of 0.6123 Hz in the current implementation.

4.4.2 Physical Testing Environment

The physical testing was conducted at the Intelligent Systems Lab at Memorial University. The sensor board detailed in Section 4.4.1 was fixed to a rotary table in the lab and powered via a USB charger. All sensor data was recorded for each test run and tested without lights on to ensure the sun sensor algorithm would detect the flashlight mounted in the room. A motion capture system was used to record the

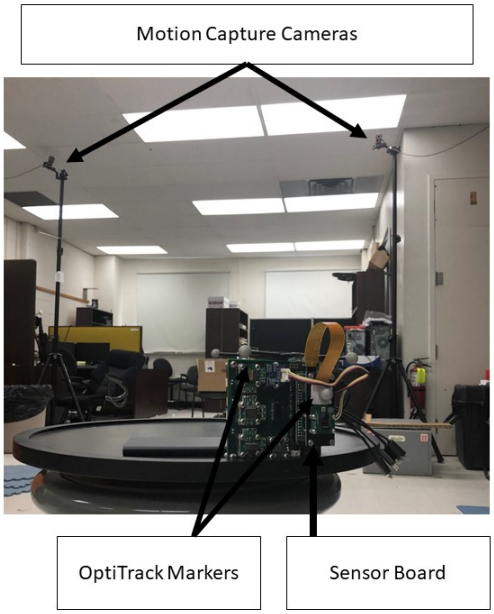


Figure 4.9: The lab test environment at Memorial University

ground truth of the sensor suite with motion capture markers mounted to multiple points of the sensor board. The lab environment is seen below in Figure 4.9. The testing consisted of rotating the rotary table around and holding it at stationary points to assist in the data alignment between the ground truth and sensor recordings.

4.4.3 Physical Testing Results

For the physical testing, the sensor board was run for nine minutes, with the sensor board held stationary for the initial minute to adjust and tune the Kalman filter. The rotary table was rotated to a new fixed position and held motionless for the remainder of the thirty seconds every thirty seconds.

In Figure 4.10 the RMSE of the data is presented. The right error parameterization was able to achieve accurate attitude estimation performance comparable to that of

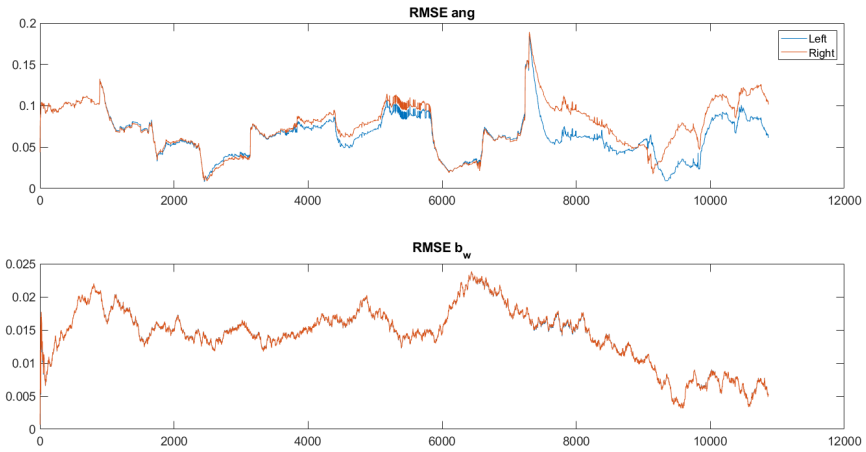


Figure 4.10: The left and right quaternion parameterization RMSE performance for the nine minute lab physical testing.

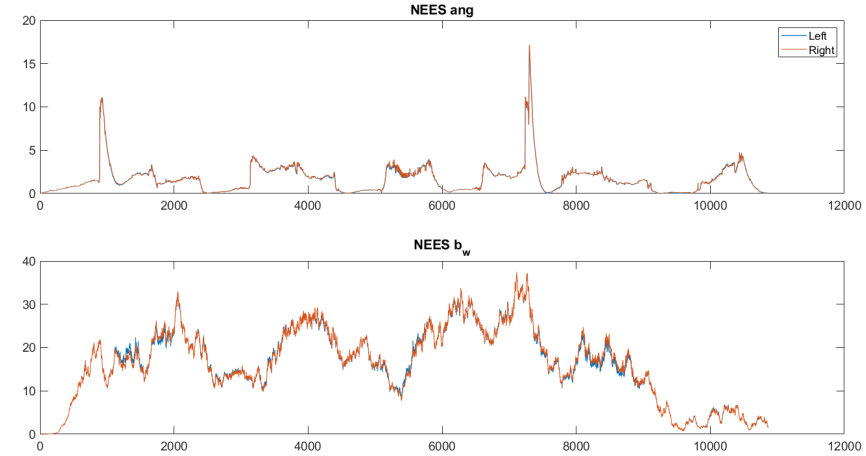


Figure 4.11: The left and right quaternion parameterization NEES performance for the nine minute lab physical testing.

the left version of the filter.

There is no major performance difference between the left versus right error quaternion estimation for the trajectories considered in the current experimental study. Figure 4.11 presents the NEES for both error parameterizations. In both the angular and gyroscope bias plots, there is negligible variation in the performance of the parameterizations. The physical testing did not demonstrate the same trends from the simulation environment. However, it did demonstrate that the right quaternion performance is comparable to the left quaternion performance for actual experimental data. The lack of error propagation from the left quaternion parameterization in the simulation can be attributed to the duration of the eclipse period in the lab testing. The complete test set lasts for nine minutes in the lab testing with a 2-3 minute maximum period where the sun is unobservable. In the simulation environment, the eclipse period lasts for approximately 45 minutes which is the eclipse period of a CubeSat in an ISS orbit. This short eclipse period will not result in the drift as seen in the simulation. When looking at the start of the eclipse, the left and right quaternion error parameterizations have comparable performances until a portion of the eclipse has passed. Due to hardware limitations, the sensor board was unable to collect a test set for a full orbital period and was limited to the 9-minute test set. If the set lasted for a longer period, it is expected that the same trend as seen from the simulation would occur. While this experimental test validates the accurate estimation of the right error parameterization for attitude filtering, experimental validation of the improved NEES performance during eclipse periods was not completed. In order to properly demonstrate the NEES performance, further improvements to the experimental testbed are necessary.

4.5 Fault Detection Strategy

For fault detection, the confidence bounds of the EKF filter are used to determine if the sensor measurements are outside of the bounds. This is accomplished using Mahalanobis distance squared, which is the measure between a sample point and a distribution of points. The algorithm is as follows:

$$
\begin{gathered}
S = HPH' + R_\nu \\
NIS = e' S^{-1} e \\
if(NIS > \chi^2(.95)) \quad Fault = true \\
then: \quad skip\ EKF\ correction\ update \\
record\ \ sensor\ data \\
else: \\
\begin{pmatrix} \hat{\boldsymbol{q}}_{k+1} \\ \hat{\boldsymbol{b}}_{\omega_{k+1}} \end{pmatrix} = \begin{pmatrix} Exp_{S3}(K_q \tilde{\boldsymbol{y}}) * \hat{\boldsymbol{q}}_{k+1|k} \\ \hat{\boldsymbol{b}}_{\omega_{k+1|k}} + K_{b_\omega} \tilde{\boldsymbol{y}} \end{pmatrix} \leftarrow K = \begin{pmatrix} K_q \\ K_{b_\omega} \end{pmatrix} \\
P = (I - KH)P(I - KH)^T + KR_dK^TP
\end{gathered}
\tag{4.17}
$$

where P is the covariance matrix of the EKF filter, R_ν is the measurement noise matrix, NIS is the normalized innovation squared, and e is the error between the measured sensor data and the reference data.

A Chi-Square test is used to determine if the normalized innovation squared is greater than the Chi-Square value $\chi^2(.95)$. If the normalized innovation squared value exceeds 95% of the distribution samples for the degrees of freedom of the error bounds, the measurement is declared a fault instance. To improve the stability of the fault detection, multiple fault instances need to be detected in a row for a fault to be flagged. The overall process of the fault detection strategy is presented in Figure

4.12. The process starts with the standard collection of sensor data for the EKF filter update. Next, the fault detection algorithm highlighted above runs to determine if any possible faults occur. If no faults are flagged, the satellite attitude determination algorithm runs as expected and updates the EKF filter matrices. If a fault is detected, the EKF sensor update is skipped, and the logging of the sensor data starts. This is referred to as the start of the fault gating procedure. The fault instances are continuously recorded to determine the duration of the fault. To improve the fault gating detection, a sliding average of the fault instances is recorded to ensure no dropouts of the fault gating occur. If the fault lasts for less than two orbits, the fault is not flagged as a persistent fault, and the logged sensor data is cleared. If the fault lasts longer than two orbits, the fault is deemed to be a persistent fault and requires additional analysis. The fault duration period of two orbits is chosen due to the nature of magnetic disturbances. Any magnetorquer actuation or communication downlink would only occur for a brief period of an orbit. Thus, this accommodates any common disturbances during typical satellite operations. The recorded sensor data is then downlinked to Earth for analysis. The recorded sensor data is processed with non-linear $\mathbb{S}^3$ manifold optimization to determine the magnetic bias or if any further fault correction is required.

4.6 Sliding Window Filtering for Fault Recovery

In the case of a persistent fault, a non-linear sliding window filter is implemented to recover the unknown bias encountered in the sensor. This process uses a machine learning regression algorithm to best fit the data. Traditionally the process requires

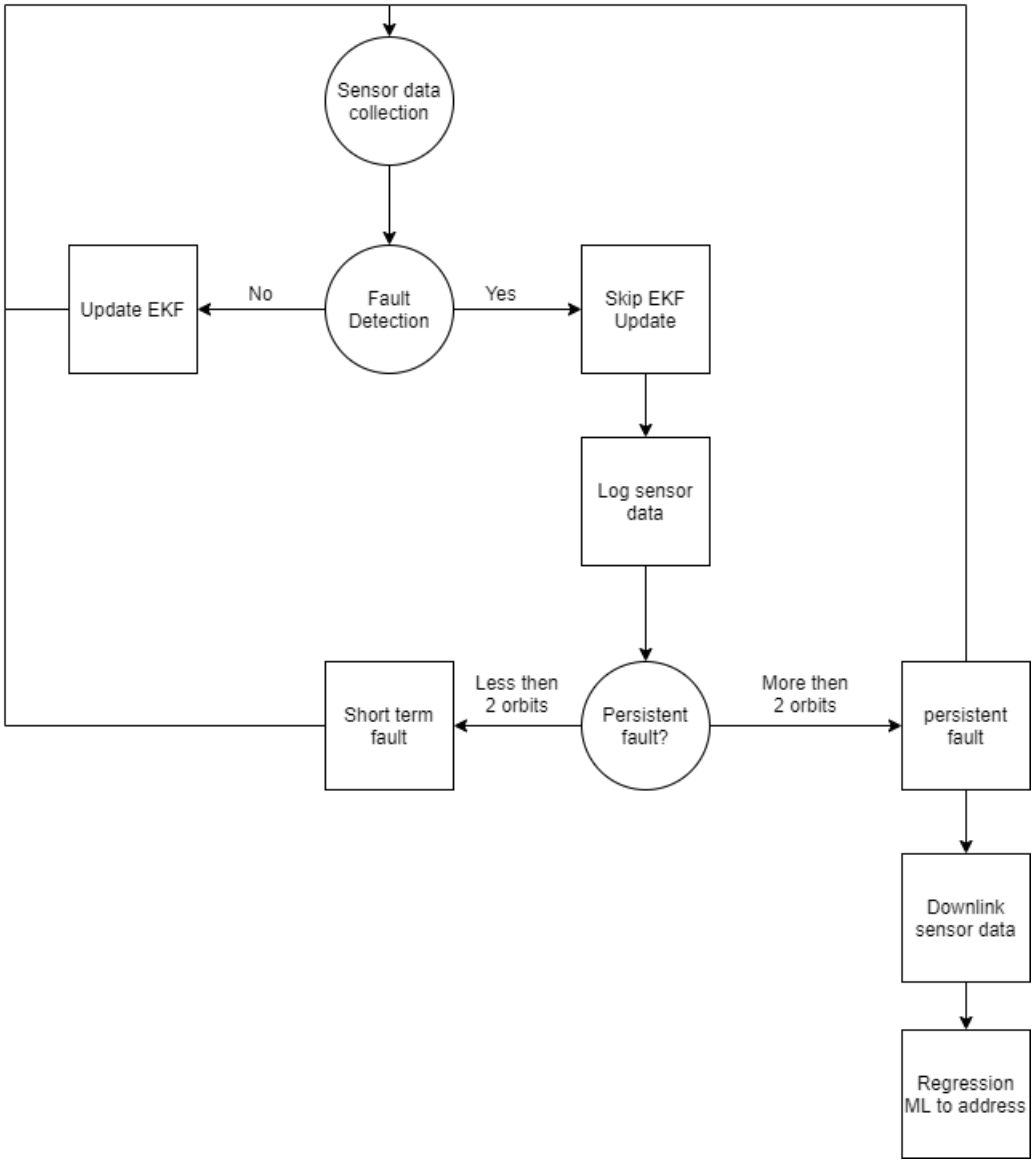


Figure 4.12: Fault detection process flow diagram

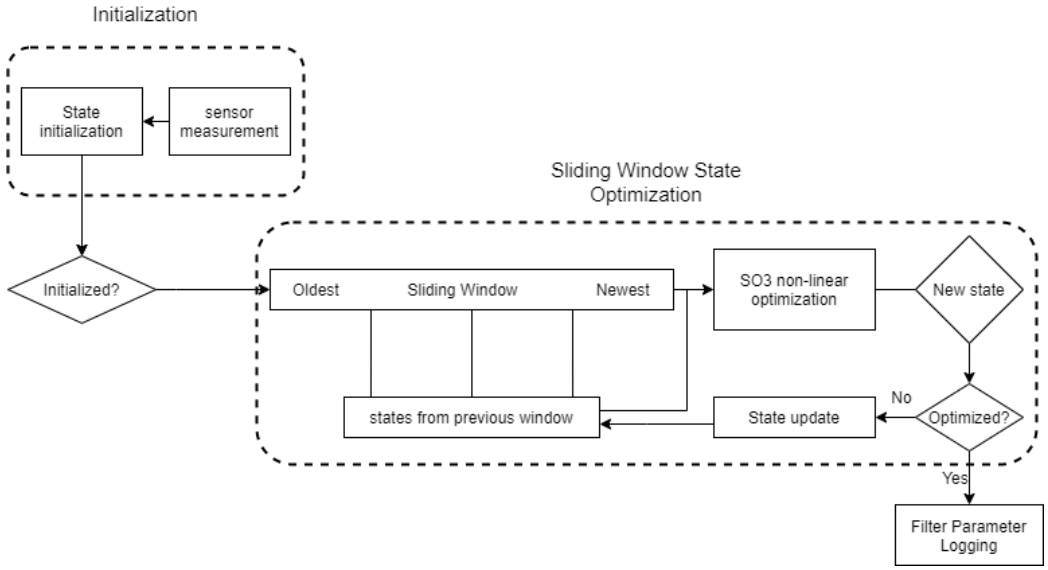
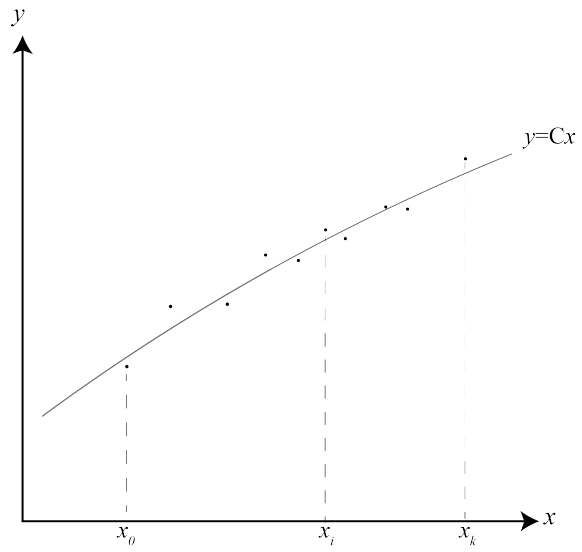


Figure 4.13: A flow diagram of the sliding window filter using regression machine learning to estimate the magnetic bias state.

magnetometer data from specific 3D calibration trajectories. However, when several sensor data sources are available, the magnetic bias can be estimated as shown in [154]. This recalibration can be thought of as a batch least squares regression problem using field data which is often tackled using machine learning optimization libraries. Regression in machine learning uses a cost function minimization to optimize a set of parameters e.g. $\boldsymbol{x}$ to best fit a dataset e.g. $\boldsymbol{y}$. An example regression problem is seen in Figure 4.14, where e_i is the error and ν_i is the Gaussian noise with covariance R.

A cost function for the regression problem can be defined as follows:

$$J(\mathrm{X}) = \frac{1}{2}\sum_{k=0}^{K}(\boldsymbol{y}_k - C\boldsymbol{x}_k)^T R^{-1}(\boldsymbol{y}_k - C\boldsymbol{x}_k) \tag{4.18}$$



$$e_i(x) = y_i - Cx_i \pm \nu_i$$

Figure 4.14: A sample regression problem to demonstrate how the model is best fit.

where we find the combined state vector (which has states of all time steps) $\mathrm{X} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_K \end{bmatrix}$ that minimizes $J(\mathrm{X})$ given $\mathbf{y}_k \; \forall \; k = 0...K$ such that it follows a model $\mathbf{y}_k = C\mathbf{x}_k + \boldsymbol{\nu}_k$ where $\boldsymbol{\nu}_k = \mathcal{N}(0, R_k)$. I.e.

$$\hat{\mathbf{X}} = \underset{X}{\operatorname{argmin}} \frac{1}{2} \sum_{k=0}^{K} (\mathbf{y}_k - C\mathbf{x}_k)^T R^{-1} (\mathbf{y}_k - C\mathbf{x}_k)$$

In machine learning, this problem is solved using an optimization library, which takes the cost function and numerically minimizes it against the set of parameters $\boldsymbol{X}$, to provide the optimum $\hat{\boldsymbol{X}}$ that minimizes the cost. The optimization method varies on the application. Common methods used for regression include Gradient Descent [155], Gauss-Newton [156], Levenberg-Marquardt [157], and trust region convergence methods [158]. The choice of method dictates the convergence speed, solution quality,

and numerical stability.

There are three important elements to consider when solving these optimization problems. The first is to provide a good initial condition for the unknown parameters $\boldsymbol{X}$. Second is to provide analytical Jacobians for the cost function as it improves the convergence to the correct solution more stable and probable. Lastly, the learning rate and learning direction should be provided for a fast and robust convergence.

4.6.1 Initialization of Optimization Parameters

The initialization depends on the type of problem. In some problems, you can find a solution for the states using measurements, e.g., solving the Whaba's problem for satellite attitude determination. In some cases, the system model can be used to predict the future states with enough reliability starting from an initial state, e.g. gyroscope integration in short periods for attitude determination. Furthermore, assumptions can be made for states like bias states as they remain mostly constant. Depending on the application, a combination of these methods could be used.

4.6.2 Use of Analytical Jacobians

To figure out the analytical Jacobians, the following formulation is used for ease of implementation.

$$\hat{\boldsymbol{X}} = \underset{X}{\operatorname{argmin}} \frac{1}{2} \sum_{k=0}^{K} (\boldsymbol{y}_k - C\boldsymbol{x}_k)^T R^{-1} (\boldsymbol{y}_k - C\boldsymbol{x}_k)$$

define: $\boldsymbol{e}_k = \boldsymbol{y}_k - C\boldsymbol{x}_k$ $\quad \boldsymbol{e} = \begin{bmatrix} \boldsymbol{e}_0 \\ \vdots \\ \boldsymbol{e}_k \\ \vdots \\ \boldsymbol{e}_K \end{bmatrix}$ $\quad W = \text{diag}(R_0, ...R_k, ...R_K)$ $\quad H_k = \frac{-\partial \boldsymbol{e}_k}{\partial \boldsymbol{x}_k}|_{\boldsymbol{x}_K = \hat{\boldsymbol{x}}_k} = -C_k$ $\quad \mathbb{H} = \text{diag}(H_0, H_1, ...H_k, ...H_K)$

Then the optimization cost function becomes:

$$J(x) = \frac{1}{2}\boldsymbol{e}(\mathrm{X})^T W^{-1} \boldsymbol{e}(\mathrm{X})$$

with the analytical Jacobian J taking the form:

$$\frac{\partial J}{\partial \boldsymbol{x}} = -\mathbb{H}$$

Note that the combined measurement matrix $\mathbb{H}$ captures measurement models corresponding to residuals of all time steps.

4.6.3 Learning Rate and Learning Direction

For learning direction, a Gauss-Newton solution strategy can be followed with a line search parameter α for learning rate. The complete algorithm can be summarized using the following steps:

- starting with the initial condition $\hat{\boldsymbol{X}}$
- evaluate $e(\hat{\boldsymbol{X}}), \mathbb{H}(\hat{\boldsymbol{X}}), W(\hat{\boldsymbol{X}})$
- find $\delta \boldsymbol{X} = (\mathbb{H}^T W^{-1} \mathbb{H}^T)^{-1} \mathbb{H}^T W^{-1} e(\hat{\boldsymbol{X}})$
- update the current solution $\hat{\boldsymbol{X}} \rightarrow \hat{\boldsymbol{X}} + \alpha \delta \boldsymbol{X}$ where α is the learning rate.

- Repeat the process until some convergence criteria is met.

After convergence $\hat{\boldsymbol{X}} = [\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_1, ..., \hat{\boldsymbol{x}}_K]$ is the solution vector for the state. It's covariance is encoded as $(\mathbb{H}^T W^{-1} \mathbb{H})^{-1} =$

$$\begin{bmatrix} P_0 & & & & & \\ & P_1 & & & & \\ & & \ddots & & & \\ & & & P_k & & \\ & & & & \ddots & \\ & & & & & P_K \end{bmatrix}$$

where P_k is the covariance of the state estimate at time step k.

4.6.4 Regression Algorithm for State Estimation of Manifolds

While the solution above provides the answer to the problem in vector space, the magnetometer drift problem does not only exist in a vector space due to the usage of quaternions. As a result, the optimization should be carried out considering the $\mathbb{S}_3$ group parameterization where a "lift-solve-retract" strategy [159] can be followed to optimize the state variables. This change corresponds to the retraction $\oplus$ and inverse mapping $\ominus$ operation usage where applicable in the algorithm when dealing with state perturbations and error calculations related to quaternions.

given:

system model :	$\boldsymbol{x_k} = f(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, \boldsymbol{w}_k)$	$\boldsymbol{w}_k \sim \mathcal{N}(0, Q_w)$
measurement model :	$\boldsymbol{y_k} = h(\boldsymbol{x}_k, \boldsymbol{\nu}_k)$	$\boldsymbol{\nu}_k \sim \mathcal{N}(0, R_w)$
set of inputs :	$\boldsymbol{u} = (\boldsymbol{u}_1^T, \dots \boldsymbol{u}_k^T, \dots, \boldsymbol{u}_K^T)^T$	$k = 1, \dots, K$
set of measurements :	$\boldsymbol{y} = (\boldsymbol{y}_0^T, \dots \boldsymbol{y}_k^T, \dots, \boldsymbol{y}_K^T)^T$	$k = 0, \dots, K$
initial condition :	$\mathcal{N}(\boldsymbol{x}_{init,0}, P_{init,0})$	
find states :	$\boldsymbol{x} = (\boldsymbol{x}_0^T, \dots \boldsymbol{x}_k^T, \dots, \boldsymbol{x}_K^T)^T$	$k = 0, \dots, K$

by solving the optimization problem:

$$\hat{\boldsymbol{X}} = \underset{X}{\operatorname{argmin}}\, J(\boldsymbol{x})$$

where:

$$J(\boldsymbol{x}) = \sum_{k=0}^{K} (J_{u,k}(\boldsymbol{x}) + J_{y,k}(\boldsymbol{x}))$$

with cost of priors:

$$J_{u,k}(\boldsymbol{x}) \quad \tfrac{1}{2} e_{\nu,k}(\boldsymbol{x})^T W_{\nu,k}^{-1} e_{\nu,k}(\boldsymbol{x})$$

$$e_{\nu,k}(\boldsymbol{x}) \quad \begin{cases} \boldsymbol{x}_{init,0} \ominus \boldsymbol{x_0}, & k = 0 \\ f(\boldsymbol{x}_{k-1}, \boldsymbol{\nu}_k, 0) \ominus \boldsymbol{x}_k & k = 1, \dots, K \end{cases}$$

$$W_{\nu,k} \quad \begin{cases} P_{init,0} & k = 0 \\ Q_{w_k} & k = 1, \dots, K \end{cases}$$

with cost of measurements:

$$\begin{aligned} J_{y,k}(\boldsymbol{x}) &= \tfrac{1}{2} e_{y,k}(\boldsymbol{x})^T W_{y,k}^{-1} e_{y,k}(\boldsymbol{x}) \\ e_{y,k}(\boldsymbol{x}) &= y_k \ominus h(\boldsymbol{x}_k, 0) \\ W_{y,k} &\quad R_{\nu,k} \end{aligned}$$

The solution to the above problem using Gauss-Newton optimization with analytical Jacobians is:

define:

$$\delta(\boldsymbol{x}) = \begin{bmatrix} \delta\boldsymbol{x}_0 \\ \delta\boldsymbol{x}_1 \\ \vdots \\ \delta\boldsymbol{x}_k \\ \vdots \\ \delta\boldsymbol{x}_K \end{bmatrix} \qquad \mathbb{H} = \left[\begin{array}{ccccccc} 1 & 0 & & & & & \\ -F_0 & 1 & 0 & & & & \\ & -F_1 & 1 & 0 & & & \\ & & & \ddots & \ddots & 0 & \\ & & & & -F_{K-1} & 1 & \\ \hline H_0 & & & & & & \\ & H_1 & & & & & \\ & & & \ddots & & & \\ & & & & & & H_K \end{array}\right]$$

$$e(\hat{\boldsymbol{x}}) = \left[\begin{array}{c} e_{u,0}(\hat{\boldsymbol{x}}) \\ e_{u,1}(\hat{\boldsymbol{x}}) \\ \vdots \\ \hline e_{y,0}(\hat{\boldsymbol{x}}) \\ \vdots \\ e_{y,K}(\hat{\boldsymbol{x}}) \end{array}\right] \qquad \begin{aligned} W = \text{diag}(&P_{init,0}, Q_1, \ldots, Q_k, \ldots, Q_K, R_0, \ldots, R_k, \\ &\ldots, R_K) \\ Q_k &= G_{w,K} Q_w G_{w,K}^T \\ R_k &= G_{\nu,K}, R_\nu, G_{\nu,K}^T \end{aligned}$$

where:

$$F_{k-1} = \lim_{\delta x_{k-1} \to 0} \frac{f(\hat{\boldsymbol{x}}_{k-1} \oplus \delta\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, 0) \ominus f(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, 0)}{\delta x_{k-1}}$$

$$G_{w,k} = \lim_{w_k \to 0} \frac{f(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, w_k) \ominus f(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, 0)}{w_k}$$

$$H_k = \lim_{\delta x \to 0} \frac{h(\hat{\boldsymbol{x}}_k \oplus \delta\boldsymbol{x}_k, 0) \ominus h(\hat{\boldsymbol{x}}_k, 0)}{\delta\boldsymbol{x}_k}$$

$$G_{\nu,k} = \lim_{\nu_k \to 0} \frac{h(\hat{\boldsymbol{x}}_k, \boldsymbol{\nu}_k) \ominus h(\hat{\boldsymbol{x}}_k, 0)}{\boldsymbol{\nu}_k}$$

until convergence ($<$ max iterations, $\delta \boldsymbol{x}>$ tolerance)

| evaluate $H, W, \boldsymbol{e}$

| find $\delta \boldsymbol{x}^* = (H^T W^{-1} H)^{-1} H^T W^{-1} e(\hat{\boldsymbol{x}})$

| update $\hat{\boldsymbol{x}} \leftarrow \hat{\boldsymbol{x}} \oplus \alpha \delta \boldsymbol{x}^*$

| with user defined learning rate α

end

4.6.5 Regression Algorithm for Magnetic Bias Estimation

Combining all this together results in the algorithm for satellite magnetic bias estimation using orbit data. The algorithm for the calibration process is as follows:

Optimization Algorithm for Mag Bias :

Given :

$$\begin{aligned}
\boldsymbol{s_e} &= \begin{bmatrix} * * * \end{bmatrix}^T & E(\boldsymbol{x}_{0,init}, \boldsymbol{x}_{0,init}^T) &= P_{0,init} \\
\boldsymbol{b_e} &= \begin{bmatrix} * * * \end{bmatrix}^T & E(\boldsymbol{\eta_{b_w}}, \boldsymbol{\eta_{b_w}}^T) &= Q_{b_w} \\
\hat{\boldsymbol{x}}_0 &= \boldsymbol{x}_{0,init} = (\boldsymbol{q}_{0,init}, \boldsymbol{b}_{w,init}, \boldsymbol{b}_{m,init}) & E(\boldsymbol{\eta}_{b_m}, \boldsymbol{\eta}_{b_m}^T) &= Q_{b_m} \\
\boldsymbol{u_k} &= \boldsymbol{\omega}_{m,k} \qquad k = 1, \ldots, K & E(\boldsymbol{\eta}_{mag}, \boldsymbol{\eta}_{mag}^T) &= Q_{mag} \\
\boldsymbol{y_k} &= (\boldsymbol{y}_{mag,k}, \boldsymbol{y}_{sun,k}) \qquad k = 0, \ldots, K & E(\boldsymbol{\eta}_{sun}, \boldsymbol{\eta}_{sun}^T) &= Q_{sun}
\end{aligned}$$

Find : initialization for $\hat{\boldsymbol{x}}_k \quad k = 1, \ldots, K$

by $\hat{\boldsymbol{q}}_k \leftarrow$ Whaba's solution $(\boldsymbol{y}_{mag,k}, \boldsymbol{y}_{sun,k}) \forall\ k$ not in eclipse

$\hat{\boldsymbol{b}}_{w,k} \leftarrow 0 \qquad , k = 0, \ldots, K$

$\hat{\boldsymbol{b}}_{m,k} \leftarrow 0 \qquad , k = 0, \ldots, K$

$\hat{\boldsymbol{q}}_k \leftarrow 0.5\hat{\boldsymbol{q}}_{k-1} * \mathrm{Exp}_{\mathbb{S}_3}((\boldsymbol{\omega}_{m,k} - \boldsymbol{b}_{w,k-1})dt)\ \forall\ k$ in eclipse

Optimization loop :

while convergence criteria is not satisfied

Call function : Build optimization matrices$(\mathbb{H}(\hat{\boldsymbol{X}}), \mathrm{W}(\hat{\boldsymbol{X}}), e(\hat{\boldsymbol{X}}))$

$\boldsymbol{\delta X}^* = (\mathbb{H}^T W^{-1} \mathbb{H}^T)^{-1} \mathbb{H}^T W^{-1} \boldsymbol{e}$

$\hat{\boldsymbol{X}} \leftarrow \hat{\boldsymbol{X}} \oplus \alpha \delta \boldsymbol{X}^*$

end

extract $\hat{\boldsymbol{x}}_k' s$ from $\hat{\boldsymbol{X}}$

extract $\hat{P}_k' s$ from $(\mathbb{H}^T W^{-1} \mathbb{H}^T)^{-1}$

---

function *Build optimization matrices*()

Given :

$$\begin{aligned}
\boldsymbol{s_e} &= \begin{bmatrix} * * * \end{bmatrix}^T & E(\boldsymbol{\eta}_{b_w}, \boldsymbol{\eta}_{b_w}^T) &= Q_{b_w} \\
\boldsymbol{b_e} &= \begin{bmatrix} * * * \end{bmatrix}^T & E(\boldsymbol{\eta}_{b_m}, \boldsymbol{\eta}_{b_m}^T) &= Q_{b_m} \\
\hat{\boldsymbol{x}_k} &= (\hat{\boldsymbol{q}}_k, \hat{\boldsymbol{b}}_{w,k}, \hat{\boldsymbol{b}}_{m,k}) & E(\boldsymbol{\eta}_{mag}, \boldsymbol{\eta}_{mag}^T) &= Q_{mag} \\
\boldsymbol{u_k} &= \boldsymbol{\omega}_{m,k} \qquad k = 0, \ldots, K & E(\boldsymbol{\eta}_{sun}, \boldsymbol{\eta}_{sun}^T) &= Q_{sun} \\
\boldsymbol{y_k} &= (\boldsymbol{y}_{mag,k}, \boldsymbol{y}_{sun,k}) \qquad k = 0, \ldots, K \\
\boldsymbol{x}_{init,0} &= (\boldsymbol{q}_{init,0}, \boldsymbol{b}_{w,init,0}, \boldsymbol{b}_{m,init,0})
\end{aligned}$$

$$
\begin{aligned}
&\textit{initial cond} \quad \boldsymbol{x}_{init,0} = (\boldsymbol{q}_{init,0}, \boldsymbol{b}_{w,init,0}, \boldsymbol{b}_{m,init,0}) \\
&\boldsymbol{e}_{\nu,0}(\hat{\boldsymbol{x}}_0) \quad = \boldsymbol{x}_{init,0} \ominus \hat{\boldsymbol{x}}_0 \quad = (\mathrm{Log}_{\mathbb{S}_3}(\hat{\boldsymbol{q}}_0^{-1} * \boldsymbol{q}_{init,0}), (\hat{\boldsymbol{b}}_{w,init,0} - \hat{\boldsymbol{b}}_{w,0}), (\hat{\boldsymbol{b}}_{m,init,0} - \hat{\boldsymbol{b}}_{m,0})) \\
&H_{\boldsymbol{\eta},0}(\hat{\boldsymbol{x}}) \quad = I_{9\times 9} \\
&W_{\boldsymbol{\nu},0}(\hat{\boldsymbol{x}}) \quad = P_{init,0}
\end{aligned}
$$

inputs :

$$
\begin{aligned}
&\boldsymbol{e}_{\nu,k}(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{x}}_{k-1}) \quad = f(\boldsymbol{x}_{k-1}\boldsymbol{y}, 0) \ominus \boldsymbol{x}_k = \mathrm{Log}_{\mathbb{S}_3}(\hat{\boldsymbol{q}}_k^{-1} * \hat{\boldsymbol{q}}_{k-1} * 0.5\mathrm{Exp}_{\mathbb{S}_3}((\boldsymbol{\omega}_{m,k} - \hat{\boldsymbol{b}_{w\,k-1}})dt), \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (\hat{\boldsymbol{b}}_{w,k-1} - \hat{\boldsymbol{b}}_{w,k}), (\hat{\boldsymbol{b}}_{m,k-1} - \hat{\boldsymbol{b}}_{m,k})) \\
&\qquad F_{k-1} \qquad\qquad\qquad\qquad\qquad\qquad = \begin{bmatrix} -\mathrm{Exp}_{\mathbb{SO}_3}((\boldsymbol{\omega}_{m,k} - \hat{\boldsymbol{b}_{w\,k-1}})dt) & -Idt \\ 0 & I \end{bmatrix} \\
&G_{w,k-1}(\hat{\boldsymbol{x}}_{k-1}) \qquad\qquad\qquad\qquad\qquad = \begin{bmatrix} Idt & 0 & 0 \\ 0 & Idt & 0 \\ 0 & 0 & Idt \end{bmatrix} \\
&\qquad W_{\nu,k}(\hat{x}_k) \qquad\qquad\qquad\qquad\qquad\qquad = \mathrm{diag}(Q_{wm}dt^2, Q_{bw}dt^2, Q_{bm}dt^2)
\end{aligned}
$$

measurements :

$$
\begin{aligned}
&\boldsymbol{e}_{y,k}(\hat{\boldsymbol{x}}_k) \quad = \begin{bmatrix} \boldsymbol{y}_{mag,k} - \hat{R}_k^T \boldsymbol{b}_e \\ \boldsymbol{y}_{sun,k} - \hat{R}_k^T \boldsymbol{s}_e \end{bmatrix} \\
&H_k(\hat{\boldsymbol{x}}_k) \quad = \begin{bmatrix} [\hat{R}_k^T \boldsymbol{b}_e]_\times & 0 & 0 \\ [\hat{R}_k^T \boldsymbol{s}_e]_\times & 0 & 0 \end{bmatrix} \\
&G_{\nu,k}(\hat{\boldsymbol{x}}_k) \quad = I_6 \\
&W_{y,k}(\hat{\boldsymbol{x}}_k) \quad = \mathrm{diag}(R_{\nu,mag}, R_{\nu,sun})
\end{aligned}
$$

build matrices :

$$
e(\hat{\boldsymbol{X}}) = \left[\begin{array}{c} e_{u,0}(\hat{\boldsymbol{x}_0}) \\ e_{u,1}(\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_0) \\ e_{u,2}(\hat{\boldsymbol{x}}_2, \hat{\boldsymbol{x}}_1) \\ \vdots \\ e_{u,K}(\hat{\boldsymbol{x}}_K, \hat{\boldsymbol{x}}_{K-1}) \\ \hline e_{y,0}(\hat{\boldsymbol{x}}_0) \\ e_{y,1}(\hat{\boldsymbol{x}}_1) \\ \vdots \\ e_{y,K}(\hat{\boldsymbol{x}}_K) \end{array}\right] \quad \mathbb{H} = \left[\begin{array}{ccccc} I & & & & \\ -F_0 & I & & & \\ & -F_1 & I & & \\ & & \ddots & \ddots & \\ & & & -F_{K-1} & I \\ \hline H_0 & & & & \\ & H_1 & & & \\ & & \ddots & & \\ & & & & H_K \end{array}\right]
$$

$$
W = \mathrm{diag}(W_{u,0}, W_{u,1}, \ldots, W_{u,K}, W_{y,0}, W_{y,1}, \ldots, W_{y,K})
$$

Return $e(\hat{\boldsymbol{X}}), \mathbb{H}(\hat{\boldsymbol{X}}), W(\hat{\boldsymbol{X}})$

When implementing the above algorithm, a sliding window optimization strategy [160] is used since solving a full set of data together is computationally demanding. A sliding window strategy is when only a window of measurements are solved at a time instead of considering the whole dataset. The solution of the previous sliding window is used to initialize the next window of measurements. This strategy drastically reduces the computational requirements of the optimization algorithm while producing comparable results to the optimization of the whole dataset as a single implementation. This process can be visualized in Figure 4.13 demonstrating the sliding window strategy to estimate and log the unknown states or the unknown magnetic bias.

4.7 Simulation Testing of the Fault Detection, Isolation, and Recovery

4.7.1 Fault Detection Results

Using the same simulation environment outlined in Section 4.3 and the fault detection method outlined in Section 4.5, the fault detection results for an increase in magnetometer bias can be observed in Figure 4.15. The magnetometer experiences a constant bias increase at the fault time. The time of the fault is at time 15200. This fault lasts for the 2.5 remaining orbits of the satellite in the simulation. The gating procedure accurately detects the fault and identifies this as a persistent fault once the fault is triggered for an extended time. Once a persistent fault is flagged, it is passed onto the last step, the fault isolation and recovery.

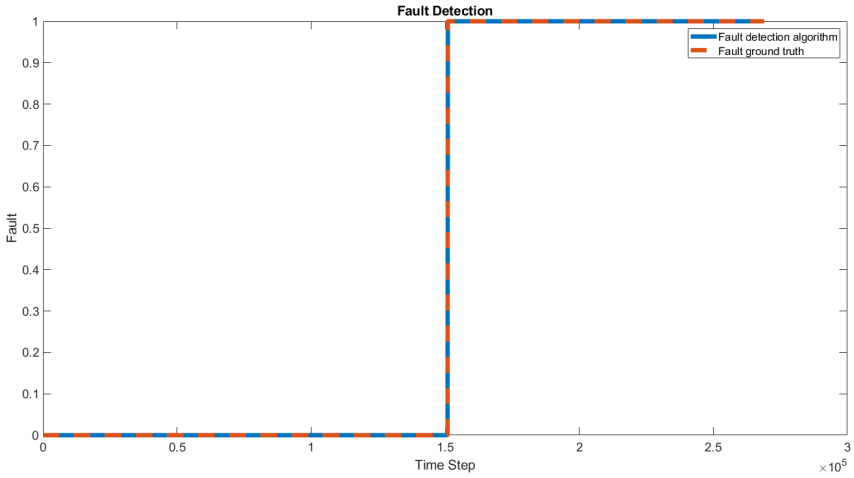


Figure 4.15: The results of the fault gating with a magnetometer bias increase half way through the third orbit.

4.7.2 Fault Isolation and Recovery Results

With the sliding window filter, the following results are observed for a small sample of the orbit in Figure 4.16. It is seen that error bounds are achieved for the state elements, as seen in the red and yellow bounds. The axis angles represent the quaternion estimation converted to Euler angles, and the gyroscope drift represents the gyroscope drift estimation b_w. Lastly, the magnetic bias represents the estimation error and bounds of the unknown magnetometer bias term. It is observed that the unknown bias is tracked within the sliding window filter, which allows for the determination of the new bias. The tracked unknown bias can be used for re-adjusting the satellite magnetometer parameters for fault recovery purposes. A small error within the code forces the estimated value outside of the range of the bounds that will be addressed in future work. Using the sliding window batch optimization, it is apparent that the confidence bounds narrow down on the unknown magnetic bias and can

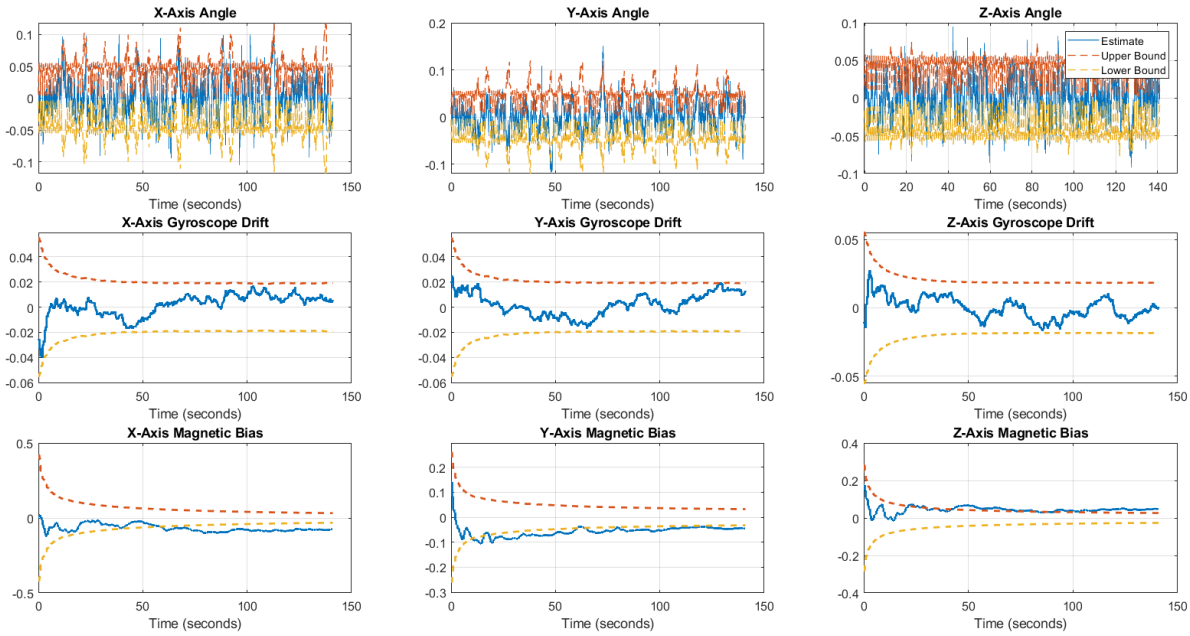


Figure 4.16: The results of the sliding window batch optimization to detect the unknown magnetic bias. The first row represents the Euler angle estimation, the second row is the gyroscope drift, and the last row represents the magnetic bias estimate.

find the correct bias term through offline optimization. Figure 4.17 demonstrates that while the RMSE fluctuates as expected in the sliding window optimization, the RMSE remains low for the duration of the trajectory.

4.8 General Chapter Summary

This chapter presents a machine learning approach through regression learning for unknown magnetic bias recalibration in small satellite attitude determination systems. To detect the faults within the sensor readings, accurate error bounds of the

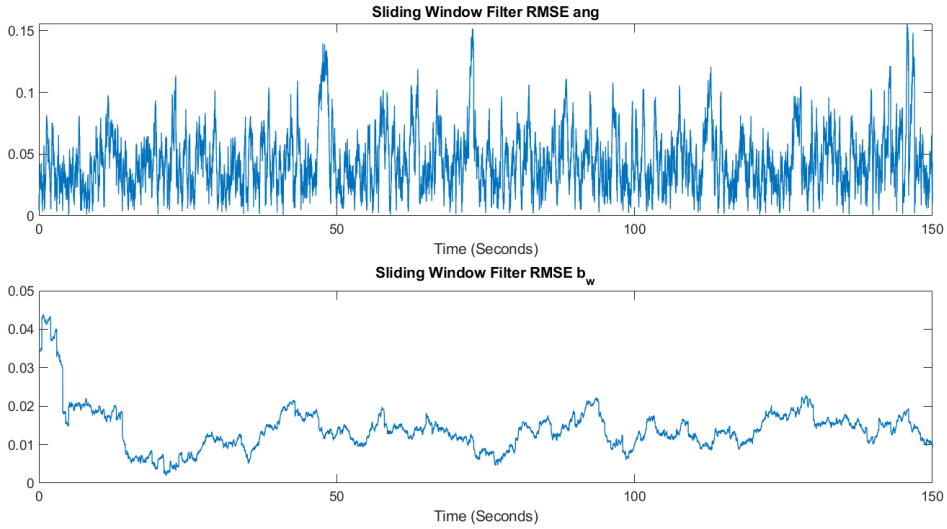


Figure 4.17: The RMSE results of the sliding window batch optimization for a small period of the simulation.

filtering algorithm are required. To meet this requirement, the right error parameterization of the attitude quaternion is implemented to ensure observability consistency compared to the standard error quaternion parameterization. The mathematical calculation of the proposed right error quaternion parameterization is presented, and the overall Multiplicative Extended Kalman Filter used for the attitude determination is described. Using the improved error bounds of the parameterization, a lightweight fault detection algorithm is implemented to identify any fault occurrences and record all sensor data for the fault period. A sliding window filter is implemented to recover the identified fault and determine the unknown magnetic bias within the attitude determination system. A simulation environment was built to validate the right error parameterization, showing a consistent NEES during the eclipse periods of the orbit, whereas the traditional left error parameterization demonstrated a large increase in

the NEES during the eclipse period. Physical testing of a sensor suite was conducted to further validate the right error parameterization in a lab environment versus the conventional error parameterization. Both the fault detection and fault recovery algorithms were implemented in a simulation environment demonstrating near-perfect fault detection and improved bias detection in the sliding window filter.

The results show that the sliding window filter can detect the unknown magnetic bias within the filtering algorithm. Additionally, the right error parameterization improves the error bounds of the real-time EKF error bounds. The right error parameterization shows a significantly improved NEES performance during eclipse periods of the orbit. In the future, this work should further investigate the impacts of varying magnetic faults on the sliding window filter. Also, increased research into the right error quaternion parameterization should be conducted and experimentally validated, as it clearly presents advantages over the standard parameterization in the eclipse period.

Chapter 5

Conclusion

5.1 General Synopsis and Significant Results

The focus of this research was to implement improved machine learning solutions in two different harsh environment applications. The first application focused on developing a sea ice classification algorithm to provide navigational aid for ships encountering sea ice conditions in polar seas. Due to the high variance in ice conditions and environmental states, ML is ideal for sea ice detection and further classification. The second environment analyzed was the application of ML for small satellites. With the increase in accessibility of nano-class satellites, Memorial University is developing its own satellite, Killick-1. With any design, trade-offs exist between subsystems resulting in potential unobservable states during an orbit due to limited sensors. With limited sensors, fault mitigation and detection are critical and prove to be an excellent field for ML. The study formed the following objectives:

1. Evaluation of the application of state-of-the-art semantic segmentation networks

for the novel application of sea ice detection and sea ice classification in polar oceans.

2. Evaluation of the recently developed right $\mathbb{S}3$ error parameterization for attitude determination of small satellites.

3. Propose a regression machine learning approach for fault detection and recovery of magnetometer faults within attitude determination systems of satellites based on nonlinear rotation error parameterization.

5.1.1 Research Summary Based on Objective I

The first objective of this study was to evaluate the pixel-wise accuracy and IoU of two semantic segmentation neural networks for the application of sea ice detection and sea ice classification. Two state-of-the-art networks were evaluated for sea ice detection, namely, SegNet and PSPNet101. Using a custom four-class sea ice detection dataset, PSPNet101 outperformed SegNet in every metric with an average IoU of 90.1% compared to 69.8% for SegNet. The pixelwise accuracies of the models demonstrate that the SegNet architecture has significantly more classification errors as no class is over 95% compared to PSPNet, which boasts all classes at least 96% accurate. The models were further expanded to classify eight classes in the expanded sea ice classification dataset. PSPNet101 boasted an average IoU of 75.5% on the expanded dataset with a combined sea ice classification IoU of 83.8%. SegNet performed poorly on the expanded dataset with an average IoU of 53.6% and a combined sea ice classification IoU of 54.8%. These results validate the applicability of deep learning methods for sea ice detection and classification using images captured onboard an ice

breaker. These can be further enhanced by incorporating additional ice types and operational data to support marine navigation and mapping applications.

5.1.2 Research Summary Based on Objective II

The second objective of this study was to evaluate the recently developed $\mathbb{S}3$ error parameterization for small satellite attitude determination. First, a mathematical model of a three sensor satellite was presented, and the observability consistency of the right $\mathbb{S}3$ error parameterization was for a satellite with only magnetometer readings during the eclipse period. A simulation environment was developed to evaluate the right error parameterization versus the conventional left error parameterization. The right quaternion error parameterization demonstrated significant improvement over the conventional left quaternion error parameterization NEES and RMSE calculations during the eclipse period of the orbit. The right parameterization was then validated using physical testing and demonstrated the same performance as the left quaternion error parameterization in a lab environment. The results can be further improved with a more intensive lab testing to demonstrate the left error parameterization drifting.

5.1.3 Research Summary Based on Objective III

The third objective of this study was to design a machine learning approach to handle magnetometer faults within an attitude determination system of satellites using a nonlinear rotation error parameterization. Using the improved error bounds from the right $\mathbb{S}3$ error parameterization, a fault detection algorithm was developed to detect any outliers within the NIS. Once the fault detection method was validated,

a sliding window filter was developed to use regression machine learning to estimate the unknown magnetic bias within the magnetometer. The sliding window filter demonstrated improved error bounds for the sensor and improved the estimation of the fixed magnetic bias. These results validate the application of machine learning for small satellite attitude determination applications, which can be further enhanced with increased sensor data and allows for complete control of the error mapping and filtering algorithms which is not readily available in existing filtering solutions.

5.2 Contributions

The resulting contributions of the thesis are as follows:

- A new neural network for sea ice detection and classification with 99.2% pixel wise accuracy and 75.5% IoU.
- A new sea ice classification and detection dataset with pixelwise labelling of 1197 images into eight classes.
- Experimental validation of the neural network for inference performance achieving 0.5 fps speed for processing on an AMD Ryzen 3800X with 32 GB of DDR4 RAM computing hardware.
- Mathematical proof and numerical validation of the consistency performance of the right $\mathbb{S}3$ error parameterization for small satellite attitude determination during the eclipse period.
- Evaluation of a magnetic anomaly detection and unknown magnetic bias recalibration method based on the nonlinear $\mathbb{S}3$ error parameterization and sliding

window machine learning regression for small satellites.

The following articles are a result of the thesis:

- **B. Dowden**, O. De Silva, and W. Huang, 2019,"Object Classification via Semantic Segmentation for Icebreakers", *Newfoundland Electrical and Computer Engineering Conference 2019 (NECEC)*, St. John's, Canada, November 19. [161]

- **B. Dowden**, O. De Silva, and W. Huang,"Sea Ice Image Semantic Segmentation Using Deep Neural Networks," *OCEANS 2020: Singapore*, Virtual, Singapore, October 5-14. [162]

- **B. Dowden**, O. De Silva, W. Huang, and D. Oldford, "Sea Ice Classification via Deep Neural Network Semantic Segmentation", *IEEE Sensors Journal*, doi: 10.1109/JSEN.2020.3031475. [163]

- **B. Dowden**, O. De Silva, and W. Huang, 2020,"Right Quaternion Parameterization for CubeSat Attitude Determination", *Newfoundland Electrical and Computer Engineering Conference 2020 (NECEC)*, St. John's, Canada, November 19. [164]

5.3 Future Directives

The work in Chapter 3 proposes two semantic segmentation neural networks that are capable of functioning efficiently in their respective environments. These datasets should be expanded to include other icebreaker journeys and more extensive ice types

distribution, including multiyear ice. Additionally, the classification of different ice objects is critical for safe navigation. Ice objects such as icebergs, bergy bits, and pressure ridges are some of the most hazardous ice objects which also need to be identified. These ice objects can be flagged with additional classifiers to provide a more robust and complete ice navigation solution.

The work in Chapter 4 outlines a regression machine learning approach for magnetometer bias detection. While the right error parameterization improved the NEES for eclipse periods, additional physical testing with a more robust test environment should be conducted to validate the mathematical and simulation results further. The SWF was only implemented on a small subset of data, and it has been assumed that the fault isolation is straightforward from the fault detection. In the future, the fault isolation methodology should be incorporated into the overall fault algorithm and analyze the performance for multiple different faults, not just magnetometer bias faults.

Lastly, it is critical to ensure proper convergence or stability of the model with any machine learning model. For the real-world implementation of both machine learning approaches presented in this work, adaptive models must be developed to continually re-train and improve based on detected errors and faults in the model. The initial implementation of each system would require human analysis of the results to validate the model further. A system should be developed for the ice navigation application to increase training based on high fault classes and continually re-train on the faults. For the regression machine learning in Chapter 4, increased varied fault experimentation is required to increase the stability of the model. For full implementation, a cross-validation model should be implemented to evaluate the machine learning output

versus mathematical models.

Bibliography

[1] H. Rashidi, N. Tran, E. Betts, L. Howell, and R. Green, "Artificial Intelligence and Machine Learning in Pathology: The Present Landscape of Supervised Methods," *Academic Pathology*, vol. 6, p. 237428951987308, Sept. 2019.

[2] C. Piech, "Stanford University CS221," 2013.

[3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *MICCAI*, (Cham), pp. 234–241, 2015.

[4] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intel*, vol. 39, pp. 2481–2495, Dec. 2017.

[5] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," in *CVPR*, pp. 6230–6239, July 2017.

[6] M. E. Johnston and G. W. Timco, "Guide for Understanding and Identifying Old Ice in Summer," p. 8.

[7] C. Brooks, *Two months breaking ice (in under five minutes)*. May 2013.

[8] D. Hebb, *The Organization of Behavior: A Neuropsychological Theory*. Psychology Press, 2005.

[9] "Samuels Checkers Player," in *Encyclopedia of Machine Learning* (C. Sammut and G. I. Webb, eds.), pp. 881–881, Boston, MA: Springer US, 2010.

[10] H. Kagaya, K. Aizawa, and M. Ogawa, "Food Detection and Recognition Using Convolutional Neural Network," in *Proceedings of the 22nd ACM international conference on Multimedia*, MM '14, (New York, NY, USA), pp. 1085–1088, Association for Computing Machinery, Nov. 2014.

[11] L. M. Dang, S. I. Hassan, S. Im, and H. Moon, "Face image manipulation detection based on a convolutional neural network," *Expert Systems with Applications*, vol. 129, pp. 156–168, Sept. 2019.

[12] J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," in *2017 International Conference on Computing Networking and Informatics (ICCNI)*, pp. 1–9, Oct. 2017.

[13] E. Anthi, L. Williams, M. Rhode, P. Burnap, and A. Wedgbury, "Adversarial attacks on machine learning cybersecurity defences in Industrial Control Systems," *Journal of Information Security and Applications*, vol. 58, p. 102717, May 2021.

[14] S. Banerjee, B. Samynathan, J. A. Abraham, and A. Chatterjee, "Real-Time Error Detection in Nonlinear Control Systems Using Machine Learning As-

sisted State-Space Encoding," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, pp. 576–592, Mar. 2021.

[15] A. Mcgovern and K. Wagstaff, "Machine learning in space: Extending our reach," *Machine Learning*, vol. 84, pp. 335–340, Sept. 2011.

[16] HarperCollins, "Harsh," in *Collins English Dictionary*, HarperCollins Publishers.

[17] K. C. A. Khanzode and R. Sarode, "Advantages and Disadvantages of Artificial Intelligence and Machine Learning: A Literature Review," *International Journal of Library & Information Science*, vol. 9, pp. 30–36, Apr. 2020.

[18] G. Camps-Valls, "New machine-learning paradigm provides advantages for remote sensing," *Spie Newsroom*, Jan. 2008.

[19] S. Wang, W. Chaovalitwongse, and R. Babuska, "Machine Learning Algorithms in Bipedal Robot Control," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, pp. 728–743, Sept. 2012.

[20] I. G. Maglogiannis, *Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in EHealth, HCI, Information Retrieval and Pervasive Technologies*. IOS Press, 2007.

[21] M. Shanmukhi, K. Durga, M. Madela, and K. Keerthana, "Convolutional neural network for supervised image classification," *International Journal of Pure and Applied Mathematics*, vol. 119, pp. 77–82, Jan. 2018.

[22] Q. Ha, K. Watanabe, T. Karasawa, Y. Ushiku, and T. Harada, "MFNet: Towards real-time semantic segmentation for autonomous vehicles with multi-spectral scenes," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5108–5115, Sept. 2017.

[23] N. Zimmerman, A. A. Presto, S. P. N. Kumar, J. Gu, A. Hauryliuk, E. S. Robinson, A. L. Robinson, and R. Subramanian, "A machine learning calibration model using random forests to improve sensor performance for lower-cost air quality monitoring," *Atmospheric Measurement Techniques*, vol. 11, pp. 291–313, Jan. 2018.

[24] M. Usama, J. Qadir, A. Raza, H. Arif, K.-l. A. Yau, Y. Elkhatib, A. Hussain, and A. Al-Fuqaha, "Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges," *IEEE Access*, vol. 7, pp. 65579–65615, 2019.

[25] I. C. Education, "What is Unsupervised Learning?," Apr. 2021.

[26] A. S. Polydoros and L. Nalpantidis, "Survey of Model-Based Reinforcement Learning: Applications on Robotics," *Journal of Intelligent & Robotic Systems*, vol. 86, pp. 153–173, May 2017.

[27] R. Miorelli, A. Kulakovskyi, B. Chapuis, O. DAlmeida, and O. Mesnil, "Supervised learning strategy for classification and regression tasks applied to aeronautical structural health monitoring problems," *Ultrasonics*, vol. 113, p. 106372, May 2021.

[28] M. R. Segal, "Machine Learning Benchmarks and Random Forest Regression," Apr. 2004.

[29] N. Shahid, I. H. Naqvi, and S. B. Qaisar, "One-class support vector machines: analysis of outlier detection for wireless sensor networks in harsh environments," *Artificial Intelligence Review*, vol. 43, pp. 515–563, Apr. 2015.

[30] M. J. Islam, C. Edge, Y. Xiao, P. Luo, M. Mehtaz, C. Morse, S. S. Enan, and J. Sattar, "Semantic Segmentation of Underwater Imagery: Dataset and Benchmark," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1769–1776, Oct. 2020.

[31] A. Lay-Ekuakille, M. A. Ugwiri, J. D. Okitadiowo, C. Chiffi, and A. Pietrosanto, "Computer Vision for Sensed Images Approach in Extremely Harsh Environments: Blast Furnace Chute Wear Characterization," *IEEE Sensors Journal*, vol. 21, pp. 11969–11976, May 2021.

[32] I. Kubat and G. Timco, "Vessel Damage in the Canadian Arctic.," in *Proceedings 17th International Conference on Port and Ocean Engineering under Arctic Conditions*, p. 11, NRC, June 2003.

[33] N. Zakhvatkina, V. Smirnov, and I. Bychkova, "Satellite SAR Data-based Sea Ice Classification: An Overview," *Geosciences*, vol. 9, p. 152, Apr. 2019.

[34] I. M. Organization, "Guidelines For Ships Operating In Arctic Ice-Covered Waters," Dec. 2002.

[35] J. Miller and K. Satterfield, "Iceberg detection by radar," in *OCEANS '85 - Ocean Engineering and the Environment*, pp. 405–410, Nov. 1985.

[36] M. Langer and J. Bouwmeester, "Reliability of CubeSats - Statistical Data, Developers' Beliefs and the Way Forward," in *Proceedings of the AIAA/USU Conference on Small Satellites, SSC16-X-2*, (Logan, United States), p. 12, Aug. 2016.

[37] T. Inamori, "In-Orbit Magnetic Disturbance Estimation and Compensation Using UKF in Nano-Satellite Mission," in *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, June 2012.

[38] T. Zhang, K. Wu, D. Su, S. Huang, and G. Dissanayake, "An Invariant-EKF VINS Algorithm for Improving Consistency," *arXiv:1702.07920 [cs]*, Mar. 2017.

[39] A. Joshi, V. Gavriloiu, A. Barua, A. Garabedian, P. Sinha, and K. Khorasani, "Intelligent and learning-based approaches for health monitoring and fault diagnosis of RADARSAT-1 attitude control system," in *2007 IEEE International Conference on Systems, Man and Cybernetics*, pp. 3177–3183, Oct. 2007.

[40] X. Zhang, J. Jin, Z. Lan, C. Li, M. Fan, Y. Wang, X. Yu, and Y. Zhang, "ICENET: A Semantic Segmentation Deep Network for River Ice by Fusing Positional and Channel-Wise Attentive Features," *Remote Sensing*, vol. 12, p. 221, Jan. 2020.

[41] E. Kim, G. S. Dahiya, S. Lset, and R. Skjetne, "Can a computer see what an

ice expert sees? Multilabel ice objects classification with convolutional neural networks," *Results in Engineering*, vol. 4, p. 100036, Dec. 2019.

[42] H. Boulze, A. Korosov, and J. Brajard, "Classification of Sea Ice Types in Sentinel-1 SAR Data Using Convolutional Neural Networks," *Remote Sensing*, vol. 12, p. 2165, Jan. 2020.

[43] T. Reid, T. Walter, P. Enge, and A. Fowler, "Crowdsourcing Arctic Navigation Using Multispectral Ice Classification and GNSS," in *27th International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS 2014*, vol. 1, 2014.

[44] M. W. M. Said, O. De Silva, G. K. I. Mann, C. Daley, J. R. Dolny, D. Oldford, and R. G. Gosine, "LiDAR and Vision Based Pack Ice Field Estimation for Aided Ship Navigation," report, Memorial University of Newfoundland, St. John's, Newfoundland and Labrador, Aug. 2017.

[45] O. De Silva, G. Mann, R. Gosine, C. Daley, J. Dolny, and D. Oldford, "Vision Analysis of Pack Ice for Potential Use in a Hazard Warning and Avoidance System," *Memorial University of Newfoundland*, 2016.

[46] A. Barrau and S. Bonnabel, "Invariant kalman filtering," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 237–257, 2018.

[47] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. USA: Prentice-Hall, Inc., 2006.

[48] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, pp. 62–66, Jan. 1979.

[49] Y. Yang and H. Yan, "An adaptive logical method for binarization of degraded document images," *Pattern Recognition*, vol. 33, pp. 787–807, May 2000.

[50] T. Peli and D. Malah, "A study of edge detection algorithms," *Computer Graphics and Image Processing*, vol. 20, pp. 1–21, Sept. 1982.

[51] J. M. White and G. D. Rohrer, "Image thresholding for optical character recognition and other applications requiring character image extraction," *IBM J. Res. Dev*, vol. 27, pp. 400–411, 1983.

[52] P. Wellner, "Interacting with paper on the DigitalDesk," *Communications of the ACM*, vol. 36, pp. 87–96, July 1993.

[53] D. Bradley and G. Roth, "Adaptive Thresholding using the Integral Image," *Journal of Graphics Tools*, vol. 12, pp. 13–21, Jan. 2007.

[54] V. Jumb, M. Sohani, and A. Shrivas, "Color Image Segmentation Using K-Means Clustering and Otsu s Adaptive Thresholding," 2014.

[55] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 881–892, July 2002.

[56] N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image Segmentation Using

K -means Clustering Algorithm and Subtractive Clustering Algorithm," *Procedia Computer Science*, vol. 54, pp. 764–771, 2015.

[57] S. Tatiraju and A. Mehta, "Image segmentation using k-means clustering, em and normalized cuts," *University of California Irvine*, 01 2008.

[58] A. Brhret, "abreheret/PixelAnnotationTool," Sept. 2019.

[59] S. Derivaux, G. Forestier, C. Wemmert, and S. Lefvre, "Supervised image segmentation using watershed transform, fuzzy classification and evolutionary computation," *Pattern Recognition Letters*, vol. 31, no. 15, pp. 2364–2374, 2010.

[60] M. Straka, A. La Cruz, A. Kochl, M. Sramek, E. Groller, and D. Fleischmann, "3D watershed transform combined with a probabilistic atlas for medical image segmentation," *Journal of Medical Informatics & Technologies*, vol. Vol. 6, 2003.

[61] F. B. Tek, A. G. Dempster, and I. Kale, "Blood Cell Segmentation Using Minimum Area Watershed and Circle Radon Transformations," in *Mathematical Morphology: 40 Years On* (C. Ronse, L. Najman, and E. Decencire, eds.), Computational Imaging and Vision, (Dordrecht), pp. 441–454, Springer Netherlands, 2005.

[62] M. Turker and E. Sumer, "Buildingbased damage detection due to earthquake using the watershed segmentation of the postevent aerial images," *International Journal of Remote Sensing*, vol. 29, no. 11, pp. 3073–3089, 2008.

[63] S. Beucher, "Watershed, Hierarchical Segmentation and Waterfall Algorithm," in *Mathematical Morphology and Its Applications to Image Processing* (J. Serra

and P. Soille, eds.), Computational Imaging and Vision, pp. 69–76, Dordrecht: Springer Netherlands, 1994.

[64] P. Acharjya and D. Ghoshal, "An Overview on Watershed Transform and Its Consequences," *International Journal of Engineering and Innovative Technology*, vol. 1, pp. 168–172, May 2012.

[65] B. Manjunath, C. Shekhar, and R. Chellappa, "A new approach to image feature detection with applications," *Pattern Recognition*, vol. 29, pp. 627–640, Apr. 1996.

[66] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, Nov. 2004.

[67] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," in *Computer Vision ECCV 2006* (A. Leonardis, H. Bischof, and A. Pinz, eds.), vol. 3951, pp. 404–417, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[68] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893 vol. 1, June 2005.

[69] H. Goncalves, L. Corte-Real, and J. A. Goncalves, "Automatic Image Registration Through Image Segmentation and SIFT," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, pp. 2589–2600, July 2011.

[70] Y. Li, Y. Wang, W. Huang, and Z. Zhang, "Automatic image stitching us-

ing SIFT," in *2008 International Conference on Audio, Language and Image Processing*, pp. 568–571, July 2008.

[71] H. Tamimi, H. Andreasson, A. Treptow, T. Duckett, and A. Zell, "Localization of mobile robots with omnidirectional vision using Particle Filter and iterative SIFT," *Robotics and Autonomous Systems*, vol. 54, pp. 758–765, Sept. 2006.

[72] T. Kalinke, C. Tzomakas, and W. V. Seelen, "A Texture-based Object Detection and an adaptive Model-based Classification," in *in Procs. IEEE Intelligent Vehicles Symposium98*, pp. 341–346, 1998.

[73] A. Wibmer, H. Hricak, T. Gondo, K. Matsumoto, H. Veeraraghavan, D. Fehr, J. Zheng, D. Goldman, C. Moskowitz, S. W. Fine, V. E. Reuter, J. Eastham, E. Sala, and H. A. Vargas, "Haralick texture analysis of prostate MRI: utility for differentiating non-cancerous prostate from prostate cancer and differentiating prostate cancers with different Gleason scores," *European Radiology*, vol. 25, pp. 2840–2850, Oct. 2015.

[74] H. Murray, A. Lucieer, and R. Williams, "Texture-based classification of sub-Antarctic vegetation communities on Heard Island," *International Journal of Applied Earth Observation and Geoinformation*, vol. 12, pp. 138–149, June 2010.

[75] P. Mhangara and J. Odindi, "Potential of texture-based classification in urban landscapes using multispectral aerial photos," *South African Journal of Science*, vol. 109, pp. 1–8, Jan. 2013.

[76] Y. Deng and B. Manjunath, "Unsupervised Segmentation of Color-Texture Regions in Images and video," *Pattern Analysis and Machine Intelligence*, vol. 13, pp. 800–810, Jan. 2001.

[77] S. Geffray, N. Klutchnikoff, and M. Vimond, "Illumination Problems in Digital Images. A Statistical Point of View," *J. Multivar. Anal.*, vol. 150, pp. 191–213, Sept. 2016.

[78] H. Y. Chong, S. J. Gortler, and T. Zickler, "A perception-based color space for illumination-invariant image processing," *ACM Transactions on Graphics*, vol. 27, pp. 1–7, Aug. 2008.

[79] N. Wright and C. Polashenski, "Open Source Algorithm for Detecting Sea Ice Surface Features in High Resolution Optical Imagery," *The Cryosphere Discussions*, pp. 1–37, Sept. 2017.

[80] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to End Learning for Self-Driving Cars," *arXiv:1604.07316 [cs]*, Apr. 2016.

[81] H. Abdi, "A NEURAL NETWORK PRIMER," 1994.

[82] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, p. e00938, Nov. 2018.

[83] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with

deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84–90, May 2017.

[84] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, (Boston, MA, USA), pp. 3431–3440, June 2015.

[85] W. He, M. Wu, M. Liang, and S.-K. Lam, "CAP: Context-Aware Pruning for Semantic Segmentation," in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 959–968, Jan. 2021.

[86] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *CVPR*, pp. 770–778, June 2016.

[87] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," *arXiv:1903.11027 [cs, stat]*, Mar. 2019.

[88] L. Leal-Taix, A. Milan, I. Reid, S. Roth, and K. Schindler, "MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking," *arXiv:1504.01942 [cs]*, Apr. 2015.

[89] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Computer Vision ECCV 2014*, Lecture Notes in Computer Science, pp. 740–755, 2014.

[90] C. Mariscal-Garca, W. Flores-Fuentes, D. Hernndez-Balbuena, J. C. Rodrguez-Quionez, O. Sergiyenko, F. F. Gonzlez-Navarro, and J. E. Miranda-Vega, "Clas-

sification of Vehicle Images through Deep Neural Networks for Camera View Position Selection," in *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*, pp. 1376–1380, June 2020.

[91] J. J. Sanchez-Castro, J. C. Rodrguez-Quionez, L. R. Ramrez-Hernndez, G. Galaviz, D. Hernndez-Balbuena, G. Trujillo-Hernndez, W. Flores-Fuentes, P. Mercorelli, W. Hernndez-Perdomo, O. Sergiyenko, and F. F. Gonzlez-Navarro, "A Lean Convolutional Neural Network for Vehicle Classification," in *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*, pp. 1365–1369, June 2020.

[92] A. King, S. M. Bhandarkar, and B. M. Hopkinson, "Deep Learning for Semantic Segmentation of Coral Reef Images Using Multi-View Information," in *CVPRW*, (Long Beach), pp. 1–10, 2019.

[93] R. Lguensat, M. Sun, R. Fablet, P. Tandeo, E. Mason, and G. Chen, "EddyNet: A Deep Neural Network For Pixel-Wise Classification of Oceanic Eddies," in *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 1764–1767, July 2018.

[94] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, p. 60, July 2019.

[95] L. Torrey and J. Shavlik, *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, ch. Chapter 11 Transfer Learning, pp. 242–264. 2010.

[96] C. Hall, "Attitude Determination," in *Spacecraft Attitude Dynamics and Control*, Virginia Tech, 2003.

[97] J. Farrell, *Aided Navigation: GPS with High Rate Sensors*. USA: McGraw-Hill, Inc., 1 ed., 2008.

[98] F. Wittmann, O. Lambercy, and R. Gassert, "Magnetometer-Based Drift Correction During Rest in IMU Arm Motion Tracking," *Sensors*, vol. 19, p. 1312, Jan. 2019.

[99] D. Gautam, A. Lucieer, C. Watson, and C. McCoull, "Lever-arm and boresight correction, and field of view determination of a spectroradiometer mounted on an unmanned aircraft system," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 155, pp. 25–36, Sept. 2019.

[100] L. Euler, "Nova methodus innumerabiles aequationes differentiales secundi gradus reducendi ad aequationes differentiales primi gradus," *Commentarii academiae scientiarum Petropolitanae*, pp. 124–137, Jan. 1732.

[101] Tait, "VIII.On the Rotation of a Rigid Body about a Fixed Point," *Transactions of the Royal Society of Edinburgh*, vol. 25, no. 2, pp. 261–303, 1869.

[102] D. Brezov, C. Mladenova, and I. Mladenov, "New Perspective on the Gimbal Lock Problem," *AIP Conference Proceedings*, vol. 1570, pp. 367–374, Mar. 2013.

[103] Y. Dong, "8 - MEMS inertial navigation systems for aircraft," in *MEMS for Automotive and Aerospace Applications* (M. Kraft and N. M. White, eds.),

Woodhead Publishing Series in Electronic and Optical Materials, pp. 177–219, Woodhead Publishing, Jan. 2013.

[104] W. Premerlani and P. Bizard, "Direction Cosine Matrix IMU: Theory," *DIY DRONE: USA*, Jan. 2009.

[105] B. Smeresky and A. Rizzo, *Kinematics: On Direction Cosine Matrices*. IntechOpen, Jan. 2020.

[106] Karsten Groekatthfer and Zizung Yoon, "Introduction into quaternions for spacecraft attitude representation," tech. rep., Technical University of Berlin, Belin, Germany, May 2012.

[107] N. Bickford, "Why Do the Unit Quaternions Double-Cover the Space of Rotations?," p. 17, 2019.

[108] C. F. F. Karney, "Quaternions in molecular modeling," *Journal of Molecular Graphics and Modelling*, vol. 25, pp. 595–604, Jan. 2007.

[109] B. D. Tapley, B. E. Schutz, and G. H. Born, "Chapter 2 - The Orbit Problem," in *Statistical Orbit Determination* (B. D. Tapley, B. E. Schutz, and G. H. Born, eds.), pp. 17–91, Burlington: Academic Press, Jan. 2004.

[110] M. S. Mohammed, H. Boussadia, A. Bellar, and A. Adnane, "Performance comparison of attitude determination, attitude estimation, and nonlinear observers algorithms," *Journal of Physics: Conference Series*, vol. 783, p. 012017, Jan. 2017.

[111] M. Shuster and S. d. Oh, "Three-axis attitude determination from vector observations | Journal of Guidance, Control, and Dynamics," *Journal of Guidance and Control*, vol. 4, Feb. 1981.

[112] Y. Cheng and M. Shuster, "An Improvement to the Implementation of the QUEST Algorithm," *Journal of Guidance, Control, and Dynamics*, vol. 37, pp. 301–305, Jan. 2014.

[113] C. Park and P. J. G. Teunissen, "Integer least squares with quadratic equality constraints and its application to GNSS attitude determination systems," *International Journal of Control, Automation and Systems*, vol. 7, pp. 566–576, Aug. 2009.

[114] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASMEJournal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[115] E. Lefferts, F. Markley, and M. Shuster, "Kalman Filtering for Spacecraft Attitude Estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, pp. 417–429, Sept. 1982.

[116] E. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, (Lake Louise, Alta., Canada), pp. 153–158, IEEE, 2000.

[117] L. Markley, "Attitude Error Representations for Kalman Filtering," *Journal*

of Guidance Control and Dynamics - J GUID CONTROL DYNAM, vol. 26, pp. 311–317, Mar. 2003.

[118] L. Cao, D. Qiao, and X. Chen, "Laplace 1 Huber based cubature Kalman filter for attitude estimation of small satellite," *Acta Astronautica*, vol. 148, pp. 48–56, July 2018.

[119] . Odry, I. Kecskes, P. Sarcevic, Z. Vizvari, A. Toth, and P. Odry, "A Novel Fuzzy-Adaptive Extended Kalman Filter for Real-Time Attitude Estimation of Mobile Robots," *Sensors*, vol. 20, p. 803, Jan. 2020.

[120] Q. Zhou, Z. Li, G. Yu, H. Li, and N. Zhang, "A novel adaptive Kalman filter for Euler-angle-based MEMS IMU/magnetometer attitude estimation," *Measurement Science and Technology*, vol. 32, p. 045104, Feb. 2021.

[121] J. L. Crassidis and F. L. Markley, "Minimum Model Error Approach for Attitude Estimation," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 6, pp. 1241–1247, 1997.

[122] A. Nadler, I. Bar-Itzhack, and H. Weiss, "On algorithms for attitude estimation using GPS," in *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No.00CH37187)*, vol. 4, pp. 3321–3326 vol.4, Dec. 2000.

[123] J. Sol, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," 2021.

[124] S. Bonnabel, "Symmetries in observer design: Review of some recent results

and applications to ekf-based slam," *Computing Research Repository - CORR*, vol. 422, 05 2011.

[125] M. Brossard, A. Barrau, and S. Bonnabel, "Exploiting symmetries to design ekfs with consistency properties for navigation and slam," *IEEE Sensors Journal*, vol. 19, pp. 1572–1579, Feb 2019.

[126] P. F. Orr, A. Zoccheddu, L. Sassu, C. Mattia, R. Cozza, and S. Arena, "Machine Learning Approach Using MLP and SVM Algorithms for the Fault Prediction of a Centrifugal Pump in the Oil and Gas Industry," *Sustainability*, vol. 12, p. 4776, Jan. 2020.

[127] Y. Li, "A Fault Prediction and Cause Identification Approach in Complex Industrial Processes Based on Deep Learning," *Computational Intelligence and Neuroscience*, vol. 2021, p. 6612342, Mar. 2021.

[128] S. Yin, B. Xiao, S. X. Ding, and D. Zhou, "A Review on Recent Development of Spacecraft Attitude Fault Tolerant Control System," *IEEE Transactions on Industrial Electronics*, vol. 63, pp. 3311–3320, May 2016.

[129] Q. Shen, D. Wang, S. Zhu, and E. K. Poh, "Integral-Type Sliding Mode Fault-Tolerant Control for Attitude Stabilization of Spacecraft," *IEEE Transactions on Control Systems Technology*, vol. 23, pp. 1131–1138, May 2015.

[130] L. Cao and X. Chen, "Minimum sliding mode error feedback control for inner-formation satellite system with J2 and small eccentricity," *Science China Information Sciences*, vol. 59, p. 072203, June 2016.

[131] Y. Cheng, B. Jiang, N. Lu, T. Wang, and Y. Xing, "Incremental locally linear embedding-based fault detection for satellite attitude control systems," *Journal of the Franklin Institute*, vol. 353, pp. 17–36, Jan. 2016.

[132] F. N. Pirmoradi, F. Sassani, and C. W. de Silva, "Fault detection and diagnosis in a spacecraft attitude determination system," *Acta Astronautica*, vol. 65, pp. 710–729, Sept. 2009.

[133] A. Rahimi, K. Dev Kumar, and H. Alighanbari, "Fault detection and isolation of control moment gyros for satellite attitude control subsystem," *Mechanical Systems and Signal Processing*, vol. 135, p. 106419, Jan. 2020.

[134] H. A. Talebi, K. Khorasani, and S. Tafazoli, "A Recurrent Neural-Network-Based Sensor and Actuator Fault Detection and Isolation for Nonlinear Systems With Application to the Satellite's Attitude Control Subsystem," *IEEE Transactions on Neural Networks*, vol. 20, pp. 45–60, Jan. 2009.

[135] Z. Tu, F. Fei, M. Eagon, X. Zhang, D. Xu, and X. Deng, *Redundancy-Free UAV Sensor Fault Isolation And Recovery.* Nov. 2018.

[136] J. Yang, Y. Chen, and L. Zhang, "An Efficient Approach for Fault Detection, Isolation, and Data Recovery of Self-Validating Multifunctional Sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, pp. 543–558, Mar. 2017.

[137] M. Badura, P. Batog, A. Drzeniecka-Osiadacz, and P. Modzel, "Regression methods in the calibration of low-cost sensors for ambient particulate matter measurements," *SN Applied Sciences*, vol. 1, p. 622, May 2019.

[138] C. Ahmadizadeh and C. Menon, "Investigation of Regression Methods for Reduction of Errors Caused by Bending of FSR-Based Pressure Sensing Systems Used for Prosthetic Applications," *Sensors*, vol. 19, p. 5519, Dec. 2019.

[139] B. Grandvallet, A. Zemouche, M. Boutayeb, and S. Changey, "A sliding window filter for real-time attitude independent TAM calibration," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 467–472, Dec. 2010.

[140] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[141] E. a. C. C. Canada, "Manual of Ice (MANICE)," Dec. 2015.

[142] M. S. Government of Canada; Transport Canada; Safety and Security, "Arctic Ice Regime Shipping System (AIRSS) Standards - TP 12259," Jan. 2010.

[143] O. P. Smith, United States National Ocean Service Office of Response and Restoration, United States National Environmental Satellite Data and Information Service, and National Ice Center, *Observers Guide to Sea Ice*. Silver Spring, Md: U.S. Department of Commerce, ebook ed., 2007.

[144] A. Mouzakitis, "Classification of fault diagnosis methods for control systems," *Measurement and Control*, vol. 46, no. 10, pp. 303–308, 2013.

[145] J. Li, K. Pan, and Q. Su, "Sensor fault detection and estimation for switched

power electronics systems based on sliding mode observer," *Applied Mathematics and Computation*, vol. 353, pp. 282–294, 2019.

[146] H. Wu, X. Pei, J. Li, H. Gao, and Y. Bai, "An improved magnetometer calibration and compensation method based on levenbergmarquardt algorithm for multi-rotor unmanned aerial vehicle," *Measurement and Control*, vol. 53, no. 3-4, pp. 276–286, 2020.

[147] R. Alonso and M. D. Shuster, "Complete Linear Attitude-Independent Magnetometer Calibration," *The Journal of the Astronautical Sciences*, vol. 50, pp. 477–490, Dec. 2002.

[148] H. Geng, Y. Liang, Y. Liu, and F. E. Alsaadi, "Bias estimation for asynchronous multi-rate multi-sensor fusion with unknown inputs," *Information Fusion*, vol. 39, pp. 139–153, Jan. 2018.

[149] N. G. M. Team and B. G. Survey, "World Magnetic Model 2020," 2020. Publisher: NOAA National Centers for Environmental Information.

[150] J.-P. Berrut and L. N. Trefethen, "Barycentric Lagrange Interpolation," *SIAM Rev.*, vol. 46, pp. 501–517, Jan. 2004.

[151] F. Li and L. Chang, "MEKF With Navigation Frame Attitude Error Parameterization for INS/GPS," *IEEE Sensors Journal*, vol. 20, pp. 1536–1549, Feb. 2020.

[152] P. Jorgensen, O. Awad, and R. Bishop, "Analysis and Design of a Sub-Optimal MEKF for Low Earth Orbit Attitude Estimation Using a Radically Inexpensive

MEMS IMU," in *Proceedings of the AIAA/USU Conference on Small Satellites*, (Logan, United States), Utah State University, Aug. 2020.

[153] M. Mahooti, "High Precision Orbit Propagator," *MATLAB File Exchange*, 2020.

[154] F. Vitiello, F. Causa, R. Opromolla, and G. Fasano, "Onboard and External Magnetic Bias Estimation for UAS through CDGNSS/Visual Cooperative Navigation," *Sensors*, vol. 21, p. 3582, Jan. 2021.

[155] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400 – 407, 1951.

[156] R. G. Pratt, C. Shin, and G. J. Hick, "GaussNewton and full Newton methods in frequencyspace seismic waveform inversion," *Geophysical Journal International*, vol. 133, pp. 341–362, May 1998.

[157] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[158] X. Zhang, J. Zhang, and L. Liao, "An adaptive trust region method and its convergence," *Science in China Series A: Mathematics*, vol. 45, pp. 620–631, May 2002.

[159] P.-A. Absil, C. Baker, and K. Gallivan, "Trust-Region Methods on Riemannian Manifolds," *Foundations of Computational Mathematics*, vol. 7, pp. 303–330, July 2007.

[160] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[161] B. Dowden, O. De Silva, and W. Huang, "Object Classification via Semantic Segmentation for Icebreakers," in *Newfoundland Electrical and Computer Engineering Conference (NECEC)*, (St. John's, Newfoundland and Labrador), IEEE Newfoundland and Labrador Section, Nov. 2019.

[162] B. Dowden, O. De Silva, and W. Huang, "Sea Ice Image Semantic Segmentation Using Deep Neural Networks," in *Global Oceans 2020: Singapore U.S. Gulf Coast*, pp. 1–5, Oct. 2020.

[163] B. Dowden, O. De Silva, W. Huang, and D. Oldford, "Sea Ice Classification via Deep Neural Network Semantic Segmentation," *IEEE Sensors Journal*, vol. 21, pp. 11879–11888, May 2021.

[164] B. Dowden, O. De Silva, and W. Huang, "Right Quaternion Parameterization for CubeSat Attitude Determination," in *Newfoundland Electrical and Computer Engineering Conference (NECEC)*, (St. John's, Newfoundland and Labrador), IEEE Newfoundland and Labrador Section, Nov. 2020.