The Pennsylvania State University

The Graduate School

EGO-MOTION ESTIMATION FROM DOPPLER AND SPATIAL DATA IN SONAR, RADAR, OR CAMERA IMAGES

A Dissertation in

Mechanical Engineering

by

Christopher D. Monaco

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

December 2019

The dissertation of Christopher D. Monaco was reviewed and approved* by the following:

Sean N. Brennan
Professor, Department of Mechanical Engineering
Dissertation Advisor, Committee Chair

Hosam K. Fathy
Bryant Early Career Professor, Department of Mechanical Engineering

Jack W. Langelaan
Associate Professor, Department of Aerospace Engineering

Russell C. Burkhardt
Assistant Research Professor, Graduate Program in Acoustics

Kurt A. Hacker
Systems and Emerging Capabilities Division Head, Applied Research Laboratory
Special Member

Karen A. Thole
Head of the Department of Mechanical Engineering

*Signatures are on file in the Graduate School

ABSTRACT

This dissertation focuses on the development of novel ego-motion estimation algorithms for automotive and underwater vehicles. Ego-motion estimation analyzes the perceived motion of the environment from onboard sensor(s) to estimate the vehicle's own motion within that environment, and thus ego-motion estimates are a critical asset for vehicle localization frameworks. Specifically, this dissertation focuses on utilizing RADAR or SONAR sensors with a particular emphasis on their Doppler measurements. This dissertation is comprised of several peer-reviewed research contributions. First, it presents background information and foundational research before presenting three novel ego-motion estimation algorithms: one reliant on measurements from a single RADAR, another reliant on monocular camera images and measurements from a single RADAR, and the last reliant on measurements from a 2D Forward Looking SONAR. All three algorithms rely on the same premise: decoupling ego-motion estimation to utilize different measurement domains for their respective strengths. Specifically, they utilize Doppler and spatial measurements for translational and rotational ego-motion estimation, respectively. This methodology has been shown to yield benefits in accuracy and computational cost. The presented novel algorithms have the potential to improve the localization accuracy of autonomous vehicles, particularly in denied low-visibility environments where an *a priori* map and/or landmarks are nonexistent.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

"All our cognition starts from the senses, goes from there to the understanding, and ends with reason, beyond which there is nothing higher..."

– Immanuel Kant, *The Critique of Pure Reason*

Chapter 1

Introduction

1.1 Motivation

Society is increasingly utilizing autonomous vehicles to handle our most unsanitary, dangerous, and/or mundane tasks, and these autonomous vehicles fundamentally depend on fast and accurate localization. Many of their actions are localization-dependent, including but not limited to: reaching a set destination or waypoints, following a set trajectory, and/or knowing location-based operational constraints [3]. Thus, localization estimates that are delayed and/or inaccurate could trigger undesirable and/or unsafe actuation.

There are two main types of localization techniques: global and local. Global techniques localize the vehicle at a set pose (position and orientation) in space. Many vehicles triangulate their position by analyzing signals sent from transmitters with known positions; examples include the Global Navigation Satellite System (GNSS) and underwater acoustic positioning systems. Furthermore, vehicles can estimate their orientation via magnetic sensors that measure their alignment with the Earth's magnetic field. Alternatively, vehicles can estimate their full pose by matching observed landmarks with those recorded in an *a priori* map. While these techniques have enabled the operation of many automated vehicles, there are also many applications in which these global techniques are not sufficient. For instance, the Global Navigation Satellite System cannot provide the measurement accuracy/frequency required for many autonomous vehicles [4]. In addition, magnetic heading sensor inaccuracies have been shown to be a principal source of navigation error [5, 6]. Furthermore, transmitter networks are expensive to deploy, their signals are blocked by many physical mediums, and they can be spoofed or jammed by an adversary. Similarly, map generation and map matching pose substantial hardware, data compression, and algorithmic challenges, many of which are active areas of research today. Landmark-based localization is especially challenging for areas that are previously unexplored, insufficiently mapped, constantly changing, and/or lack distinct features to serve as landmarks [3]. Consequently, unfortunately, there are many situations where global techniques are not sufficient.

Conversely, local techniques approach localization in a fundamentally different way. Local techniques (also known as dead-reckoning navigation) depend on integrating estimates of the vehicle's motion to estimate the vehicle's pose *relative* to a prior pose. For example, "odometry" is very commonly utilized. Odometry measures the rotation of the vehicle's locomotion actuators in a medium (e.g. wheels on a surface) to estimate the vehicle's velocities. Furthermore, many vehicles rely on an Inertial Navigation System (INS). An INS depends on linear acceleration and rotational velocity measurements from an onboard Inertial Measurement Unit (IMU). Similarly, these estimates can be integrated to calculate relative pose. While these techniques have enabled decades of successful automated vehicles, they too are insufficient for many applications. For ground vehicles, wheel odometry is particularly vulnerable to "slip" on friction-limited or granular surfaces, corrupting their velocity estimates [7]. For underwater vehicles, many techniques rely on measuring the vehicle's velocities relative to the water column, thus, their navigation estimate can be easily corrupted by unknown water currents [8, 9]. IMU measurements are vulnerable to walking biases and random noise, corrupting their measurements as well. Lastly, all local techniques share the same shortcoming: even small ego-motion errors can cause pose errors to grow beyond acceptable limits within a relatively short period of

time [10].

For these reasons, a truly robust autonomous vehicle simultaneously deploys many of the techniques previously described. This offers several levels of redundancy when one or more systems falter. Furthermore, localization frameworks often rely on velocity estimates to impose smoothness constraints and for fast *a priori* pose estimates [11, 12]. But perhaps most importantly, a sensor fusion framework can integrate several measurement sources to generate a holistic optimal estimate despite any measurement's faults in isolation. These frameworks offer a powerful ability: even if a measurement's uncertainty is high, as long as its uncertainty is known, it will *decrease* the state estimate uncertainty [13]. This is particularly important for vehicles since pose uncertainty grows with time between global position locks. So, additional measurements are critical for restraining this growth. This extends the time in which the vehicle has a valid pose estimate, increasingly the probability that a vehicle will soon localize itself globally and/or complete its assigned task.

It is due to the pose uncertainty's growth with time that fast and accurate localization depends on fast and accurate measurements. For this goal, ego-motion estimation offers a promising solution. Ego-motion estimation analyzes the perceived motion of the environment from an onboard sensor to estimate the vehicle's own motion within that environment (i.e. the ego-vehicle's ego-motion) [14]. Ego-motion estimation is another local technique that is complementary to other techniques because ego-motion estimation is not susceptible to the same error sources. Furthermore, unlike map-based methods, ego-motion estimation does not require *a priori* information. Unfortunately, however, like other local techniques, ego-motion estimation's resulting pose uncertainty grows without bound in isolation.

In addition, there is a extraordinarily active field of research on Simulation Localization and Mapping (SLAM) methods that, as the name suggests, simultaneously build a map online and localize within it. SLAM is very powerful because it also does not require *a priori* information but is designed to prevent unbounded error growth. Ego-motion estimation is intimately related to SLAM but differs due to different trade-off priorities. SLAM aims to create a globally consistent trajectory and map. To do so, SLAM must keep track of (and correct) many map landmarks and pose estimates. Ego-motion estimation, on the other hand, prioritizes real-time performance. To do so, it only tracks features long enough to estimate motion, but does not generate a global map. This avoids the costly creation, storage, matching, and correction of map landmarks. Furthermore, as will be discussed later, ego-motion estimation assumes that points/features cannot move far between measurements. Consequently, these methods are typically more robust/tolerant to the information offered by the environment than what could feasibly serve as a map landmark [15]. Fortunately, these characteristics make ego-motion estimation and SLAM very complementary to each other. For example, ego-motion estimation can provide fast and accurate relative pose estimates for SLAM's front-end processes, improving SLAM's slower back-end optimization convergence [16].

Due to their complementary nature, the line between ego-motion estimation and SLAM is often blurred. For example, like SLAM, many modern ego-motion estimation techniques aim to improve their trajectory's consistency. To do so, they analyze a sliding window of previous measurements via a process known as windowed bundle adjustment (BA). As the window grows, like SLAM, consistency is increasingly favored over performance. This closely emulates SLAM's pose refinements but does not include SLAM's creation of a globally consistent map. Therefore, ego-motion estimation variants can be considered to reside on a spectrum that is characterized on one end by SLAM. At this end, SLAM's global consistency capacity can be attributed to its ability to perform loop closures, i.e. smoothly fuse path trajectories when the same area is revisited. The trajectory correction that occurs after a loop closure is detected is the reason why SLAM can prevent unbounded error growth. Consequently, for trajectories that do not include loop closures, ego-motion estimation may be a more appealing route than a more complex and computationally expensive SLAM-based solution [17].

Because of these promising attributes, ego-motion estimation is particularly valuable for the applications that comprise this research's main focus: autonomous automotive vehicles and autonomous underwater vehicles (AUVs). Autonomous automotive vehicles are being designed to operate in dense urban areas in which buildings, bridges, and/or tunnels block Global Positioning System (GPS) signals [4]. While the autonomous vehicles currently in development are extremely reliant on maps, there are many roads in which map landmarks are sparse or non-existent. It is in these areas in which ego-motion estimation is

one of the only viable solutions between global position locks. On the other hand, underwater vehicles present one of the most challenging localization problems in the autonomous vehicles field. Strong water currents, hydrodynamic uncertainties, and the vehicle's six degrees of freedom quickly degrade odometry-based estimates. In addition, almost all electromagnetic signals are attenuated within a short distance underwater. GPS signals cannot penetrate water and long-baseline (LBL) acoustic networks are economically and logistically impractical to deploy for most missions. Signal attenuation also prevents the use of most sensors (e.g. cameras, LIDAR) for even moderate distances from solid objects [18]. Furthermore, underwater environments typically contain less distinct and trackable features than the environments for ground vehicles.

Combined, these challenges served as this dissertation's motivation. Therefore, this dissertation's research focused on developing novel ego-motion estimation algorithms for autonomous automotive and underwater vehicles. Due to their analogous nature, this research was based on RADAR and SONAR sensors, respectively. The automotive field is already very crowded with camera- and LIDAR-based solutions. However, by focusing on RADAR instead, this research aimed to contribute to an underdeveloped area of research. Specifically, it was motivated to more fully leverage RADAR's characteristic strengths: the ability to measure Doppler shift and robustness to inclement weather [19, 20]. For underwater vehicles, ego-motion estimation research is fairly sparse.[1] This is most likely because AUV perception relies almost exclusively on SONAR since only acoustic signals can propagate far through water. Therefore, this research was motivated to improve the relatively few algorithms available that aid navigation in some of the most challenging environments. Lastly, by focusing on two very different vehicle domains, this research aimed to leverage its cross-disciplinary scope to share insights between their respective fields and to gain a more comprehensive understanding of ego-motion estimation as a whole. The resulting holistic insights are discussed in detail in Chapter 8.

1.2 Research Goals

This research had the following high-level goal: **develop novel real-time ego-motion estimation algorithms based on RADAR or SONAR measurements**. More specifically, this goal was decomposed into the following three sub-goals:

1. *Improve accuracy and/or reduce computational cost.* The rationale of developing a novel ego-motion estimation algorithm is to yield advances in accuracy and/or computational cost. However, unfortunately, advances in one often come at the cost of the other. Thus, it more useful to consider an algorithm's fictitious "benefit-to-cost" ratio of accuracy and computational cost. Therefore, this research aimed to develop novel algorithms with better "benefit-to-cost" ratios than their predecessors.

   It is important to note that computational cost should not be judged simply by its ability to beat the real-time operation threshold and/or the sensor's update rate. Robust autonomous vehicles must perform a multitude of tasks; only one of which is ego-motion estimation. Consequently, computational cost reductions should always be pursued as they advance the capabilities of the entire system, yet balanced against algorithm performance.

2. *Promote practical implementations, i.e. emphasize robustness to a wide variety of measurements and environments.* This research strongly prioritized designing its algorithms so that they can be easily and widely implemented in practice. Due to this philosophy, the algorithm development process focused on minimizing sensor measurement and/or environmental assumptions.

   Most importantly, this research aimed to develop algorithms that were robust to outliers, specifically those caused by dynamic agents in the environment. Furthermore, there was a particular focus on the algorithms' inherent ability to handle low resolution measurements and/or relatively "featureless" environments. This promotes the algorithms' applicability, particularly in situations where other localization solutions can fail.

[1] Due to the rarity of large dynamic agents in an underwater vehicle's environment, ego-motion estimation is typically just called "motion estimation" in underwater vehicle literature.

Moreover, this research aimed to produce "plug-and-play" software modules that run in real-time without *a priori* knowledge of the environment. Thus, to maintain robustness and adaptability, there was a specific focus to avoid hard-coded parameters and thresholds whenever possible. If parameters were required, there was a focus to calculate them based on sensor measurements and/or the sensor's intrinsic parameters. Lastly, this research only considered methods that could provide their estimates' corresponding uncertainties. This is critical for the integrating its estimates within an overarching sensor fusion framework, the end goal of implementing these algorithms in practice.

Robustness often comes at the expense of accuracy and computational cost. However, for this research, robustness was generally deemed to be more important.

3. *Validate algorithm accuracy, computational costs, and robustness using real measurements and/or high-fidelity simulation data.* This research aimed to validate its algorithms as thoroughly as possible given the available time and resources. There was a specific emphasis to use real measurements, or, at the very least, high-fidelity simulation data, to instill confidence that the developed algorithms can handle the nuances present in real-world applications. This validation was critical for evaluating whether the aforementioned accuracy, computational cost, and robustness sub-goals were met.

   The author was unaware of any RADAR or SONAR datasets that were suitable to validate this research's algorithms. Fortunately, however, the author instead had access to platforms/frameworks that were able to support algorithm validation. The author was a member of two research groups that largely influenced this research's automotive and underwater vehicle focus. Specifically, the author was a member of the Penn State Intelligent Vehicles and Systems Group (IVSG) and a Graduate Research Assistant at the Penn State Applied Research Laboratory (ARL), respectively.[2] Thus, IVSG provided the instrumented semi-truck that supported the validation of this research's RADAR- and camera-based algorithms. ARL provided the high-fidelity simulation framework that supported the validation of this research's SONAR-based algorithms. Consequently, this research partially focused on developing the aforementioned platforms/frameworks so that they can sufficiently support this validation sub-goal.

1.3 Outline of the Following Chapters

The following chapters roughly follow the natural progression of this dissertation's research. Specifically, each chapter's contributions were inspired by the insights of the one preceding it. First, Chapter 2 provides a background on current Doppler- and spatial-based ego-motion estimation techniques. Then, the following five chapters are a collection of the five peer-reviewed contributions to the field that resulted from this research. These chapters were specifically formatted so that the reader does not need to read the entire dissertation to extract specific insights; each chapter can be read quickly in isolation.

Specifically, Chapter 3 discusses the effects of Doppler Velocity Log placement on the navigation accuracy of underwater vehicles. This chapter finds that, in stark contrast to their standard placement, almost all practical AUVs could benefit from the Doppler Velocity Log being mounted far from the vehicle's reference frame to minimize navigation errors. Chapter 4 discusses the motion limitations imposed by the range flow constraint for featureless image-based ego-motion estimation techniques. This chapter derives the operational limits of these techniques to conclude that they are ill-suited for even moderately fast vehicles. Chapter 5 presents RADARODO, a novel RADAR-based ego-motion estimation algorithm. This chapter demonstrates that, compared to related techniques, RADARODO has a significantly lower computational cost and yields accurate estimates that are better suited for integration within a "tightly-coupled" sensor fusion framework. Chapter 6 presents MonoRAD, a novel ego-motion estimation algorithm that uniquely utilizes monocular camera images and measurements from a single RADAR. This chapter demonstrates that MonoRAD is more accurate or comparable to GNSS Doppler and high-precision IMU measurements, respectively. Chapter 7

[2]This research was largely funded by the Penn State Applied Research Laboratory.

presents SONARODO, a novel SONAR-based ego-motion estimation algorithm. This chapter demonstrates that SONARODO offers accuracy and computational cost advantages over its related methods.

However, this dissertation's chapters are most insightful when they are viewed holistically. To demonstrate this, the relationships between chapters are visualized in Figure 1.1. Chapter 3's study of Doppler Velocity Log placement and Chapter's 4's study of the Range Flow Constraint focus on current Doppler- and spatial-based ego-motion estimation techniques, respectively. They collectively demonstrate that, while Doppler-based techniques are very effective, they still have limitations that necessitate including spatial-based techniques. Yet, many spatial-based techniques are ill-suited for the RADAR and SONAR measurements of automotive and underwater vehicles. Consequently, as shown in Figure 1.1, these insights served as the foundation for the novel RADARODO, MonoRAD, and SONARODO algorithms in Chapters 5, 6, and 7, respectively.

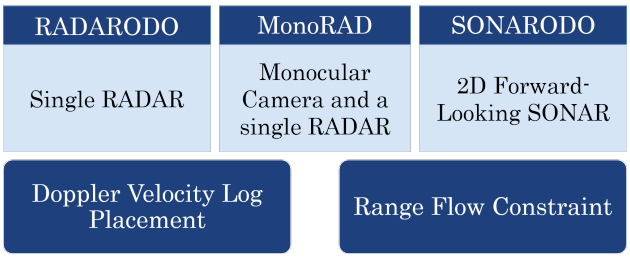


Figure 1.1: Visualization of the relationships between this dissertation's peer-reviewed research contributions. Specifically, the research presented in Chapters 3 and 4 served as the foundation for the novel ego-motion estimation algorithms presented in Chapters 5, 6, and 7.

Thus, all three of these novel algorithms rely on the same premise: decoupling ego-motion estimation to utilize different measurement domains for their respective strengths. Specifically, they utilize Doppler and spatial measurements for translational and rotational ego-motion estimation, respectively. While each algorithm has nuances specific to its sensor suite or vehicle domain, they all largely follow the high-level process visualized in Figure 1.2. As the figure illustrates, they first estimate their sensor's translational motion from its Doppler measurements. Then, they remap its spatial measurements according to the translation estimate so that only rotational differences remain. Finally, they estimate their sensor's rotation from the remapped spatial data.

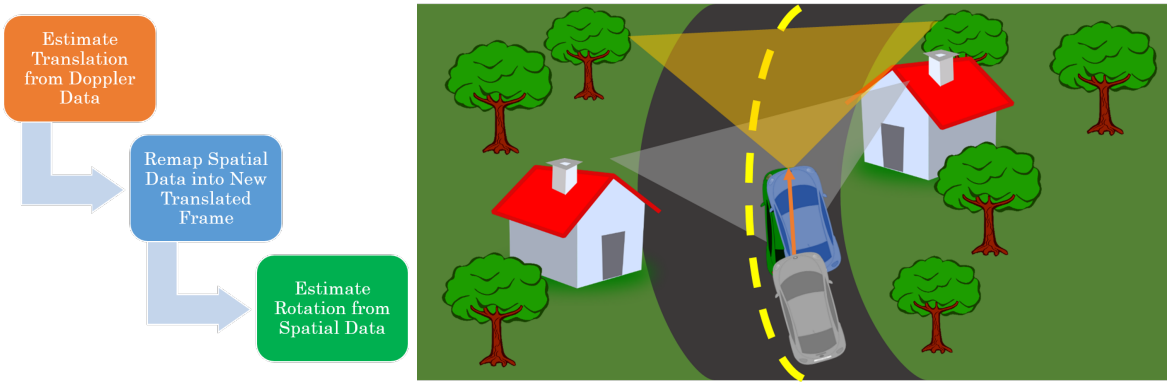


Figure 1.2: Visualization of the RADARODO, MonoRAD, and SONARODO algorithms' shared high-level sub-processes. The sub-processes (left) are visualized by their corresponding colors in the illustration (right).

This dissertation research's natural progression is mostly implied throughout the aforementioned chapters. However, Chapter 8 concludes the dissertation by explicitly discussing its research holistically in order to

present its collective findings.

Chapter 2

Ego-Motion Estimation

This chapter provides an overview of current ego-motion estimation techniques. Since most ego-motion estimation algorithms were designed for camera and/or LIDAR measurements, this chapter is largely focused on those methods. However, current RADAR- or SONAR-based techniques are often close adaptations of the more accessible camera- or LIDAR-based approaches. Therefore, this background serves as a critical foundation for all subsequent chapters.

Ego-motion estimation analyzes the perceived motion of the environment from an onboard sensor to estimate the vehicle's own motion within that environment (i.e. the ego-vehicle's ego-motion) [14]. From the sensor's point-of-view, its ego-motion is equal but opposite to the perceived motion of its environment. Ego-motion estimation techniques can be broadly classified by whether they utilize spatial or Doppler measurements.

2.1 Spatial Techniques

Spatial techniques analyze how the environment appears to move in space between consecutive measurements. The inverse of the environment's measured pose change is then the sensor's pose change within that environment. Thus, dividing this by the time between measurements yields the sensor's ego-motion.

In general, the environment's pose change is estimated via a process called "registration" or "alignment." Registration estimates the transformation required so that transforming one measurement set minimizes its cumulative differences with another. In other words, registration aligns the measurements. This transformation is measurement-dependent, but corresponds to the environment's pose change. Thus, the transformation with the best alignment yields the estimated pose change. The aforementioned differences are also measurement-dependent; they can be differences in pose, positions, image locations, or pixel intensities. To calculate these differences, the correspondences between the consecutive measurements' data points must be determined. Essentially, a correspondence should be established if two data points likely correspond to the same point in the environment, albeit measured at different times and poses. These correspondences are non-trivial, in fact, they lie at the crux of many spatial ego-motion estimation techniques.

For this dissertation, spatial ego-motion estimation techniques will be classified as either *correspondence-based* or *correspondence-free*. Correspondence-based methods are the most intuitive; they utilize some defining attribute or characteristic to establish unique correspondences. Conversely, *correspondence-free* methods do not establish unique correspondences. While some of these methods establish some form of correspondence to calculate measurement differences, they are merely temporary because they continually change throughout the estimation process.[1]

[1]Spatial ego-motion estimation techniques are technically classified by whether they are *dense or sparse* and *direct or indirect*. As pointed out by Engel et al., "the distinction between dense and sparse is not synonymous to direct and indirect – in fact, all four combinations exist." Sparse methods analyze a select subset of the data (e.g. extracted features or a subset of points) while dense methods analyze the data as a whole. Indirect methods create and analyze an intermediate form of the data (e.g. extracting/matching features) while direct methods analyze the sensor measurements themselves [21]. For clarity and brevity, this dissertation instead classified methods as being *correspondence-based* or *correspondence-free*.

2.1.1 Correspondence-based

Correspondence-based techniques establish unique correspondences. To do so, they typically extract a sparse set of distinct and consistent "features" to match or track in the next measurement. However, features can only be extracted if the environment has a sufficient "texture." Similarly, features can only be extracted if the sensor can capture this texture. Thus, correspondence-based techniques often require measurements from high-resolution cameras, LIDARs, SONARs, or RADARs.

Camera-based (Visual Odometry)

The notion to use a stream of camera images to estimate ego-motion was first proposed by Moravec in the 1980s [22]. However, it wasn't until 2004 that Nister et al. provided the first robust real-time implementation and named the process "Visual Odometry" (VO) [23]. This landmark paper sparked the creation of a large, diverse, and mature field of research. Today, Visual Odometry techniques efficiently and accurately estimate ego-motion with a drift rate less than conventional odometry for a wide variety of sensors, vehicles, and environments [17]. Unlike many other techniques, VO offers real-time performance with typical measurement rates from 20 - 40 Hz [24–30].

While VO variants can be correspondence-based or correspondence-free, correspondence-based VO methods are by far a more robust and mature field of research. They depend on extracting image features based on their pixel intensity (i.e. brightness) information. While many feature extraction methods exist today, those based on "corner" features are the most popular. This can be mainly attributed to their low position ambiguity, low computational cost, and the "corner-richness" of many environments [22, 31]. Lines are another often utilized feature [12, 14, 32].

Feature correspondences can then be established via matching or tracking. Feature matching depends on creating and matching feature "descriptors" [33,34]. Since feature matching depends on an exhaustive search, it can handle large pose changes for methods such as Visual Simultaneous Localization and Mapping (V-SLAM) [28, 29]. However, ego-motion estimation has the advantage in that its features cannot feasibly move or change much between images. Therefore, its search radius and consistency requirements are much more relaxed, enabling efficient feature tracking instead [32]. Feature tracking analyzes local pixel intensity changes to capture optical flow, i.e. how a moving feature "flows" from one image location to another [31, 35–37]. For instance, the optical flow vectors in Figure 2.1 were critical for estimating its camera's ego-motion.

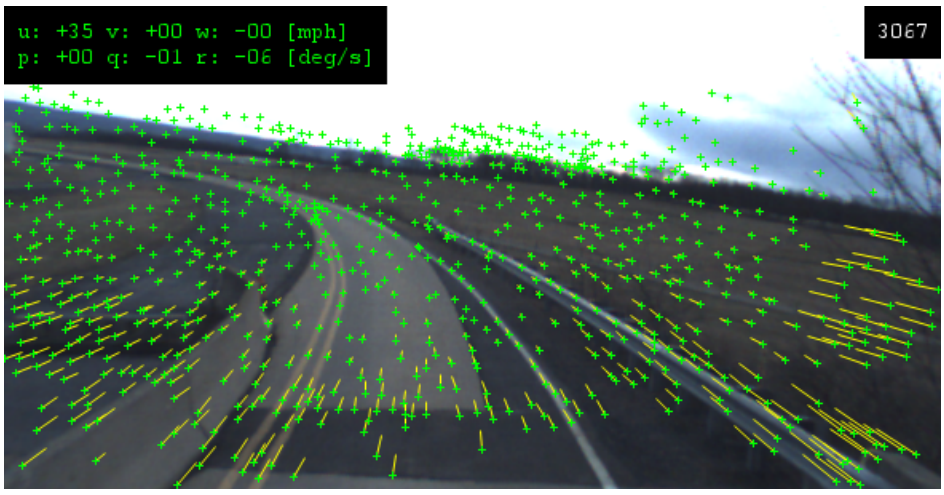


Figure 2.1: Image captured from a moving automotive vehicle's onboard camera. Extracted corner features are shown in green, their optical flow vectors are visualized in yellow.

Since monocular cameras are essentially bearing sensors, they are unable to directly measure the depths

of points in their environment. While bearing information alone can provide rotational ego-motion estimates, their points' depths are required for the correct *scale* of translational ego-motion estimates. Fortunately, there are several VO variants that can estimate both depth and translational ego-motion. Some variants depend on a stereo camera [7], some depend on assumptions about the environment [38,39], some depend on integrating IMU measurements [40–42], and others depend on matching features over many frames [28,29,43]. They all have different strengths and weaknesses when it comes to their accuracy, computational efficiency, and their ability to reject outliers and dynamic agents in their environment [17].

LIDAR-based

LIDAR-based methods depend on registering their 3D point cloud measurements. Unlike image-based methods, point clouds do not have the associated visual information needed to extract distinct image features. Therefore, features must be extracted based on purely geometric information. For instance, some of the most promising 3D feature descriptors extract consistent and distinct 3D planes, edges, corners, cylinders, or spheres [44,45]. Then, 3D feature matching can proceed in a manner very similar to its image counterpart: features are matched to establish correspondences.

Unfortunately, 3D feature matching has a greater computational cost than its 2D feature counterpart due to the expense of extracting and matching 3D features. For instance, this methodology was a sub-process in the popular Lidar Odometry and Mapping (LOAM) algorithm, where it could only execute around 10 Hz [46]. Furthermore, 3D scenes typically don't have as much "texture" as images. In other words, 3D environments are often too smooth or planar to generate enough distinct geometric features, preventing and/or limiting ego-motion estimation. This issue is greatly exacerbated by large scenes that are captured by sensors with poor angular resolutions [47].

SONAR- or RADAR-based

SONAR measurements can come in several forms. Like LIDAR, SONAR can directly measure range. Therefore, 3D SONARs often provide point cloud measurements. However, 2D SONARs are much more popular. These sensors can measure range and azimuth, but cannot determine their points'/targets' depression angles. Consequently, like cameras, their measurements are provided as images and often referred to as "SONAR images." While the bins in camera images correspond to depression and azimuth angles, the bins in SONAR images correspond to ranges and azimuth angles. Thus, a Range-Azimuth SONAR image is a polar representation of the SONAR's environment. These polar images can also be converted to their Cartesian equivalents.

While the pixel intensities in camera images capture visual information, the intensities in SONAR images represent the signal intensities of their SONAR's echo returns. These intensities are affected by many factors, but are largely determined by their targets' material composition and texture. Consequently, "imaging" SONARs can produce high resolution SONAR images that often resemble camera images. For example, Figure 2.2 shows a SONAR image where a shipwreck is clearly visible.

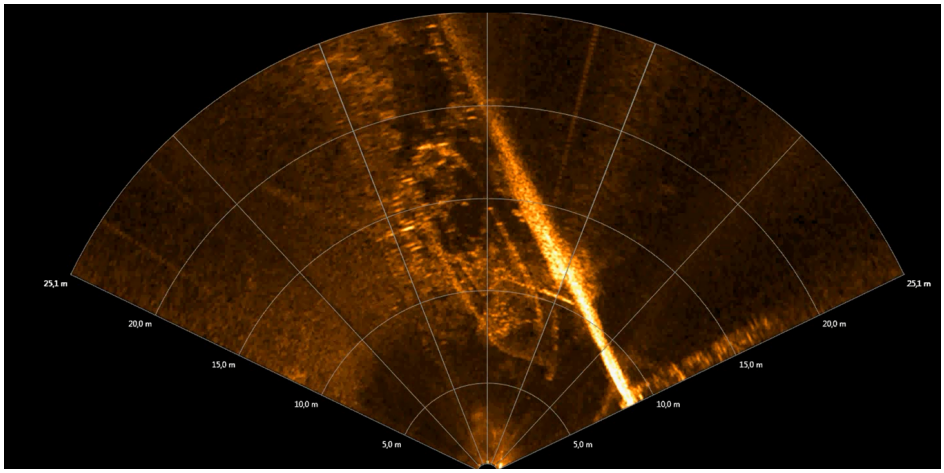


Figure 2.2: A SONAR image from a 2D imaging SONAR where a shipwreck is clearly visible [1].

Therefore, many ego-motion estimation techniques rely on extracting distinct corner or line features from SONAR images [48–52]. However, there features are not reliable for relatively featureless environments and/or SONAR images with poor resolutions and signal-to-noise ratios. In general, many SONAR sensors operate at lower frequencies than "imaging" SONARs to achieve practical maximum ranges. Unfortunately, this often results in significantly reduced range resolutions. Consequently, these feature-based techniques are ill-suited for many applications [53].

Due to their analogous nature, the aforementioned principles also apply to RADAR and RADAR images.

2.1.2 Correspondence-free

Unlike the methods described in Section 2.1.1, correspondence-free methods do not establish unique correspondences between a sparse set of features. Therefore, they are often more robust to relatively featureless environments and measurements from sensors with lower resolutions.

Camera-based (Visual Odometry)

Unlike the methods described in Chapter 2.1.1, correspondence-free spatial ego-motion estimation techniques do not extract a sparse set of image features. Instead, Dense Visual Odometry (DVO) techniques analyze the image as a whole. Most DVO techniques are photometric; they depend on optical camera images. Specifically, they estimate the pose change that would minimize the cumulative intensity differences between two images when one has been "warped" according to that pose change. To do so, they use the image's spatial and temporal derivatives within a rigid "coarse-to-fine" estimation framework. Thus, the iterative warping process continually changes the corresponding pixel intensities [25, 32, 54].

Compared to its feature-based counterparts, Dense Visual Odometry methods are much more robust to their environment's visual texture. Specifically, they are able to extract information from homogeneous planar surfaces that lack features. Furthermore, unlike specialized feature extraction/matching pipelines, they are inherently agnostic to the feature type (corners, lines, etc.) [55]. There are also variants of DVO that rely only on geometric information, i.e. depth images provided by infrared depth cameras [55–57].

LIDAR-based

Unlike the methods in Section 2.1.1, most LIDAR-based algorithms do not extract and match 3D features. For instance, the widespread Iterative Closest Point (ICP) algorithm does not establish unique correspondences. Instead, it establishes *temporary* correspondences using a very simplistic assumption: a point corresponds to the point closest to it in the other point cloud. Then, similar to correspondence-based methods, the

cumulative distance between all corresponding points are minimized to estimate the pose change. However, this simple assumption is violated for even moderate pose changes. Thus, as the name implies, ICP follows an iterative process that continually cycles between re-establishing correspondences and re-registering its points clouds. Using this methodology, ICP often converges to the true pose change.

Unfortunately, this basic premise has many flaws. Fortunately, ICP's active research field has largely mitigated them by offering an abundance of variants and improvements. For example, the use of kd-trees has reduced the computational costs associated with its distance calculations. Furthermore, the use of the point-to-plane metric enables sliding, which is critical for environments with many planar surfaces. Essentially, point-to-plane variants represent one point cloud as a set of planes. Consequently, its point-to-plane metric reduces the distance between a point and its corresponding plane, but also allows it to move freely within that plane. Segal et al. extended this functionality even further by introducing a "plane-to-plane" method called Generalized-ICP (G-ICP) [58]. Today, G-ICP is one of the most widely used and cited ICP variants. Despite its many improvements, ICP variants are still too computationally expensive to offer real-time performance for many vehicles. For example, Fang et al. reported that G-ICP, on average, executes at 11 Hz for a benchmark dataset. Even FastICP, an ICP variant optimized for speed, was reported to execute at 20 Hz [26, 27].

A relatively recent alternative to ICP is the Normal Distribution Transform (NDT). NDT represents point clouds as a combination of normal distributions. In other words, NDT creates piecewise continuous and differentiable probability density functions. Then, it maximizes the points' correlation with these distributions via an iterative non-linear optimization [59, 60]. While NDT was originally developed with a point-to-distribution error metric, it was extended to utilize a distribution-to-distribution error metric instead [61, 62]. Note the strong parallels between NDT's and ICP's error metrics. However, compared to either a point or a plane, NDT's distributions can more robustly represent their underlying points. Furthermore, NDT does not require any point correspondences [60]. Consequently, NDT has been shown to be more robust and faster than ICP [63].

Unfortunately, NDT is not without its shortcomings. NDT divides the captured space into bins or cubes to generate the aforementioned piecewise normal distribution functions. Therefore, its convergence basin and accuracy are very sensitive to the chosen bin size. Furthermore, discontinuities at the bins' edges introduce estimate inaccuracies. Fortunately, these issues can be mitigated with NDT variants that utilize multi-scale k-means clustering [64]. These issues can also be mitigated with similar algorithms that depend on Gaussian Mixture Models (GMMs) [65–67].

SONAR- or RADAR-based

As discussed in Section 2.1.1, 2D SONAR and RADAR image pixels represent thousands of range-azimuth possibilities that are only differentiated by their signal/pixel intensities. Thus, unlike LIDAR, SONAR and RADAR images do not offer distinct point cloud measurements. Therefore, many techniques attempt to extract points from SONAR or RADAR images based on whether they are likely to correspond to prominent features within the environment. Consequently, points are typically extracted from pixels with high signal intensities and/or high signal intensity gradients [68–71]. Given the extracted point cloud, many SONAR- or RADAR-based techniques then deploy the methods discussed in Section 2.1.2. For SONAR, most methods rely on NDT variants or GMM-based approaches [66, 68, 69, 71] while few methods rely on ICP [70]. For RADAR, most methods rely on NDT variants or GMM-based approaches [72–74].

More recently, global search techniques have been directly deployed on the SONAR images in lieu of iterative registration techniques. Specifically, they often utilize Fourier-based registration [53, 75–78].

2.2 Doppler Techniques

RADAR and SONAR have the relatively unique ability to measure the frequency shifts of their reflected signals. Therefore, they can accurately and "instantaneously" measure their targets' relative radial velocities by analyzing their Doppler shifts. Furthermore, unlike many methods in Section 2.1, these measurements

do not rely on correspondences over time. Consequently, there are several Doppler ego-motion estimation techniques that take advantage of this unique measurement domain.

Fortunately, there is a closed form solution for the sensor's translational ego-motion based on these Doppler range rate measurements. This solution merely requires that targets are measured at different angles within the sensor's field-of-view. However, since Doppler measurements can only observe radial motion, rotational ego-motion cannot be directly estimated.

This relationship serves as the foundation for many commercially-available SONAR-based solutions. Specifically, there are SONAR sensors called Doppler Velocity Logs (DVLs) that are specialized to estimate translational ego-motion. In fact, these sensors are currently ubiquitous among Autonomous Underwater Vehicles (AUVs) [79]. For RADAR, there are actually techniques to estimate both the translational *and* rotational ego-motion of automotive vehicles. However, they require either strict motion assumptions [80], multiple distributed RADAR sensors [81], and/or fusion with the RADAR's spatial data [72–74].

As mentioned in Chapter 1, this dissertation's research focused on RADAR and SONAR sensors. Consequently, this research aimed to contribute to the relatively underdeveloped research field of Doppler techniques.

Chapter 3

Doppler Velocity Log Placement Effects on Autonomous Underwater Vehicle Navigation Accuracy

Abstract – Doppler Velocity Logs (DVLs) provide an important correcting measurement to the Inertial Navigation Systems (INSs) of Autonomous Underwater Vehicles (AUVs). DVLs are typically co-located near the Inertial Measurement Unit (IMU) at the AUV's center of buoyancy (CB) to permit a direct correction of the AUV's linear velocities. However, recent research has demonstrated that mounting the DVL at a lever-arm offset from the CB changes the observability of vehicle states. This chapter expands on previous research to explore the effects of DVL placement on AUV navigation accuracy in both loosely- and tightly-coupled frameworks. This chapter finds that, in stark contrast to their standard placement, almost all practical AUVs could benefit from the DVL being mounted far from the CB to minimize navigation errors when integrated into a tightly-coupled framework. These results suggest that the DVL's mounting location should be considered during the design stages of an AUV due to its effect on navigation accuracy.

3.1 Introduction to Doppler Velocity Log Placement Effects

Accurate localization is a particularly difficult challenge for the navigation of autonomous underwater vehicles (AUVs). "Global" techniques localize by utilizing a network of transmitters with known positions, like the Global Navigation Satellite System (GNSS), a Long Baseline (LBL) network, and/or an Ultra-Short Baseline (USBL) network. However, these global techniques are typically not practical for AUVs due to their need for an *a priori* network deployment and/or range limits imposed by the attenuation of high-frequency signals in water. Consequently, most AUVs primarily depend on "relative" localization techniques that estimate pose by integrating sensor measurements. For example, many AUVs depend on propeller-based odometry. Unfortunately, these estimates can be corrupted by propeller slip in water, water cavitation, and/or environmental disturbances due to water currents. Therefore, many AUVs also depend on an Inertial Navigation System (INS) that relies on an Inertial Measurement Unit (IMU). However, these IMU measurements are susceptible to noise and walking biases that drastically corrupt the pose estimate within a relatively short period of time. These errors are compounded by AUV motion's six degrees of freedom [10, 18, 83, 84].

Fortunately, fusing measurements from other sensors can dramatically improve AUV navigation accuracy. For instance, Doppler Velocity Logs (DVLs) can aid vehicle velocity estimates. In "bottom-tracking" mode, a DVL transmits multiple beams of acoustic "pings" downward and then detects their reflections off the

This chapter was originally published in the proceedings for the 2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV) [82].

seafloor. The Doppler shifts (i.e. frequency shifts) of these reflected beams are then used to calculate the speed of the DVL in the Earth's inertial frame [79].

DVLs typically utilize three or four beams to calculate their linear velocities in all three dimensions. When mounted at the vehicle's center of buoyancy (CB), the DVL can directly measure the linear velocities of the vehicle. This explains why DVLs are typically mounted at or near this location [85, 86]. Furthermore, since the IMU is also typically mounted near the CB, the DVL can directly correct INS linear velocity estimates. In fact, this co-location is so pervasive that IMU+DVL sensors are often available as an integrated unit.

However, the co-location of these sensors is not always practical due to mechanical packaging constraints. When a DVL is mounted at a lever-arm offset from the CB, the vehicle's rotational motion is detected by the DVL as linear motion corresponding to that offset. This offset is commonly handled by either a "loosely-coupled" or "tightly-coupled" framework [84, 87]. Both of these frameworks depend on accurate knowledge of the lever-arm offset, which may not always be feasible. Therefore, previous research has typically focused on improving lever-arm estimates for a variety of vehicles and sensors. While these analyses examine the effects that lever-arm offset estimate errors have on corrupting vehicle state estimates, interestingly, they only partially examine purposefully using lever-arm offsets to estimate additional vehicle states. For example, previous research has noted that the lever-arm offset dictates which vehicle states are observable from position and velocity sensors [87–90]. For automobiles, Kellner et al. took advantage of a Doppler RADAR sensor's front mounting location to not only estimate an automobile's longitudinal velocity but also its yaw rate [80]. Yet, these analyses do not extend their results to the effects that lever-arm offsets can have on vehicle navigation accuracy.

This chapter expands upon previous research by analyzing the effects of DVL placement on underwater vehicle navigation accuracy for both loosely- and tightly-coupled frameworks. Unlike related literature, this chapter assumes that the DVL lever-arm offset and alignment is precisely known. Specifically, this chapter seeks to determine the ideal DVL mounting location to maximize navigation accuracy for a given AUV and its desired application. To the author's knowledge, this relationship has not yet been published in literature.

3.2 Relationship between Vehicle Motion and DVL Measurements

Ideally, each beam of a DVL strikes and is reflected off of a static seafloor "target." The relationship between that target's range from the DVL $r_{T/D}$ and its location in the DVL's xyz body-fixed coordinate system[2] $(x_{T/D},\, y_{T/D},\, z_{T/D})$ is shown in Equation 3.1:

$$r_{T/D}^2 = x_{T/D}^2 + y_{T/D}^2 + z_{T/D}^2 \tag{3.1}$$

The perceived motion of these points from the vehicle-mounted DVL should correspond to motion opposite that of the DVL's own ego-motion. Therefore, the perceived velocities of a point in the DVL's xyz coordinate frame $(\dot{x}_{T/D},\, \dot{y}_{T/D},\, \dot{z}_{T/D})$ are related to the vehicle's ego-motion as follows:

$$\begin{pmatrix} \dot{x}_{T/D} \\ \dot{y}_{T/D} \\ \dot{z}_{T/D} \end{pmatrix} = \begin{pmatrix} -v_{x,D} - z_{T/D}\omega_y + y_{T/D}\omega_z \\ -v_{y,D} + z_{T/D}\omega_x - x_{T/D}\omega_z \\ -v_{z,D} - y_{T/D}\omega_x + x_{T/D}\omega_y \end{pmatrix} \tag{3.2}$$

where $v_{x,D}$, $v_{y,D}$, and $v_{z,D}$ represent the DVL's linear velocities and ω_x, ω_y, and ω_z represent the DVL and vehicle's shared angular velocities in the xyz body-fixed coordinate system. Combining Equation 3.2 with the time derivative of Equation 3.1 yields the relationship between a DVL beam's Doppler velocity measurement, the relative location of its seafloor "target," and the vehicle's ego-motion:

$$\dot{r}_{T/D} = -\frac{x_{T/D}}{r_{T/D}} v_{x,D} - \frac{y_{T/D}}{r_{T/D}} v_{y,D} - \frac{z_{T/D}}{r_{T/D}} v_{z,D} \tag{3.3}$$

[2]For simplicity, the vehicle's and DVL's xyz body-fixed coordinate systems will be assumed to be aligned although their origins differ by the lever-arm offset. Subscripts will be used to indicate which coordinate system is being referenced. For clarity, this chapter will consistently use the body-fixed xyz coordinate system. Note that this convention directly corresponds to the typical $uvw - pqr$ vehicle velocity coordinate system.

Interestingly, Equation 3.3 shows that the vehicle's angular velocities are not captured by Doppler velocity measurements. This can be explained by representing the beam's "target" in the spherical coordinate system. The target's perceived motion due to the DVL's rotational motion will reside in the spherical coordinate system's $\theta - \phi$ plane. However, Doppler velocity measurements can only observe motion along the r-axis. Fortunately, while the DVL cannot directly detect rotational motion, it can detect the linear motion produced by the vehicle's rotational motion and the DVL's lever arm offset. This relationship is described in all six degrees of freedom by Equation 3.4:

$$\vec{v}_D = \vec{v}_V + \vec{\Omega} \times \vec{r}_{D/V} \tag{3.4}$$

where $\vec{v}_D$ is the DVL's linear velocities, $\vec{v}_V$ is the vehicle's linear velocities, $\vec{\Omega}$ is a vector of the vehicle's and DVL's shared angular velocities, and $\vec{r}_{D/V}$ is the vector describing the DVL's lever arm offset from origin of the vehicle's body frame (co-located with the IMU at the center of buoyancy). This relationship is expanded with more detail in Equation 3.5:

$$\underbrace{\begin{bmatrix} v_{x,D} \\ v_{y,D} \\ v_{z,D} \end{bmatrix}}_{\vec{z}} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & z_{D/V} & -y_{D/V} \\ 0 & 1 & 0 & -z_{D/V} & 0 & x_{D/V} \\ 0 & 0 & 1 & y_{D/V} & -x_{D/V} & 0 \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} v_{x,V} \\ v_{y,V} \\ v_{z,V} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}}_{\vec{x}} \tag{3.5}$$

where $x_{D/V}$, $y_{D/V}$, and $z_{D/V}$ represent the individual components of the lever-arm offset, $\vec{x}$ is the vehicle state vector, and H is the measurement transformation matrix, and $\vec{z}$ is the DVL measurement vector.

3.3 Vehicle State Estimate Uncertainty Analysis

The common rationale behind placing the DVL at or near the center of buoyancy can be easily explained with Equation 3.5. When $\vec{r}_{D/V} \approx \vec{0}$, the H terms associated with the angular velocities are eliminated, permitting the DVL to directly measure (and correct) the vehicle's linear velocity estimates. Conversely, the angular velocity estimates remain uncorrected. However, when a lever arm offset exists, $\vec{x}$ contains 6 unknown state variables while H contains only 3 equations.[3] This non-unique transformation prohibits direct vehicle state estimates. Fortunately, using a loosely- or tightly-coupled framework to integrate DVL data can re-enable valid state estimates, albeit in different ways. This section will analyze the resulting vehicle state estimate variances for each framework.

To aid this analysis, the AUV's motion will be assumed to be planar for the remainder of this chapter. The author felt that this was best for several reasons. First, eliminating degrees of freedom is critical to clearly explain high-level trends. Secondly, the hydrodynamic geometries of most AUVs restricts the DVL to be mounted only along the longitudinal axis (with a minimal set $z_{D/V}$). As will be shown later, this longitudinal offset greatly influences corrections of yaw rate (the only rotational motion in planar motion). Lastly, the author recognizes the drastically corrupting effects of using an INS to measure pose in all in six degrees of freedom. However, AUVs commonly use pressure sensors, inclinometers, accelerometers, and the vehicle's self-stabilizing buoyancy characteristics to greatly mitigate roll and pitch errors. The AUV will be simulated with a commonly used IMU and DVL: the Honeywell HG1930 IMU [91] and the Rowe Technologies SeaPilot DVL [92][4].

[3] To maintain generality between three beam and four beam DVLs (i.e. Janus configuration DVLs), only the outputted *xyz* velocities will be analyzed in this chapter. Independently analyzing the four beams of Janus configuration DVLs will change the matrices, but since four measurements create an overdetermined system, the overarching conclusions will remain. Analyzing cases of partial DVL dropouts are beyond the scope of this chapter.

[4] The IMU measurements were assumed to be corrupted by zero-mean Gaussian white-noise and bias random walk while the DVL measurements were assumed to be corrupted by zero-mean Gaussian white-noise and scaling factor errors.

One of the greatest assets of INS-aiding sensors is their ability to actively estimate and compensate for IMU biases in a Kalman Filter framework. Therefore, the corresponding IMU biases were augmented to the state vector in the discrete planar motion system matrices in Equation 3.6:

$$\vec{x}_{k+1} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & -\delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & -\delta t & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} v_x \\ v_y \\ \omega_z \\ b_{\dot{v}_x} \\ b_{\dot{v}_y} \\ b_{\omega_z} \end{bmatrix}}_{\vec{x}_k} + \underbrace{\begin{bmatrix} \delta t & 0 & 0 \\ 0 & \delta t & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \omega_z \end{bmatrix}}_{\vec{u}_k} \tag{3.6}$$

where δt is the time between the navigator algorithm's executions[5], $\vec{u}$ is the IMU measurement vector that serves as the system's inputs, and $\vec{x}$ is the augmented planar state vector.

3.3.1 Loosely-Coupled Framework

The popular loosely-coupled (LC) framework handles lever-arm offsets via "lever-arm compensation." Essentially, using lever-arm offset and rotational motion estimates, this framework removes the influence of rotational motion on DVL estimates to generate "virtual" linear velocity measurements. This can then be used to directly correct the vehicle's linear velocity estimates. The planar lever-arm compensation equations are shown in Equation 3.7:

$$\begin{aligned} v_{x,V,virtual} &= v_{x,D} \\ v_{y,V,virtual} &= v_{y,D} - x_{D/V}\hat{\omega}_z^- \end{aligned} \tag{3.7}$$

where $v_{x,V,virtual}$ and $v_{y,V,virtual}$ are the virtual vehicle linear velocities generated by lever-arm compensation and $\hat{\omega}_z^-$ is the *a priori* yaw rate estimate. Note that $v_{y,V,virtual}$ depends on an *a priori* estimate, which itself is error-prone. Consequently, this uncertainty will propagate to the uncertainty of $v_{y,V,virtual}$ as detailed in Equation 3.8:

$$\mathrm{Var}(v_{y,V,virtual}) = \mathrm{Var}(v_{y,D}) + x_{D/V}^2 \mathrm{Var}(\hat{\omega}_z^-) \tag{3.8}$$

The effects of the DVL's longitudinal offset on vehicle state uncertainty can be best explained with Figure 3.1. This figure shows the vehicle state estimate 1σ values after the aforementioned planar AUV, with a loosely-coupled (LC) framework, was simulated moving forward at 1 knot for 100 seconds at varying DVL longitudinal offsets from the CB. Figure 3.1 and Equation 3.7 help explain the rationale behind the DVL's standard placement at/near the CG. As the lever-arm offset increases, $\mathrm{Var}(v_{y,V,virtual})$ increases, increasing the v_y estimate's 1σ value. Conversely, when $x_{D/V} = 0$, these relationships reduce to the DVL directly measuring the vehicle's linear velocities, as expected. For this planar AUV, the v_x 1σ value is the same for all longitudinal offsets. Furthermore, it is noteworthy that this LC framework cannot correct rotational motion estimates.

[5]For simulation simplicity, the navigator was assumed to only execute when a correcting DVL measurement was available.

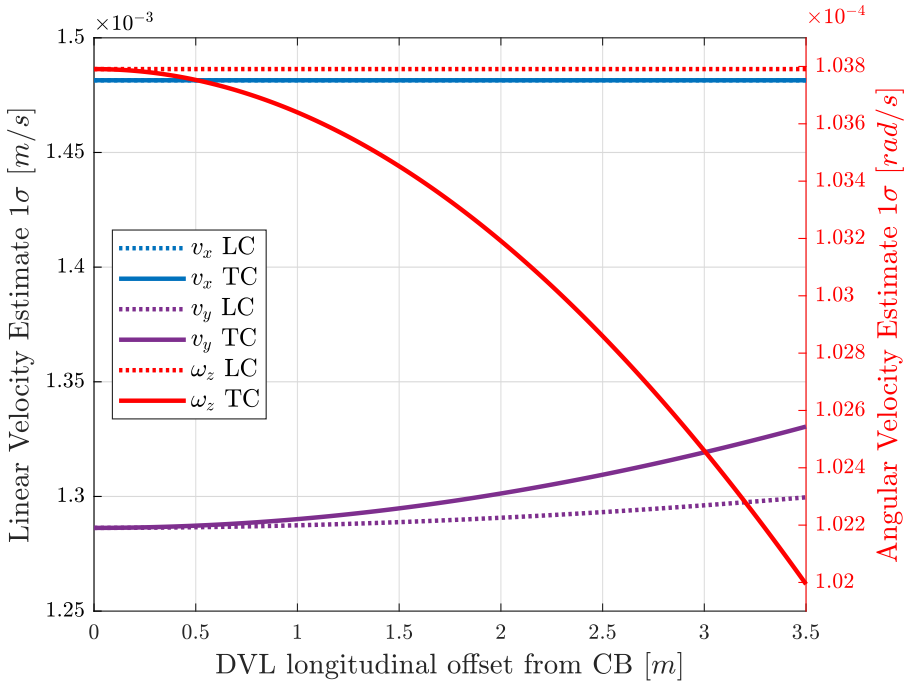


Figure 3.1: Vehicle state estimate 1σ values after a set mission for the loosely-coupled (LC) and tightly-coupled (TC) frameworks at varying DVL longitudinal offsets from the CB.

3.3.2 Tightly-Coupled Framework

In contrast to the loosely-coupled framework, the tightly-coupled (TC) framework uses INS motion estimates to generate "virtual" DVL measurements. Unlike the LC framework which compares real/virtual linear velocities, the TC framework compares real/virtual DVL measurements. Essentially, this TC framework calculates what the DVL measurements should be if the INS estimates are correct. Then, the actual DVL measurements are used to correct their virtual counterparts, thereby correcting their associated vehicle state estimates. Furthermore, unlike the previous LC framework, this permits the correction of both linear *and* rotational motion estimates. Lastly, this framework is particularly robust to cases of partial DVL transducer dropouts [84, 87].

Figure 3.1 is again useful to illustrate the effects of the DVL's longitudinal offset on vehicle state uncertainty. A planar AUV was also simulated for 100 seconds, but this time utilized a tightly-coupled (TC) framework. A planar variant of the H matrix in Equation 3.5 was used to transform the *a priori* state estimates into virtual DVL measurements. This figure exposes a very important characteristic of the tightly-coupled framework: as the longitudinal offset increases, the ω_z 1σ value decreases at the expense of an increasing v_y 1σ value. Essentially, this is because the longitudinal offset adjusts the relative observability between the vehicle's lateral linear velocity and yaw rate. Again, the longitudinal offset has no effect on the v_x 1σ value for this planar AUV.

These simulations permit clear comparisons between the two frameworks. Both frameworks exhibit identical performances for AUVs without a lever-arm offset. Also, both frameworks have v_x estimates that are agnostic to the DVL's longitudinal offset. Furthermore, for both frameworks, the v_y estimate uncertainty increases non-linearly with increasing longitudinal offset. In contrast, the LC framework v_y uncertainty increases at slower rate than the TC framework's. However, the ω_z estimate uncertainty is where the two frameworks truly differ. The LC framework's estimate is barely affected by increasing longitudinal offset while the TC framework estimate's uncertainty exhibits non-linear decay. Essentially, compared to the LC framework, the TC framework sacrifices a greater v_y uncertainty for a reduction in ω_z uncertainty.

3.4 Relative Sensitivity of Navigation Position Accuracy to Vehicle State Errors

Section 3.3 described both the inter- and intra-framework tradeoffs between the lateral velocity v_y and the yaw rate ω_z uncertainty. This begs the question: what is the relative importance of each vehicle state's uncertainty with respect to navigation accuracy? To provide insight, two parallel AUVs were simulated with a planar motion model. Although their ground truth motion was purely forward, one AUV had INS estimates that were *only* corrupted by a lateral accelerometer bias (Accel. Y) while the other was *only* corrupted by a yaw rate bias (Gyro. Z). These constant biases were equivalent to their respective 1σ Bias Repeatability values to accurately capture their relative performance in IMUs. The relative navigation position errors were plotted in Figure 3.2.

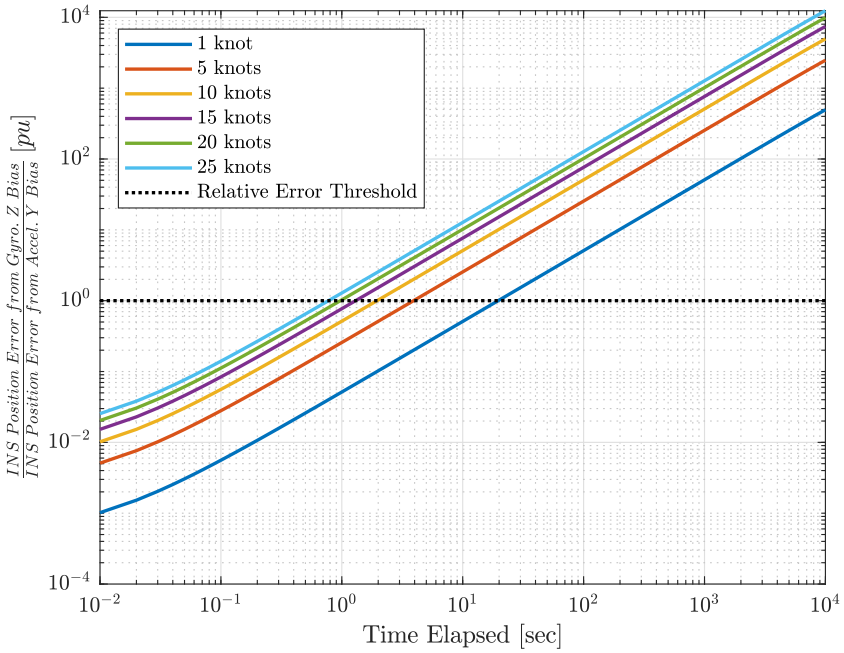


Figure 3.2: Factor of the INS position error due to Gyroscope Z biases over the INS position error due to Accelerometer Y biases versus time for for varying longitudinal speeds and mission durations.

This plot captures several important trends. The y-axis represents the factor of the INS position error due to the Gyro. Z bias over the INS position error due to the Accel. Y bias. Values greater than one (the "Relative Error Threshold") indicate an INS estimate that has been more corrupted by the Gyro. Z bias than its Accel. Y counterpart. These simulations were conducted for varying longitudinal speeds and mission durations. Therefore, the point at which an AUV crosses the Relative Error Threshold provides valuable insight for the relative importance of lateral velocity vs. yaw rate errors. Fast-moving AUVs crossed this threshold quickly in less than a second while slow-moving AUVs still crossed this threshold briskly in less than 20 seconds. After this threshold crossing, for all speeds, the relative error is greater than one with a positive slope. *Combined, this indicates that navigation error is increasingly more sensitive to yaw rate error than lateral velocity error over time for almost all practical AUVs and AUV missions.*

This agrees with the findings of Woodman et al. who stated that, "[i]n practice it is the accuracy of the gyroscopes and not the accelerometers which limit the overall accuracy of most INSs...[d]rift which occurs due to accelerometer noise is only significant in the first few seconds of the algorithm's application" [10].

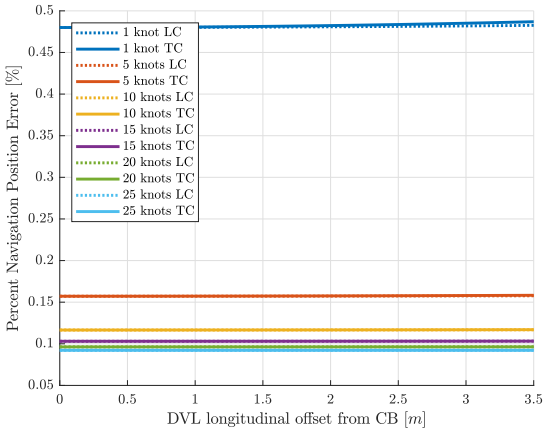
While magnetometers are commonly proposed to mitigate heading errors, fluctuating internal and external magnetic abnormalities may limit the benefits of utilizing this sensor in many environments [83].

3.5 Effects of DVL Placement on Navigation Accuracy

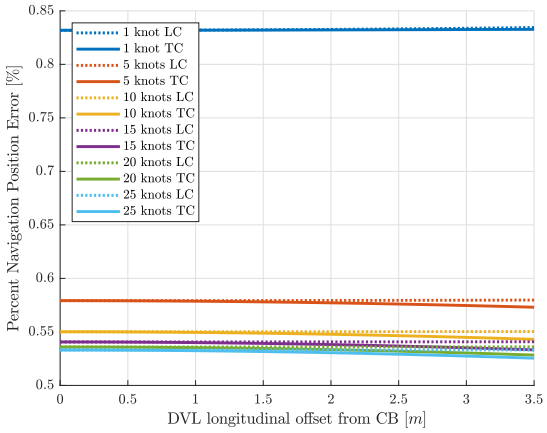
To explore the effects that DVL placement has on navigation accuracy, numerous planar AUV simulations were conducted. The simulated AUVs were split into two main groups: one utilizing loosely-coupled (LC) DVL frameworks and the other utilizing tightly-coupled (TC) frameworks. The ground truth motion for all AUVs was purely forward, however, they moved at widely varying forward speeds (v_x). Furthermore, the AUVs were simulated with DVLs at different longitudinal offsets from the CB. Lastly, the AUVs were simulated with varying mission durations. A kinematic Kalman Filter was used to calculate their state covariance matrices. The Kalman Filter utilized the system matrices, state vector, and input vector defined in Equation 3.6. To calculate global position, a separate navigator algorithm was used to handle the non-linearity of the navigation equations. This navigator integrated body-fixed motion estimates that were *only* corrupted by adding their corresponding 1σ values from that step's *posterior* state estimate covariance matrix. The percent navigation position errors for all simulated AUVs as a function of their DVL's longitudinal offset from the CB was plotted in Figure 3.3.

Many conclusions can be drawn from Figure 3.3. As predicted by Figure 3.1, AUVs with a zero DVL lever-arm offset produced the same navigation accuracy from both the LC and TC frameworks. Also, both frameworks produced near equivalent accuracy for short-duration missions, regardless of lever-arm offset. However, *the TC framework offered an increasing accuracy advantage over the LC framework for increasing DVL longitudinal offsets and/or mission durations.*

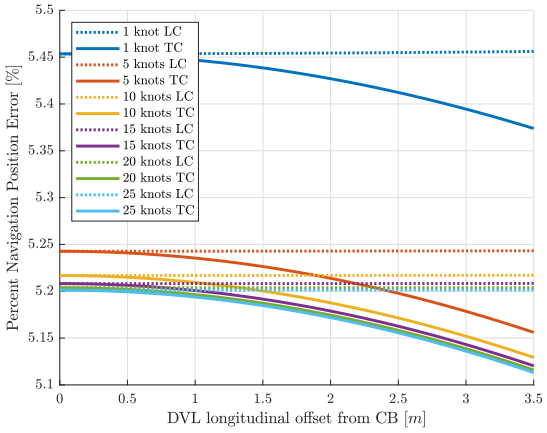
This finding is internally consistent with previous findings. Although it is imperceptible at this scale in Figure 3.3, the LC's navigation error increases slightly with increasing offsets. This can be attributed to the LC lateral velocity's uncertainty increasing with an increasing offset (as seen in Figure 3.1). In contrast, Figure 3.3 shows that, for the vast majority of configurations, the TC's navigation error *decreases* with an increasing offset. Like the LC configuration, the TC's lateral velocity uncertainty increases with increasing offset. However, unlike the LC framework, Figure 3.1 shows that this also corresponds to a decreased yaw rate uncertainty. Consequently, since Figure 3.2 demonstrated that navigation accuracy is more sensitive to yaw rate errors than lateral velocity errors for almost all practical use cases, Figure 3.3 shows that the TC navigation error is reduced for those cases.



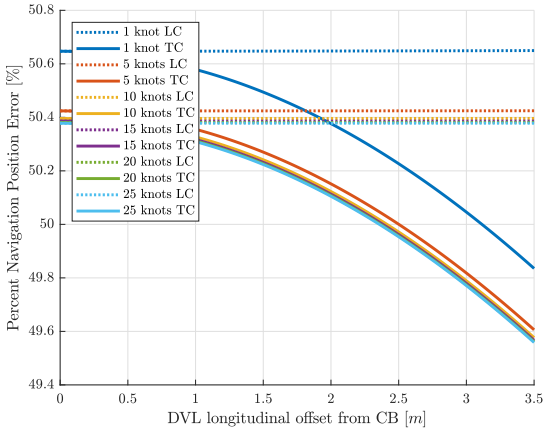
(a) Mission duration of 10 seconds.



(b) Mission duration of 100 seconds (≈ 1.7 min.).



(c) Mission duration of 1,000 seconds (≈ 16.7 min.).



(d) Mission duration of 10,000 seconds (≈ 2.8 hr.).

Figure 3.3: Percent navigation position error for various AUVs as a function of the DVL's longitudinal offset from the CB.

3.6 Design Considerations for DVL Placement

For the reasons previously explained, DVLs are typically placed at or near the AUV's center of buoyancy. In practice, various packaging requirements might render this infeasible. Figure 3.3 showed the navigation position errors that can be expected for these AUVs. However, this figure exposes deeper insights than just navigation accuracy predictions.

There are three main types of lines plotted in Figure 3.3. Some "curve up" (i.e. navigation error increases with increasing longitudinal offset) while others "curve down" (i.e. navigation error decreases with increasing offset). The transition between these are flat lines in which the navigation error remains unchanged. These three line types correspond to various "optimal" DVL mounting locations for minimizing navigation errors. Lines that curve up indicate that the CB is the optimal DVL location. In stark contrast, lines that curve down indicate that the optimal location is as far from the CB as physically possible. Flat lines indicate that DVL placement is inconsequential.

Section 3.5 explained that, *all* of the LC framework's lines in Figure 3.3 curve up as expected. This provides quantitative support for the standard practice of placing the DVL at/near the CB. However, Figure

3.3 shows that the TC framework contains a combination of all three line types. This is important because these "flat line" use cases serve as critical "crossover points." Slight deviations from these crossover points correspond to the optimal DVL location jumping drastically between the CB and a location far from it. Since Section 3.5 demonstrated that the TC framework provides a navigation accuracy advantage over its LC counterpart for almost all practical use cases, this section will explore the crossover points of the TC framework.

These crossover points are determined by an AUV's expected speed and mission duration when they utilize the TC framework. Therefore, additional simulations similar to those described in Section 3.5 were conducted to determine the circumstances in which these crossover points would occur. This crossover line was plotted in Figure 3.4.[6] The optimal DVL placement for AUVs that operate at lower speeds and/or for shorter mission durations than this line is at the standard CB location. Conversely, however, the optimal DVL placement of AUVs that operate at higher speeds and/or for longer mission durations than this line is at a location far from the CB. These results are internally consistent with the previous findings. What may be more surprising, however, is that almost all practical use cases reside on this side of the crossover line (shown shaded in green). This implies that, *in direct contrast to standard DVL mounting practices, for almost all practical AUV use cases, the optimal DVL mounting location is far from the CB when a tightly-coupled framework is utilized.*

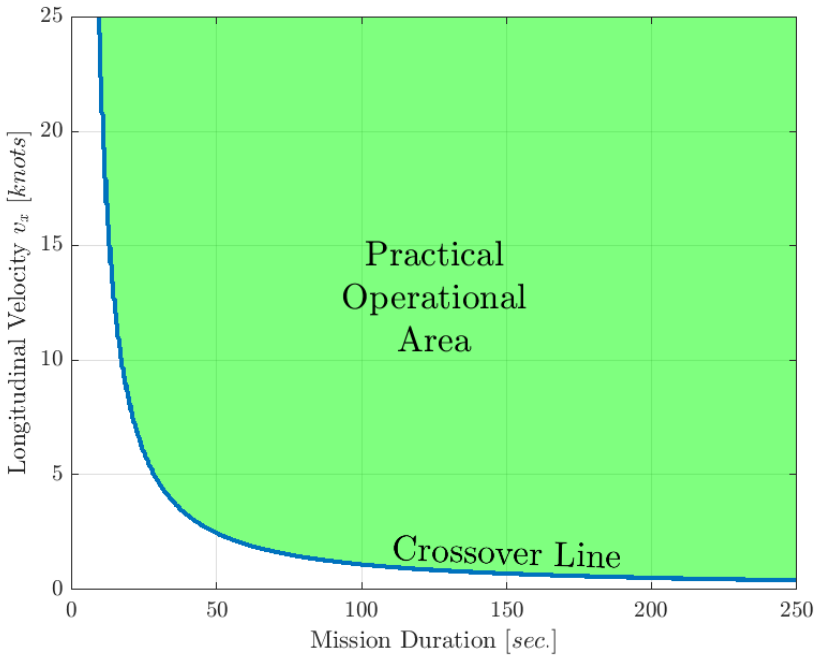


Figure 3.4: Expected AUV speed and mission duration "Crossover Line" that determines optimal DVL placement when a tightly-coupled framework is utilized.

This finding indicates that the DVL's location should be considered during an AUV's design stage due to its effect on the expected navigation accuracy. Specifically, its optimal location may be drastically different from the standard CB location. Furthermore, since it is is irrelevant if the DVL is placed fore or aft of the CB, this conclusion may relax the AUV's mechanical packaging constraints.

[6]The simulations plotted also corrupted the body-frame motion estimates by their estimate's 1σ values. The results for varying corruption values (i.e. varying the factor of σ values added) were also analyzed, however, since they corresponded to negligible (sub-second) differences, they were not plotted.

At this point, however, the scale of the navigation accuracy axes in Figure 3.3 might introduce healthy skepticism. Regardless of the DVL's optimal location, small longitudinal offsets correspond only to marginal changes in navigation accuracy. Therefore, the DVL's mounting location may be inconsequential for small AUVs (or any AUV utilizing a LC framework). In addition, while Figure 3.4 supports the general rule of thumb that the DVL should be mounted far from the CB, Figure 3.4 is perhaps better suited to illustrate the *weight* of this recommendation. For instance, the closer an AUV's expected use case is to the crossover line, the smaller this navigation benefit becomes. Therefore, some AUVs may have competing constraints (e.g. misalignment concerns) that mitigate or outweigh this benefit.

With that said, it is important to note that Figure 3.3 plots navigation position percent error. Consequently, for AUVs that travel long distances (e.g. fast AUVs and/or AUVs with long-duration missions), a marginal increase in navigation position percent error can correspond to an unacceptable position error radius. So, even small improvements to navigation accuracy can cumulatively expand a vehicle's capabilities, particularly when their implementation cost is minimal. Regardless, as the speed and mission duration requirements of an AUV are increased, so is the potential navigational accuracy benefit of placing its DVL as far from the CB as possible.

3.7 Doppler Velocity Log Placement Effects Conclusion

Doppler Velocity Logs are valuable sensors for the navigation of Autonomous Underwater Vehicles; thus, they are widely utilized. DVLs are typically mounted at or near the vehicle's CB. However, this chapter finds that, in stark contrast to their standard placement, almost all practical AUVs could benefit from the DVL being mounted far from the CB to minimize navigation errors when integrated into a tightly-coupled framework.

This chapter's findings are particularly useful during the design stages of a proposed AUV. Specifically, they demonstrate that the expected AUV mission requirements can help recommend an optimal DVL location and illustrate that recommendation's weight. Design studies are then critical for determining how this constraint is precisely weighted relative to other constraints. Overall, this chapter's main contribution is the assertion that the DVL's mounting location should be considered during the design stages of an AUV due to its effect on navigation accuracy.

Chapter 4

Ego-Motion Estimate Corruption Due to Violations of the Range Flow Constraint

Abstract – Visual odometry methods are increasingly being used to estimate a vehicle's ego-motion from range data due to the decreasing cost of range sensors and the impressive speed and accuracy of visual odometry techniques. Dense geometry-based visual odometry methods are fundamentally based on the range flow constraint equation, an equation which depends on the temporal and spatial derivatives of range images. However, these derivatives are calculated with the fundamental assumption that the range flow is magnitude-limited. When scaling this method for faster vehicles, this assumption could be violated, invaliding the range flow constraint equation and thus corrupting the resulting ego-motion estimates. This chapter derives the sensor, motion, environment, and sampling frequency conditions that would mathematically violate the range flow constraint. This information is useful for defining the operational limits of dense geometry-based visual odometry methods.

4.1 Introduction to the Range Flow Constraint

Accurate and fast localization is of paramount importance for many autonomous vehicle subsystems. Consequently, autonomous vehicles typically utilize global methods that triangulate the vehicle's pose relative to set landmarks or transmitters (e.g. GPS). However, for many applications, this framework cannot provide the necessary measurement accuracy/frequency required for autonomous operation [4]. To compensate, local (relative) localization methods are often used to estimate the displacement from a known initial pose. For instance, Inertial Navigation Systems (INS) and/or odometry are used to calculate vehicle pose from integrated measurements. However, these measurements are susceptible to errors that cause the pose estimate to drift, corrupting the estimate within a relatively short period of time [10].

To reduce the drift associated with local localization, Nister et al. developed "visual odometry," a method that uses successive images from a vehicle-mounted camera to estimate that vehicle's motion (ego-motion) [23]. This technique fundamentally depends on "optical flow," the perceived movement of image features in temporally adjacent images. Visual odometry is a welcomed supplement for local localization since it is not susceptible to INS's random-walk bias errors and is more robust to traction-limited environments than conventional odometry. Today, there are many robust techniques that use visual odometry to accurately estimate ego-motion and pose in real-time for a diverse set of vehicles, environments, and cameras [14,17,32].

This chapter was originally published in the proceedings for the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) [93].

While these feature-based visual odometry techniques have been very successful, they fail to utilize image data that does not correspond to features. Therefore, promising "dense" visual odometry (DVO) techniques have been explored [25–27, 54, 55, 94–97]. Instead of extracting and tracking specific features to estimate ego-motion, dense approaches analyze the image as a whole. Unlike feature-based methods, features are not tracked to establish correspondence. Instead, dense approaches rely on solving the *optical flow constraint equation*, which describes how ego-motion changes the image over time based on the image's temporal and spatial derivatives [55].

However, relatively few techniques have explored the use of ego-motion estimation using geometric data *only* [26, 27]. This is important since many vehicles operate in environments in which photometric data are not available. One of the most common techniques for geometry-based ego-motion estimation is Iterative Closest Point (ICP). While ICP-based techniques have produced impressive results, they typically suffer from a major downfall: the computational cost associated with directly analyzing point clouds [17, 26, 27]. This expense causes ICP pose estimates to be produced at a drastically lower frequency than visual odometry. At high vehicle speeds, odometry and INS errors can lead to large localization errors within fractions of a second. If ICP cannot correct these errors fast enough, it may provide little benefit.

To improve the execution frequency of geometry-based methods, many began to analyze range/depth[2] images instead [26, 27]. Since the overarching process (using sequential images to estimate ego-motion) and techniques are similar, the range-based methods are often also referred to as visual odometry. Analogous to how dense photometric visual odometry (DPVO) methods are based on the optical flow constraint equation, dense geometry-based visual odometry (DGVO) techniques rely on solving the *range flow constraint equation* [55–57, 98–103]. This equation describes how the range image should change with ego-motion over time. *Range flow* is analogous to optical flow with a slight difference; it describes how the projected *range* of a point flows from one pixel location to another[3] [56]. The range flow constraint equation is used for "energy minimization": finding the ego-motion that, when used to warp one range image, would minimize its difference from a temporally adjacent range image [57].

Many DGVO algorithms were initially developed based on datasets for slow moving indoor robots [104]. However, their underlying range flow constraint equation assumes that range flow is limited between frames. Consequently, faster motions can violate this assumption and corrupt the resulting ego-motion estimates. This chapter explores the conditions that cause range flow constraint violations for *static* environments. The derivation of these conditions can serve as a critical validation of these algorithms' ability to be scaled and deployed for faster vehicles.

4.2 Deriving the Range Flow

The *range flow constraint equation* describes how the derivative of a point's depth can be calculated based on the temporal and spatial derivatives of a range image. Neglecting higher order terms, Spies et al. derived the range flow constraint equation as follows:

$$\dot{Z} \simeq Z_t + Z_u \dot{u} + Z_v \dot{v} \tag{4.1}$$

where $\dot{Z}$ is the point's total derivative of depth, Z_t is the partial derivative of depth with respect to time t, (u, v) are the point's corresponding range image pixel coordinates, Z_u is the partial derivative of depth with respect to u, and Z_v is the partial derivative of depth with respect to v [56, 57].

The pinhole camera model equations describe the projection of a 3-D point onto the 2-D image plane:

$$\begin{aligned} u &= f_x \frac{x}{z} + c_x \\ v &= f_y \frac{y}{z} + c_y \end{aligned} \tag{4.2}$$

[2]Due to their seamless conversion, *range* and *depth* may be referred to interchangeably in this chapter.
[3]Due to their analogous nature, *range flow* and *optical flow* may be referred to interchangeably in this chapter.

where f_x, f_y are the focal lengths (in pixels) corresponding to the x and y directions, respectively, c_x, c_y are the coordinates of the principal point (optical center) in pixels, and x, y, z are the 3-D coordinates of the point in the camera's coordinate system.

Range flow can be calculated by taking the derivatives of their range image projections in Equation 4.2:

$$\begin{aligned} \dot{u} &= f_x \left(\frac{\dot{x}z - \dot{z}x}{z^2} \right) \\ \dot{v} &= f_y \left(\frac{\dot{y}z - \dot{z}y}{z^2} \right) \end{aligned} \tag{4.3}$$

The derivative of a point's position in the camera's coordinate system depends on the camera's (and, if rigidly mounted, vehicle's) motion in six degrees of freedom [57]:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{pmatrix} -v_x - z\omega_y + y\omega_z \\ -v_y + z\omega_x - x\omega_z \\ -v_z - y\omega_x + x\omega_y \end{pmatrix} \tag{4.4}$$

where v_x, v_y, v_z represent the camera's linear velocities and ω_x, ω_y, ω_z represent the camera's rotational velocities in the camera's coordinate system[4].

Combining Equation 4.3 and Equation 4.4, the range flow can be expressed as a function of the point's 3-D coordinates and the vehicle's motion:

$$\begin{aligned} \dot{u} &= \frac{f_x}{z^2} \Big(-zv_x + xv_z + xy\omega_x - (x^2 + z^2)\omega_y + yz\omega_z \Big) \\ \dot{v} &= \frac{f_y}{z^2} \Big(-zv_y + yv_z + (y^2 + z^2)\omega_x - xy\omega_y - xz\omega_z \Big) \end{aligned} \tag{4.5}$$

Furthermore, Equation 4.2 can be rearranged to express the projection of each point as angles from the optical axis:

$$\begin{aligned} \tan \alpha_x &= \frac{x}{z} = \frac{u - c_x}{f_x} \\ \tan \alpha_y &= \frac{y}{z} = \frac{v - c_y}{f_y} \end{aligned} \tag{4.6}$$

where α_x and α_y describe the projection's angles from the optical axis in the xz and yz planes, respectively. These angles are best described with the illustration in Figure 4.1.

[4]For this chapter, the camera will be assumed to be rigidly mounted to the vehicle in such a way that their motion is equivalent. For range/depth sensors that are not cameras, the equivalent range/depth images will be simulated via projection.

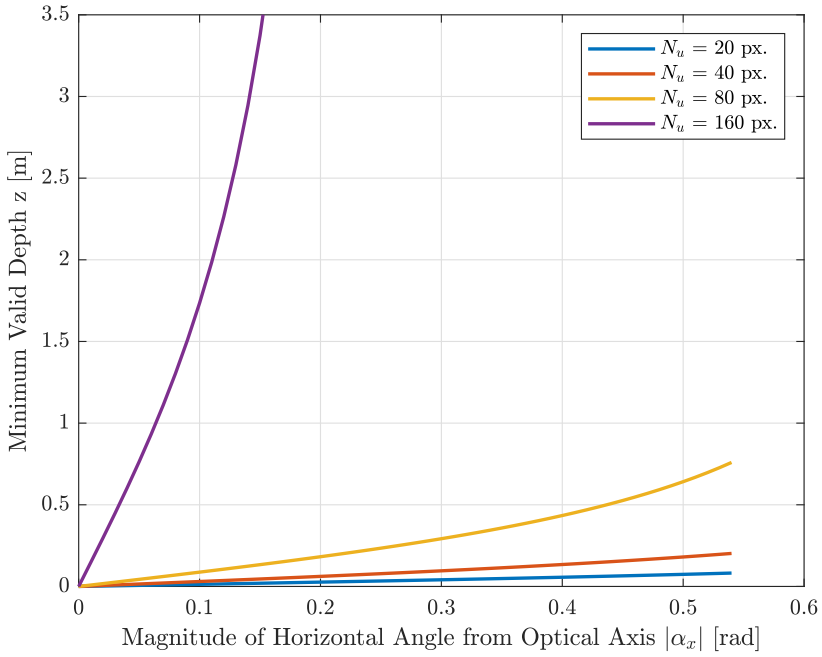


Figure 4.1: Definition of α_x and α_y as the projection's angle from the optical axis in xz and yz planes, respectively.

A range camera's field of view can be calculated from the image sensor size and focal length. The image's width refers to the number of pixels along the u-axis, which will be denoted in this chapter as N_u. Similarly, the image's height refers to the number of pixels along the v-axis (N_v). Given the size of a sensor's individual pixel in world units along the x and y axes (p_x and p_y), the image sensor's width and height in world units are $N_u p_x$ and $N_v p_y$, respectively. A range camera's field of view can then be calculated as follows:

$$\begin{aligned} \tan\frac{FOV_x}{2} &= \tan\alpha_{x,magmax} = \frac{N_u p_x}{2F} = \frac{N_u}{2f_x} \\ \tan\frac{FOV_y}{2} &= \tan\alpha_{y,magmax} = \frac{N_v p_y}{2F} = \frac{N_v}{2f_y} \end{aligned} \tag{4.7}$$

where FOV_x and FOV_y are the range camera's horizontal and vertical fields of views, respectively, $\alpha_{x,magmax}$ and $\alpha_{y,magmax}$ are the magnitudes of the maximum captured angles in the xz and yz planes, respectively, f_x and f_y are the focal lengths (in pixel units) along the x and y directions, respectively, and F is the camera's focal length in world units.

The time between consecutive range images Δt is related to the sampling frequency[5] f_s by the relationship $\Delta t = 1/f_s$. Consequently, the difference in a point's pixel location between two consecutive range images is related to the range flow as follows:

$$\begin{aligned} \Delta u &= \dot{u}\Delta t \\ \Delta v &= \dot{v}\Delta t \end{aligned} \tag{4.8}$$

Combining Equations 4.5, 4.6, 4.7, and 4.8, the point's pixel displacement can be calculated as follows:

[5]The sampling frequency will be assumed to be equal to the algorithm's execution frequency in this chapter.

$$\begin{aligned}\Delta u &= \frac{N_u}{2f_s}\cot\frac{FOV_x}{2}\left(-\frac{1}{z}v_x+\frac{\tan\alpha_x}{z}v_z+\tan\alpha_x\tan\alpha_y\omega_x-\sec^2\alpha_x\omega_y+\tan\alpha_y\omega_z\right)\\ \Delta v &= \frac{N_v}{2f_s}\cot\frac{FOV_y}{2}\left(-\frac{1}{z}v_y+\frac{\tan\alpha_y}{z}v_z+\sec^2\alpha_y\omega_x-\tan\alpha_x\tan\alpha_y\omega_y-\tan\alpha_x\omega_z\right)\end{aligned} \tag{4.9}$$

4.3 Violations of the Range Flow Constraint

The expanded range flow constraint equation (derived by combining Equations 4.1, 4.3, and 4.4) describes the relationship between a sequence of depth images and the vehicle's motion. DGVO methods exploit this equation to estimate the ego-vehicle's motion in six degrees of freedom. However, due to their reliance on the temporal and spatial image derivatives, these methods fundamentally depend on the assumption that range flow is limited between consecutive range images as follows:

$$\begin{aligned}|\dot{u}\Delta t| &< 1\\ |\dot{v}\Delta t| &< 1\end{aligned} \tag{4.10}$$

This constraint is referred to as the "small displacements restriction" [57]. The sensor, motion, environment, and sampling frequency conditions that satisfy this constraint can be calculated by combining Equations 4.8, 4.9, and 4.10:

$$\begin{aligned}\left|\frac{N_u}{2f_s}\cot\frac{FOV_x}{2}\left(-\frac{1}{z}v_x+\frac{\tan\alpha_x}{z}v_z+\tan\alpha_x\tan\alpha_y\omega_x-\sec^2\alpha_x\omega_y+\tan\alpha_y\omega_z\right)\right| &< 1\\ \left|\frac{N_v}{2f_s}\cot\frac{FOV_y}{2}\left(-\frac{1}{z}v_y+\frac{\tan\alpha_y}{z}v_z+\sec^2\alpha_y\omega_x-\tan\alpha_x\tan\alpha_y\omega_y-\tan\alpha_x\omega_z\right)\right| &< 1\end{aligned} \tag{4.11}$$

Conditions that violate Equation 4.11 will be referred to as *range flow constraint violations* and may corrupt ego-motion estimates. To the author's knowledge, the derivation of these conditional violations has not yet been published in literature.

4.4 Current Methods for Mitigating Range Flow Constraint Violations

Intuition and experience provides an independent qualitative validation of the conditions in Equation 4.11 that would violate the small displacement restriction. To address this well-known issue for optical flow, Brox et al. proposed a practical solution that utilizes a "coarse-to-fine" warping strategy [105]. Their scheme requires building a Gaussian pyramid that iteratively downsamples and smooths images to create "coarser" levels. Typically, each coarser level of the pyramid has image dimensions that are half of its neighboring finer level. Since coarser levels would have greatly reduced resolutions compared to the original image, large optical flow displacements in the original image would appear as smaller displacements in coarser levels. Theoretically, with enough levels, all optical flow vectors would satisfy the small displacement restriction at the coarsest level.

An optical flow analysis at the coarsest level would capture the bulk of the motion; however, the reduced resolution would yield a reduced precision. Therefore, to recover full precision, a "warping" strategy is utilized. This strategy uses the motion estimates from optical flow in the coarser levels to "warp" the image in the subsequent finer level. This warping process simulates a depth image captured by a camera that has already moved according to the previous level's estimate so that only small pixel displacements remain. Then

the small displacement restriction for the finer level is satisfied. Each level's warp is based on integrating the motion estimates of all preceding levels. By iteratively employing this strategy from the coarse to fine levels of the pyramid, the optical flow analysis can converge to the true displacement with the full original precision [105].

Of course, another strategy to limit optical flow displacement would be to increase the sampling and/or execution frequency. However, many current applications already operate at their hardware and/or software limits, so further improvements are often marginal. Therefore, the warping strategy previously described is typically the most practical solution.

4.5 Defining the Operational Limits

The abundance of variables in Equation 4.11 contribute to the limitless combinations that could violate the range flow constraint. This equation serves as this chapter's main contribution. It is a tool that DGVO practitioners can use to determine if these violations warrant consideration for their unique application. However, to practically demonstrate the high-level relationships between these variables and range flow, this section will make several assumptions to analyze specific scenarios.

To simplify the effects of vehicle motion, it is useful to condense the 3-D equation to its planar 2-D equivalent. This can be accomplished simply by assuming $v_x = v_y = \omega_x = \omega_z = \alpha_y = \Delta v = 0$. This corresponds to a vehicle whose motion can be described via a unicycle model with a linear velocity v_z and an angular velocity ω_y. Its motion can be fully estimated with a planar scan represented by a depth image of size $N_u \times 1$. Furthermore, excluding cases of geometric degeneracy, each point/pixel provides an independent range flow constraint equation. Therefore, some points may be in violation of the range flow small displacement restriction depending on their unique z, α_x, and α_y values while other points remain valid. By inspection of Equation 4.11, this constraint is most likely to be violated at small z values and/or large α_x, α_y angles. This constraint is also most likely to be violated when the the motion/environment combination creates a range flow vector that aligns with the image's u or v axis. For the simplified 2-D motion previously described, the "worst-case" scenario is one that contains a point at the sensor's minimum depth z_{min} and at the maximum detectable angle $\alpha_{x,magmax}$ whose range flow fully aligns with the u-axis. Consequently, this section will use this "worst-case" scenario to define the operational limits of DGVO methods (i.e. the maximum vehicle speeds that avoid range flow constraint violations) for specific applications.

Since the TUM RGB-D SLAM Dataset is the common baseline dataset for DVO algorithms, its depth sensor (the Microsoft Kinect[6] will be used for this planar analysis[7] [26, 27, 104]. Figure 4.2 illustrates the corresponding operational limits at different Gaussian pyramid levels for the "worst-case" 2-D scenario. It is apparent that the Gaussian pyramid levels are critical for ensuring range flow constraint validity as they quadruple the operational "area." However, it should be noted that there is a lower limit to the resolution of the pyramid's coarsest level. Jaimez et al. set their coarsest pyramid level's resolution to 20×15, the lowest capable of producing valid estimates [57]. Consequently, with this approach alone, there are limits to the vehicle motions that can be reliably estimated.

[6] Minimum Depth: 0.5 m; Depth Sensor Horizontal Field of View: 62.7°; Depth Image Width: 640 pixels; Sampling Frequency: 30 Hz [106] [104]

[7] It was assumed that only the row at $v \approx c_y$ was used to emulate a horizontal depth image with dimensions $N_u \times 1$.

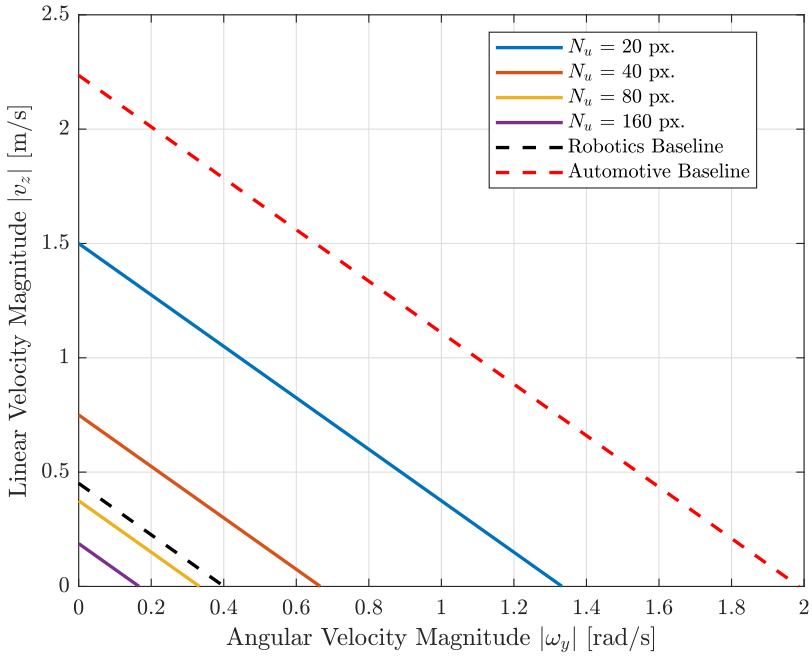


Figure 4.2: Dense geometry-based visual odometry operational limits for the Kinect depth sensor at different Gaussian pyramid levels. The "baseline" values correspond to fictional pyramid levels that contain the maximum N_u needed to eliminate any range flow violations for their associated baseline motions.

To illustrate these operational limits, robotics "baseline" values were plotted in Figure 4.2 as well. These baseline values correspond to a fictional pyramid level that contains the maximum N_u needed to eliminate any range flow violations for its associated baseline motion. This baseline motion can be considered to be representative of "typical" robotics motion[8]. Fortunately, this motion resides within the two coarsest level's operational limits, eliminating the need to further consider violations of the range flow constraint. This helps explain the success of DVO algorithms for this dataset.

However, if one were to apply these DGVO methods to faster moving vehicles, like automobiles, range flow constraint violations may regularly occur. An automotive baseline was plotted in Figure 4.2 based on the Kinect sensor in the same manner as previously described. In contrast to the robotics baseline, this baseline corresponds to the most *conservative* automotive motion[9]. Automobiles regularly exceed this speed by more than a factor of ten. Yet, this still resides outside of the operational limits of even the coarsest valid level. This indicates that, in their current form, these methods are not valid for use when scaled for these faster applications. Furthermore, unlike the Kinect, automotive sensors may have a lower f_s, lower z_{min}, and a higher $\alpha_{x,magmax}$, all of which would contribute to more stringent operational limits.

4.6 Ego-Motion Estimate Corruption

Fortunately, to mitigate the risk of collision, most faster vehicles try to maintain a space "cushion," reducing the probability that points will reside at worrisome low z values. Since Figure 4.2 was derived from the worst-case point, it is apparent that its operational limits are extraordinarily conservative. Each pixel/point

[8]The "typical" velocities were considered to be $v_z = 0.21\,m/s$ and $\omega_y = 12.27\,deg/s$. These values correspond to the median of each of the average velocities for the four Robot SLAM sequences in the TUM RGB-D dataset.

[9]The most conservative automotive motion was roughly considered to be the forward creep motion present on automatic transmission vehicles ($v_z = 5\ mph$).

corresponds to its own range flow constraint equation. Therefore, to calculate n ego-motion state variables, n independent range flow constraint equations are required. If more than n equations are available, the system is overdetermined and benefits from a least-squares solution. As a result, if only some points violate the small displacement restriction, an abundance of valid points may vastly outnumber the corrupted points, automatically reducing ego-motion estimate corruption. In other words, range flow violators may be so rare that this method may be used slightly outside of its operational limits with a minimal loss in accuracy.

Despite this, the operational limits in Figure 4.2 are useful because they designate when corruption is *possible*. If corruption is possible, DGVO practitioners should conduct a more extensive analysis using Equation 4.11 to estimate if the prevalence of violators will render ego-motion estimates too inaccurate for their specific use. The farther the expected vehicle motion strays from these conservative operational limits, the greater the ego-motion estimate is corrupted. For this reason, Figure 4.2 is best suited to illustrate the surprising accessibility of scenarios that contain violations. For many applications with faster moving vehicles, violations will occur. However, it is their expected *prevalence* in a specific application that determines whether DGVO methods are a practical solution or not. Consequently, current DGVO methods are ill-suited for faster vehicles that stray substantially from these operational limits.

To provide an illustrative example of the prevalence of range flow constraint violators, the baseline robotics scenario described in Section 4.5 was reexamined. Instead of assuming that the point resides at $\alpha_{x,magmax}$, the minimum depth for a point to be considered valid was plotted in Figure 4.3 for each horizontal angle.

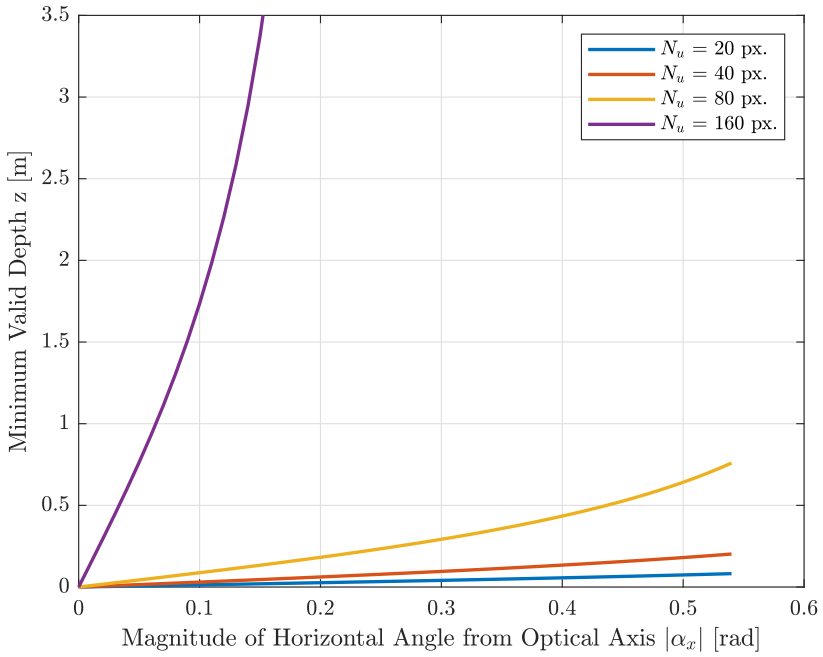


Figure 4.3: Point requirements to satisfy the range flow constraint equation for the baseline robotics scenario at different Gaussian pyramid levels

As expected, the plot validates intuition. The minimum valid depth exponentially increases with the magnitude of its horizontal angle from the optical axis. For the same linear velocity, far points that are nearly straight ahead (reside at small angles from the optical angles) should have smaller optical flow displacements than close points at large angles. Consequently, far points may provide valid ego-motion estimates that close points cannot. Furthermore, the plot shows that coarser levels of the Gaussian pyramid are critical for drastically relaxing the point requirements. For the two coarsest levels, all points within the Kinect's horizontal field of view have a minimum valid depth less than the sensor's minimum depth ($0.5\ m$). Therefore,

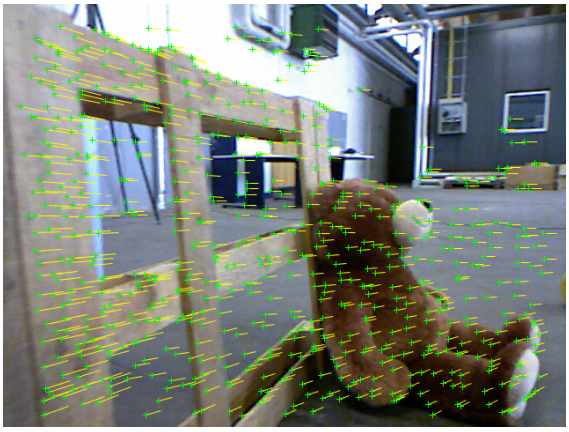
all possible points are valid *for this motion, sensor, sampling frequency, and pyramid level combination.*

The two finest levels were not plotted at all because, for the given motion, sensor, and sampling frequency combination, no valid depths existed. This can be explained by the ego-vehicle possessing an angular velocity. While the optical flow displacement of linear velocity depends on depth, the displacement attributed to angular velocity does not. Consequently, the Gaussian pyramid scheme is critical for providing valid points while the vehicle is undergoing large angular velocities.

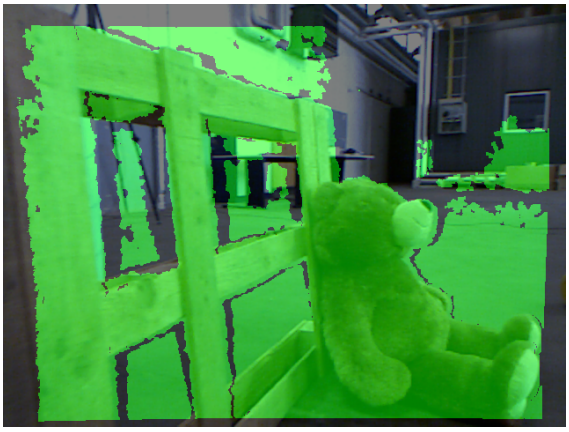
The infiltration of range flow constraint violators is best demonstrated visually as in Figure 4.4. The TUM RGB-D dataset was used to simulate optical/range flow for motion at the operational limit of the coarsest valid pyramid level[10] defined by Figure 4.2. As can be inferred from the optical flow vectors in Figure 4.4a, this motion contained both a v_z linear velocity and an ω_y angular velocity[11]. Colored overlays were used in Figure 4.4b to indicate photometric pixels with a valid corresponding depth measurement. If the overlay is green, these pixels are *not* in violation of the range flow constraint at the coarsest valid pyramid level. As predicted by Figure 4.2, Figure 4.4b's universally green overlay indicates that there are no violators within the range data for this vehicle motion.

[10] $N_u = 20\ px.$ for the coarsest valid pyramid level

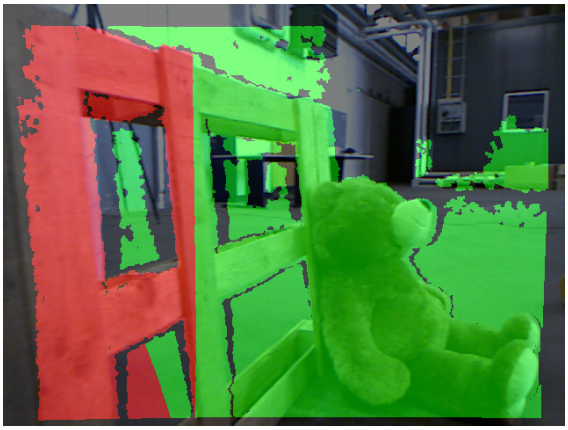
[11] Vehicle velocities: $v_z = 0.7\ m/s$; $\omega_y = 40.8\ deg/s$



(a) Image features (green) and their associated optical flow vectors (yellow) while the vehicle is undergoing linear and rotational motion



(b) Green overlay indicates pixels with a valid depth measurement that are not in violation of the range flow constraint



(c) Corresponds to vehicle speeds that are twice as fast as those represented in Figures 4.4a and 4.4b. Red overlay indicates pixels with a valid depth measurement that are in violation of the range flow constraint. The infiltration of these violators corrupt the ego-motion estimates.

Figure 4.4: Infiltration of range flow constraint violators as vehicle speeds are increased

To demonstrate the infiltration of range flow constraint violators as vehicle speeds are increased, the range image was then analyzed for a vehicle motion twice as fast as the motion previously described [12]. Figure 4.4c was annotated with colorized overlays similarly to Figure 4.4b. However, this image shows a red overlay, which indicates pixels with a valid depth measurement that are in violation of the range flow constraint. It is now readily apparent how the combination of the positive linear and angular velocities contributed to the infiltration of range flow constraint violators on the left edge of the image. It should be noted that the "wave" of red passed over some of the pixels in the gaps of the wooden palette. This range flow constraint satisfaction can be explained by their larger z values, which is critical for reducing the range flow associated with the vehicle's linear velocity.

Furthermore, Figure 4.4c illustrates the relative prevalence of range flow constraint violators. As vehicle speeds increase, the number of violators may increase, increasing ego-motion estimate corruption. This is

[12] Vehicle velocities: $v_z = 1.4\ m/s$; $\omega_y = 81.6\ deg/s$

consistent with published experimental results [57].

4.7 Range Flow Constraint Conclusion

Current DGVO techniques are well-suited for the slow indoor robotics scenarios that they were developed for. They offer the rare ego-motion estimation solution for robots in which only geometric data is available. However, this chapter's analysis demonstrates that scaling these methods for faster moving vehicles exposes practical limitations because violations of the range flow constraint will corrupt these methods' ego-motion estimates.

The equations derived in this chapter are useful for determining if these methods will generate valid ego-motion estimates given the expected sensor, motion, environment, and sampling frequency conditions. These equations imply that current DGVO methods are impractical for faster moving vehicles. However, future variants may be able to incorporate these equations to predict range flow constraint violation candidates. If the influence of these candidates can be minimized, the ego-motion estimate corruption can be mitigated, relaxing the method's operational requirements. This may offer a promising ego-motion estimation solution for fast moving vehicles that operate in challenging environments.

Chapter 5

RADARODO: Ego-Motion Estimation from Doppler and Spatial Data in RADAR Images

Abstract − Accurate and robust ego-motion estimation is critical for aiding autonomous vehicle localization in environments devoid of map landmarks or accurate GNSS measurements. RADAR measurements provide unique capabilities for ego-motion estimation while also posing unique challenges. This chapter presents the novel real-time RADAR-based ego-motion estimation algorithm, RADARODO (RADAR Odometry). RADARODO is differentiated from similar techniques by two novel attributes. First, it decouples sensor translational and rotational motion by estimating them from Doppler and spatial data, respectively. Secondly, it directly analyzes RADAR images to estimate rotational motion via correspondence-free non-linear optimization. Unlike other methods, RADARODO estimates ego-motion along with its predicted uncertainty from a single RADAR sensor without depending on a lever-arm offset or the zero side-slip assumption. The real-world results demonstrate accurate ego-motion estimates that are particularly suited for integration within a "tightly-coupled" sensor fusion framework.

5.1 Introduction to RADARODO

Many autonomous vehicle subsystems fundamentally depend on fast and accurate localization. Due to the limitations of Global Navigation Satellite System (GNSS) measurements, robust autonomous vehicles localize by matching observed landmarks with those stored in an *a priori* map [4, 11, 107]. However, there are many environments devoid of distinct and consistent landmarks. In these challenging environments, ego-motion estimation can serve as a critical aid [3]. For instance, wheel odometry measurements are often utilized. Yet, they are easily corrupted by tire slip, particularly while traversing over friction-limited or granular surfaces [108, 109]. To mitigate these ego-motion estimate errors, camera-based "Visual Odometry" [7, 17, 23, 110] or LIDAR-based odometry [46] is often used.

However, the weather conditions most conducive to tire slip (e.g. snow, rain, sleet, etc.) also correspond to substantially reduced visibility. Since both cameras and LIDAR operate in or near the visible spectrum, their measurements are affected by precipitation, resulting in extremely noisy measurements [19]. Furthermore, low-visibility conditions can reduce the measurement range or fine texture required for many camera- or LIDAR-based methods [17, 32, 107, 111].

Conversely, RADAR operates in the radio spectrum. Its signals' larger wavelengths better penetrate the atmosphere and is thus less affected by weather conditions. Furthermore, it has the rare ability to measure

At the time of this writing, this chapter is under review for publication in the IEEE Transactions on Intelligent Vehicles (T-IV).

the relative velocities of targets via Doppler shift. These qualities, along with its relatively low cost, have solidified RADAR as a foundation of many driver assistance systems [19, 109].

Therefore, at first glance, RADAR appears very compelling for ego-motion estimation. Unlike camera- and LIDAR-based methods, RADAR can measure relative motion "instantaneously" with impressive accuracy. Unfortunately, however, its Doppler measurements can only observe relative *radial* motion (i.e. range rates). Intuitively, it follows that these measurements are unable to observe angular motion, a foundation critical for direct rotational motion estimates. This is demonstrated analytically in Section 5.3.

Despite this, there are ways to recover rotational motion from Doppler measurements. RADAR sensors are often mounted at a lever-arm offset from their ego-vehicle's reference frame. At these offsets, rotational motion is also transformed into translational motion, enabling detection [80, 81].

However, these approaches depend on several important assumptions. Primarily, they depend on the sensor's lever-arm magnitude since it determines relative observabilities. Consequently, at small lever-arm offsets, it may be difficult to correct rotational ego-motion [82]. Furthermore, at these lever-arm offsets, it is often ambiguous whether the sensor's translational motion is due to the vehicle's translational motion (e.g. lateral velocity) or their shared rotational motion. This ambiguity is typically resolved by utilizing multiple sensors at different locations or by making vehicle motion assumptions. It is often fair to use the "zero side-slip" assumption; in other words, the assumption that the vehicle exhibits no lateral motion. Yet, the low-friction weather conditions most conducive to visibility and odometry issues are also most conducive to violating this side-slip assumption [112]. This issue is further exacerbated by vehicles that do not adhere to Ackermann steering geometries or vehicles that are susceptible to crosswind-induced lateral motion (e.g. multi-axle semi-trucks and buses) [113].

Alternatively, the RADAR targets' azimuthal angle changes can be uniquely observed by spatial RADAR measurements. Yet, unlike the relatively deterministic paths of LIDAR and camera optical rays, RADAR's radio waves are more susceptible to corrupting beam divergence, multi-path propagation, and interference effects [109]. Thus, the analysis of RADAR returns is more probabilistic in nature [114, 115]. This analysis, discussed further in Section 5.3.1, involves generating RADAR images that capture this uncertainty. Despite this, most automotive RADAR units only provide point targets extracted from these images, constraining many RADAR-based methods to depend on this high-level data exclusively.

Consequently, this chapter presents RADARODO (RADAR Odometry), a novel real-time RADAR-based ego-motion estimation algorithm. RADARODO is differentiated from similar techniques by two novel attributes:

1. It decouples translational and rotational motion by estimating them from Doppler and spatial data, respectively.

2. It directly analyzes RADAR images to estimate rotational motion via correspondence-free non-linear optimization.

Unlike many related methods, RADARODO estimates ego-motion (and its predicted uncertainties) from a single RADAR without depending on a lever-arm offset or the zero side-slip assumption.

5.2 Related Work

RADARODO builds upon several related RADAR ego-motion estimation methods. In their seminal work, Kellner et al. estimated translational velocities via a least-squares fit of a RADAR's measured targets' radial velocities and azimuthal angles [80]. Furthermore, they used a RANSAC scheme to classify targets as static or dynamic.

However, Kellner et al.'s "Sinefit" algorithm initially assumed that the vehicle exhibits zero side-slip to transform these estimates into yaw rates [80]. Therefore, to avoid motion assumptions and ambiguities, they extended their previous work to utilize multiple distributed RADAR sensors [81]. This extension estimated vehicle motion in all planar degrees of freedom. Due to this algorithm's success, RADARODO's translational velocity estimation subprocess (detailed in Section 5.3.2) largely mirrors this algorithm.

To permit estimates from a single RADAR, subsequent research incorporated spatial registrations of RADAR scans. Notably, Barjenbruch et al. used probabilistic correspondence-free registration with a novel joint spatial- and Doppler-based error metric [72]. By incorporating spatial data, they also did not require lever-arm offsets or motion assumptions, although their Doppler-based metric was aided by their existence. However, Barjenbruch et al. found that, "Due to the additional degree of freedom the accuracy of the yaw-rate estimation is reduced compared to the 2 DOF case" [72]. For related reasons, one of RADARODO's novel attributes is that it decouples sensor translational and rotational motion by estimating them from Doppler and spatial data, respectively.

Furthermore, Barjenbruch et al.'s probabilistic registration depended on Gaussian Mixture Models (GMMs) [65,67] of extracted point targets [72]. Unfortunately, this computationally expensive approach rendered the method infeasible for real-time deployment. Therefore, Rapp et al. reformulated the method to utilize the more efficient Normal Distribution Transform (NDT) [59, 60] algorithm instead [73, 74].

Both Barjenbruch et al. and Rapp et al. represented their point targets with spatial probability density functions. In general, different methodologies had varying success in representing their points' structure [73,74]. Yet many of these probability density functions appeared to visually resemble their points' underlying RADAR images. Specifically, the high likelihood regions of these methods' probability density functions appeared to correspond to high signal-to-noise ratio (SNR) regions in RADAR images. For this reason, in lieu of attempting to generate representative distributions, RADARODO uniquely utilizes the RADAR images themselves.

5.3 Algorithm Overview

The high-level overview of RADARODO's inputs, processes, and outputs can be visualized in Figure 5.1. As the figure illustrates, RADARODO contains two main subprocesses that are mostly decoupled: the Translational Velocity Estimator and the Rotational Velocity Estimator. RADARODO's C++ source code and this chapter's validation dataset are available online at `https://bitbucket.org/chrisdmonaco/radarodo`.

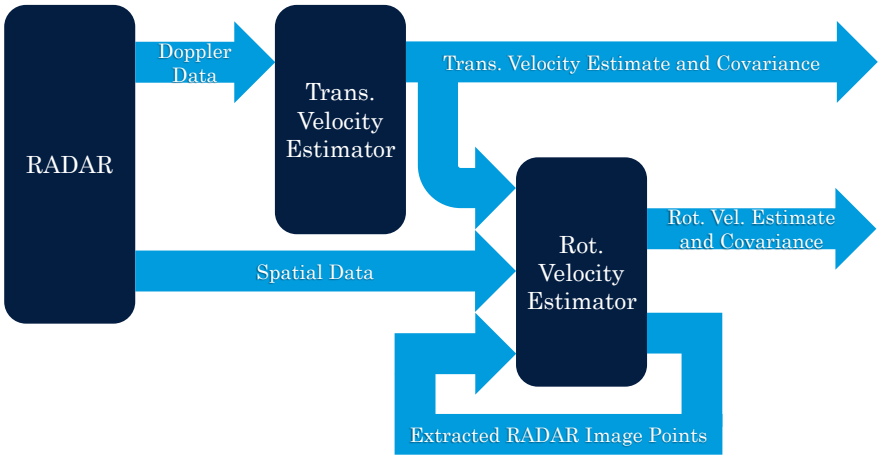


Figure 5.1: RADARODO Algorithm Overview

One of the most significant capabilities of RADAR for motion estimation is its ability to instantaneously measure relative radial velocities via Doppler shift. This phenomenon can be analytically described for planar motion by taking the time derivative of range, as seen in Equation 5.1:

$$\frac{\partial}{\partial t}\left(r^2\right) = \frac{\partial}{\partial t}\left(x^2 + y^2\right) \Rightarrow \dot{r} = \frac{x}{r}\dot{x} + \frac{y}{r}\dot{y} \tag{5.1}$$

where r is the range of a target from the RADAR sensor in the xy-plane, x is the position of the target along the sensor's longitudinal x-axis, y is the position of the target along the sensor's lateral y-axis, $\dot{r}$ is the relative velocity of the target with respect to the RADAR along the radial axis (i.e. range rate), and $\dot{x}$, $\dot{y}$ are the time derivatives of the target's position in the sensor's Cartesian coordinate system. Assuming that the target is static, the relative position's time derivative can be related to the sensor's planar ego-motion via Equation 5.2:

$$\begin{aligned} \dot{x} &= -v_x + y\omega_z \\ \dot{y} &= -v_y - x\omega_z \end{aligned} \tag{5.2}$$

where ω_z is the sensor's rotational velocity about its vertical z-axis (i.e. yaw rate) and v_x, v_y are the sensor's translational velocities along its longitudinal and lateral axes, respectively. Finally, Equations 5.1 and 5.2 can be combined to yield the relationship between range rate measurements and the sensor's ego-motion, as described in Equation 5.3:

$$\dot{r} = -\cos\theta\, v_x - \sin\theta\, v_y \tag{5.3}$$

where θ is the azimuthal angle of the target from the RADAR's x-axis. Interestingly, Equation 5.3 shows that all terms related to rotational ego-motion have cancelled each other out. This analytical result illustrates that Doppler measurements cannot directly measure rotational ego-motion.

This phenomenon may initially appear to be a RADAR shortcoming. However, this analytical decoupling can be advantageous. This becomes clearer by taking the time derivative of a target's azimuthal angle, as seen in Equation 5.4:

$$\frac{\partial}{\partial t}\left(\sin^2\theta\right) = \frac{\partial}{\partial t}\left(\frac{y^2}{x^2+y^2}\right) \Rightarrow \dot{\theta} = -\frac{\sin\theta}{r}\dot{x} + \frac{\cos\theta}{r}\dot{y} \tag{5.4}$$

Equation 5.2 can be substituted into Equation 5.4 to yield the relationship between the target's azimuthal motion and the sensor's ego-motion, as seen in Equation 5.5:

$$\dot{\theta} = \frac{\sin\theta}{r}v_x - \frac{\cos\theta}{r}v_y - \omega_z \tag{5.5}$$

Equation 5.5 demonstrates that an observed azimuthal angle change can be the result of motion in all planar degrees of freedom. Therefore, measuring these changes can yield complete planar ego-motion estimates. Unfortunately, however, a characteristic weakness of RADAR measurements is its poor angular resolutions [19]. Consequently, the coupling present in Equation 5.5 also allows azimuthal motion uncertainties to corrupt ego-motion estimates in all planar degrees of freedom. This agrees with the findings of Barjenbruch et al., as discussed in Section 5.2.

Fortunately, Equations 5.3 and 5.5 demonstrate that certain degrees of freedom can be decoupled. As shown in Equation 5.5, rotational ego-motion can only be directly observed by azimuthal angle changes. Azimuthal angle changes are determined by a target's relative position in space (r, θ). Thus, direct rotational ego-motion estimation depends on spatial measurements. Conversely, Equation 5.3 demonstrates that Doppler measurements can estimate translational ego-motion *independent* of rotational ego-motion. Furthermore, these estimates can be very accurate since range rate accuracy is one of RADAR's characteristic strengths [19, 115].

Therefore, these accurate translational velocity estimates can be substituted into Equation 5.5 so that only one degree of freedom, the yaw rate, remains. Theoretically, this should substantially simplify the convergence process and mitigate the corrupting influence of range-based local minima.

To that end, RADARODO utilizes Doppler and spatial data based on their individual strengths. Specifically, it decouples sensor translational and rotational motion by estimating them from Doppler and spatial data, respectively.

5.3.1 RADAR Image Emulation

RADAR units typically deploy Fourier analyses on returned signals to generate SNR "heatmaps." The Range-Azimuth heatmaps are commonly called RADAR images since they can resemble polar images of the observed environment. RADAR's various uncertainties are represented in those images' continuous distributions. This results in a lack of distinct and consistent edges. Thus, the size and boundaries of individual objects are often ambiguous.

To promote practical use, almost all automotive RADAR units try to resolve these ambiguities by extracting point targets. For instance, a point target may be extracted by detecting a SNR peak in the image. The provided ranges, azimuthal angles, range rates, and SNRs for each extracted point target offer a plethora of information that have been used by the related algorithms discussed in Section 5.2.

However, as with all high-level extractions, the provided points only partially describe their underlying RADAR images. Since boundaries are ambiguous, the RADAR's algorithms may generate contours that incorrectly divide a large object or group several that are nearby. The extracted point targets are then typically their contours' centroids. Yet, the centroid location can be extremely sensitive to contour inaccuracies and size [116]. Thus, an inconsistent extraction algorithm may introduce even more uncertainty than what already inherently exists with RADAR measurements.

For this reason, unlike other methods, RADARODO aims to directly utilize the underlying RADAR images as the distribution. Theoretically, the use of this lower-level data would improve estimate accuracy by avoiding errors introduced by point target extraction. Furthermore, this should reduce the algorithm's computational cost. While previous methods spent computational resources to recreate distributions from point targets, RADARODO aims to utilize a distribution that was already created for those point targets to exist. RADARODO further reduces computational costs by only using the image to estimate motion in one degree of freedom.

Unfortunately, the author is unaware of any commercially-available automotive-grade RADARs that publish its internal RADAR images in a practically accessible manner. Therefore, the author instead utilized a popular electronically scanning RADAR, the Delphi ESR 2.5 [117], for this method's validation. Consequently, for this implementation, the aforementioned RADAR images were *emulated* using real extracted point targets. RADARODO is based on the principle that RADAR images contain valuable data for state estimation that is otherwise lost during high-level extractions. Thus, RADAR images should be more readily available from RADAR sensors.

To emulate RADAR images, the point targets were utilized very similarly to the Gaussian Mixture Model process deployed by Barjenbruch et al. [72]. Yet, to realize several benefits, this chapter's implementation had notable differences. Specifically, to better represent the underlying RADAR images, the point targets' reported SNRs were taken into account. To realize real-time operation, the distribution was generated using efficient computer vision techniques in lieu of costly analytical calculations.

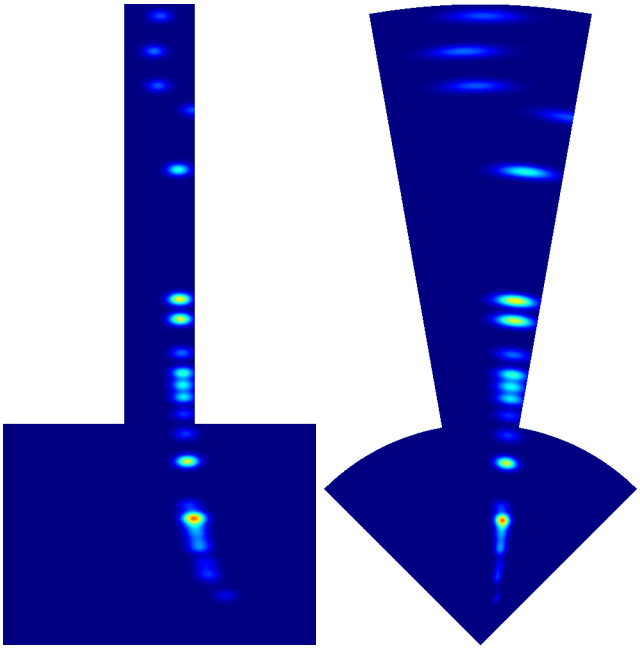
As will be discussed in Section 5.3.3, RADARODO fundamentally depends on the polar RADAR image. Consequently, this polar image was generated by summing the individual Gaussian distributions of each inlier point target, similar to the Sum of Gaussians. The center location of each distribution within the image was determined by its reported polar coordinates. Thus, the sum of these distributions yields the continuous image.

Each point's distribution was efficiently represented with their own image matrices by storing normally-distributed probability values in the image's discrete grid. Each distribution's sigma values corresponded to the sensor's expected uncertainties.[2] The distribution probability values were also scaled so that its center pixel represented its point target's SNR. Since images represent data via pixel intensities, a proportional relationship between the pixel intensity range and SNR range was derived. This relationship is shown in Equation 5.6:

$$I_\mu = \left(\frac{SNR(r,\theta)_{dB} - SNR_{min,dB}}{SNR_{max,dB} - SNR_{min,dB}} \right) I_{max} \tag{5.6}$$

[2]The size of each distribution matrix was set to $\pm 3\sigma$ to mitigate both discontinuities and inconsequential summations.

where I_μ is the pixel intensity of the point target distribution's center pixel (i.e. its mean), $SNR(r, \theta)_{dB}$ is the point target's reported SNR in decibels, $SNR_{min,dB}$ is the minimum SNR required for a point to be extracted, and $SNR_{max,dB}$ is the maximum possible SNR for extracted point targets, and I_{max} is maximum pixel intensity for the utilized image datatype. An emulated polar image (and its Cartesian equivalent) can be seen in Figure 5.2.



(a) Polar Image (b) Cartesian Image

Figure 5.2: Polar and Cartesian RADAR images (visualized as heatmaps) emulated from RADAR point targets. Enlarged sigma values used for visualization purposes.

It should be noted that this emulation is far from ideal. While the expected centroid range and angle uncertainties were known, the size and shape of their underlying distributions were not. Thus, Gaussian distributions may serve as poor representations. Therefore, this chapter was not able to fully validate the predicted accuracy benefits of directly analyzing true RADAR images.

5.3.2 Translational Velocity Estimator

Least Squares Solution

RADARODO uses Doppler data to estimate translational ego-motion by solving Equation 5.3 via Least Squares Regression. This linear system is represented as a matrix in Equation 5.7:

$$\begin{bmatrix} -\cos\theta_1 & -\sin\theta_1 \\ -\cos\theta_2 & -\sin\theta_2 \\ \vdots & \vdots \\ -\cos\theta_N & -\sin\theta_N \end{bmatrix} \underbrace{\begin{bmatrix} v_x \\ v_y \end{bmatrix}}_{\vec{v}_S} = \begin{bmatrix} \dot{r}_1 \\ \dot{r}_2 \\ \vdots \\ \dot{r}_N \end{bmatrix} \tag{5.7}$$

where 1, 2, ... N refers to the RADAR's individual extracted targets and $\vec{v}_S$ denotes the sensor's (RADAR's) translational ego-motion vector. Note that at least two linearly independent data points are required for a valid estimate. Since the utilized RADAR unit did not provide a Doppler-Azimuth image, this chapter's implementation used the provided point targets instead.

Static Target Extraction

Automotive vehicles co-exist in an environment shared with many other dynamic agents. Therefore, all detected targets cannot be assumed to be static. Consequently, Equation 5.7 was used within a RANSAC scheme to isolate inlier (static) targets from outlier (dynamic) targets.[3] Specifically, inliers were characterized by their cohesive range rate measurements.[4] Then, all inliers were used to calculate an improved translational velocity estimate.

Non-linear Optimization

While the Least Squares Solution works very well, its calculation is based on an incorrect implicit assumption. Specifically, it assumes that the RADAR has perfect measurements of its target's azimuthal angles. Therefore, these estimates can be further refined with a non-linear optimization framework that takes this uncertainty into account. This framework would converge to the optimal estimate by cumulatively minimizing the inlier targets' range rate and azimuthal errors. Overall, this optimization process is very similar to the Orthogonal Distance Regression deployed by [118] and [81] and the non-linear optimization deployed by [72–74]. However, these works do not explicitly provide the necessary underlying equations. Consequently, the rest of this section defines these equations so they are made accessible to the reader.

The range rate error metric is described in Equation 5.8 for the i^{th} point target:

$$e_{\dot{r},i} = \dot{r}_i - \hat{\dot{r}}_i \tag{5.8}$$

where the $\hat{}$ superscripts denote predicted or estimated values here and for the remainder of this chapter. Thus, $\hat{\dot{r}}_i$ can be calculated via Equation 5.3 with $\hat{v}_x$, $\hat{v}_y$, and θ_i. Furthermore, this non-linear optimization introduces the azimuthal angle error metric[5], as described in Equation 5.9:

$$e_{\theta,i} = \theta_i - \hat{\theta}_i = \theta_i - \left(\hat{\alpha} \pm \hat{\beta}_i\right) \tag{5.9}$$

where α is the azimuthal angle of the RADAR's translational motion vector. It is mathematically defined in Equation 5.10:

$$\alpha = \text{atan2}\left(v_y, v_x\right) \tag{5.10}$$

Theoretically, range rates measured in the direction of the RADAR's translational motion vector should be equal and opposite to its vector norm. Thus, the range rate's magnitude should decrease as the target deviates from this direction. Therefore, β_i is defined as the angle between the RADAR's translational motion vector and the vector from the RADAR to the i^{th} target. It is mathematically defined in Equation 5.11:

$$\beta_i = \arccos\left(-\dot{r}_i / \left\|\vec{v}_S\right\|_2\right) \tag{5.11}$$

where $\left\|\vec{v}_S\right\|_2$ denotes the Euclidean norm of the RADAR's translational ego-motion vector. Consequently, the $\pm\hat{\beta}_i$ in Equation 5.9 accounts for the fact that, given the sensor motion, each range rate measurement could have originated from two possible azimuthal angles. Specifically, it is ambiguous if $\hat{\theta}_i$ should reside to the left or right of $\hat{\alpha}$. This ambiguity can be resolved by comparing both predicted angles to the actual azimuthal angle measurement.

[3] In this implementation, only static targets were used to generate the emulated RADAR image. Due to a lack of real RADAR images, the techniques to reduce the SNRs of dynamic targets in real Range-Azimuth RADAR images were beyond the scope of this chapter.

[4] The inlier range rate residual threshold was set to $2\sigma_{\dot{r}}$, where $\sigma_{\dot{r}}$ is the expected standard deviation of the range rate measurements.

[5] For practical implementations, the azimuthal error metric, $e_{\theta,i}$ should be scaled by $\text{erf}\left(\sqrt{\hat{v}_x^2 + \hat{v}_y^2}\right)$, where $\text{erf}\left(x\right)$ is the Gauss error function. This scaling factor smoothly eliminates this error metric's influence only for translational velocities near zero. This is necessary because the expected azimuthal angles are grossly unreliable for these negligible velocities.

For the framework to understand how it should adjust its next iteration's estimate, the partial derivatives of each measurement's errors with respect to the vehicle's ego-motion must be defined. Without proof, this Jacobian, $\boldsymbol{J}_{\vec{e}_i,\vec{v}_S}$, is defined in Equations 5.12 and 5.13:

$$\boldsymbol{J}_{\vec{e}_i,\vec{v}_S} = \begin{bmatrix} \frac{\partial e_{\dot{r},i}}{\partial \hat{v}_x} & \frac{\partial e_{\dot{r},i}}{\partial \hat{v}_y} \\ \frac{\partial e_{\theta,i}}{\partial \hat{v}_x} & \frac{\partial e_{\theta,i}}{\partial \hat{v}_y} \end{bmatrix} = \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_x} \mp \frac{\partial \hat{\beta}_i}{\partial \hat{v}_x} & -\frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_y} \mp \frac{\partial \hat{\beta}_i}{\partial \hat{v}_y} \end{bmatrix} \tag{5.12}$$

where

$$\begin{bmatrix} \frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_x} & \frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_y} \\ \frac{\partial \hat{\beta}_i}{\partial \hat{v}_x} & \frac{\partial \hat{\beta}_i}{\partial \hat{v}_y} \end{bmatrix} = \begin{bmatrix} \frac{-\hat{v}_y}{\hat{v}_x^2+\hat{v}_y^2} & \frac{\hat{v}_x}{\hat{v}_x^2+\hat{v}_y^2} \\ \frac{-\hat{v}_x \dot{r}_i}{(\hat{v}_x^2+\hat{v}_y^2)^{3/2}\sqrt{1-\frac{\dot{r}_i^2}{\hat{v}_x^2+\hat{v}_y^2}}} & \frac{-\hat{v}_y \dot{r}_i}{(\hat{v}_x^2+\hat{v}_y^2)^{3/2}\sqrt{1-\frac{\dot{r}_i^2}{\hat{v}_x^2+\hat{v}_y^2}}} \end{bmatrix} \tag{5.13}$$

However, due to a variety of factors, each Doppler measurement, $\vec{z}_i$, constrains the vehicle ego-motion estimate differently. Therefore, $\boldsymbol{\Sigma}_{\hat{\vec{v}}_S,\vec{z}_i}$, the translational ego-motion covariance matrix corresponding to each Doppler measurement, must be defined. This can be calculated if the measurement's expected range rate and azimuthal uncertainties ($\sigma_{\dot{r}}$, σ_θ) are provided by the RADAR's datasheet. These measurement uncertainties can then be propagated to the vehicle ego-motion estimate's uncertainties as shown in Equations 5.14 and 5.15:

$$\boldsymbol{\Sigma}_{\hat{\vec{v}}_S,\vec{z}_i} = \begin{bmatrix} \sigma_{\hat{v}_x}^2 & \sigma_{\hat{v}_x\hat{v}_y} \\ \sigma_{\hat{v}_y\hat{v}_x} & \sigma_{\hat{v}_y}^2 \end{bmatrix}_{\vec{z}_i} = \boldsymbol{J}_{\hat{\vec{v}}_S,\vec{z}_i} \begin{bmatrix} \sigma_{\dot{r}}^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix} \boldsymbol{J}_{\hat{\vec{v}}_S,\vec{z}_i}^T \tag{5.14}$$

where

$$\boldsymbol{J}_{\hat{\vec{v}}_S,\vec{z}_i} = \begin{bmatrix} \frac{\partial \hat{v}_x}{\partial \dot{r}_i} & \frac{\partial \hat{v}_x}{\partial \theta_i} \\ \frac{\partial \hat{v}_y}{\partial \dot{r}_i} & \frac{\partial \hat{v}_y}{\partial \theta_i} \end{bmatrix} = \begin{bmatrix} -1/\cos\theta_i & \frac{-\dot{r}_i \sin\theta_i - \hat{v}_y}{\cos^2\theta_i} \\ -1/\sin\theta_i & \frac{\dot{r}_i \cos\theta_i + \hat{v}_x}{\sin^2\theta_i} \end{bmatrix} \tag{5.15}$$

For this chapter's implementation, the aforementioned error metrics, Jacobians, and covariances[6] were directly integrated into the g^2o non-linear optimization framework [119] to estimate the sensor's translational velocities. The Least Squares Solution served as the initial estimate. Furthermore, this framework calculated its estimates' expected uncertainties. This environment-dependent information is critical for later RADARODO subprocesses and integration into overarching sensor fusion frameworks. In all, this estimation process is very similar to the Orthogonal Distance Regression implemented by [118] and [81].

5.3.3 Rotational Velocity Estimator

RADARODO estimates rotational ego-motion from the spatial data captured in RADAR images. Essentially, it estimates the heading change that would maximize the correlation between the current and previous RADAR images. Dividing the sensor's heading change, $\delta\psi$, by the time between two RADAR image measurements, δt, yields the discrete time derivative that approximates the sensor's yaw rate. This relationship is captured in Equation 5.16:

$$\omega_z = \partial\psi/\partial t \approx \delta\psi/\delta t \tag{5.16}$$

Although correlation-based "template matching" is very popular, it is infamous for its computational cost. Consequently, RADARODO instead utilizes a "point-to-distribution" error metric similar to the NDT- and GMM-based methods discussed in Section 5.2. Specifically, RADARODO extracts representative pixel points from the RADAR image's "distribution" for each measurement. When the next RADAR image is received, RADARODO uses a non-linear optimizer to maximize the correlation of the previous image's points with the current RADAR image.

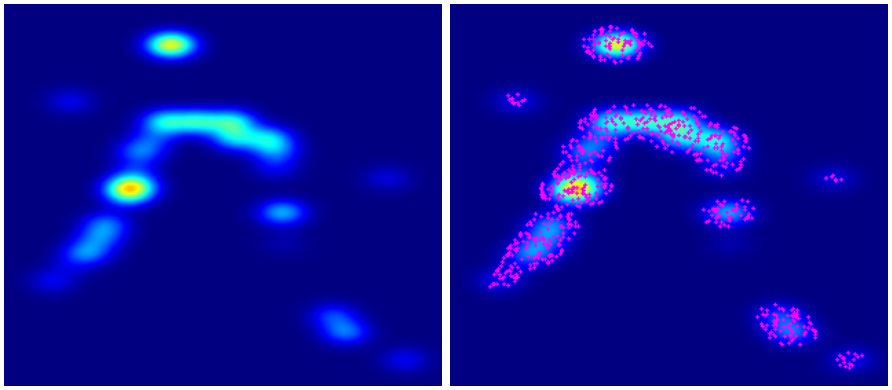
[6] More accurately, the covariance matrices' inverses (i.e. information matrices) were utilized.

Extract Representative Pixel Points

Representative pixel points are extracted from the RADAR image at the end of each RADARODO iteration. These points should robustly and succinctly represent the high-level structure of the previous RADAR image so that it can be used in the next iteration. To do so, points are extracted via uniform sampling. However, to capture the underlying distribution, each point's "weight" is stored as well. These weights are proportional to their pixel's SNR-based intensities, as shown in Equation 5.17:

$$w_{j,k} = I(r,\theta)_{j,k} / I_{max,k} \tag{5.17}$$

where $w_{j,k}$ is the j^{th} point's weight in the k^{th} image, $I(r,\theta)_{j,k}$ is the pixel intensity at the point's location, and $I_{max,k}$ is the maximum pixel intensity within that image. To avoid extracting inconsequential points, only pixels above a dynamic threshold[7] are extraction candidates. To restrain computational costs, there is also an upper limit to the number of extracted points. Figure 5.3 illustrates this process.



(a) Polar RADAR Image.

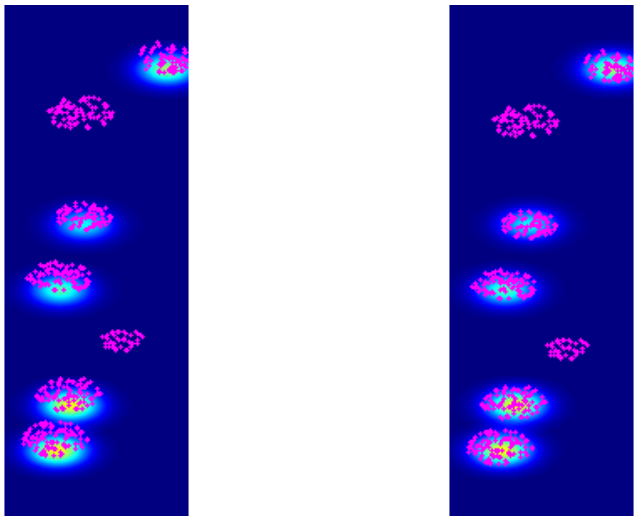
(b) Polar RADAR Image with overlaid extracted pixel points in purple.

Figure 5.3: Cropped polar RADAR image and its representative extracted pixel points (purple). Enlarged sigma values used for visualization purposes.

Transform Image and Points According to the Translational Velocity Estimate

As previously discussed, if the previous points are transformed according to the current translational velocity estimates, only azimuthal angle differences remain. These angle differences then correspond to the sensor's yaw rate. Thus, Equations 5.3 and 5.5 were used to transform the previous image's polar points based on the current translational velocity estimate and the known δt. A visualization of this transformation can be seen in Figure 5.4.

[7]The dynamic threshold was calculated as a pre-set fraction of the image's maximum observed pixel intensity.



(a) Pixel points before transformation. (b) Pixel points after transformation.

Figure 5.4: Cropped polar RADAR image overlaid with the previous image's extracted pixel points (purple) before and after being transformed according to the translational velocity estimate. Enlarged sigma values used for visualization purposes.

When only azimuthal angle differences remain, points should reside in their respective polar pixel rows. Consequently, this decoupling permits one-dimensional convergence along image rows, reducing computational cost and mitigating convergence into incorrect range-based local minima.

Unfortunately, reducing the degrees of freedom makes the process sensitive to translational velocity estimate errors. Specifically, small errors can make points "pass over or under" RADAR image distributions. Fortunately, RADAR images inherently capture range uncertainties, mitigating this possibility. Furthermore, the current polar RADAR image can be blurred to compensate for the expected translational uncertainties.

Since these uncertainties are represented in the sensor-based Cartesian coordinate system, the polar image was first transformed into its Cartesian equivalent. Then, the image was blurred with a representative Gaussian kernel before being transformed back into a polar image. This blurred polar image further relaxes range accuracy requirements by spreading the RADAR image's distributions.[8]

Estimate Rotational Velocity Via Gradient Descent

While it may be more intuitive to estimate rotational velocity by maximizing correlation, the standard for optimization frameworks is "gradient descent" minimization instead. Therefore, this problem was re-framed to cumulatively minimize each j^{th} point's weight error, $e_{w,j}$, as defined in Equation 5.18:

$$e_{w,j} = 1.0 - I(r_j, \hat{\theta}_j)/I_{max,k} \tag{5.18}$$

where $I(r_j, \hat{\theta}_j)$ is the pixel intensity of the RADAR image at the polar coordinates of the transformed pixel point and $I_{max,k}$ is the maximum observed pixel intensity of the current k^{th} image. In other words, a point's error is minimized when it resides in a pixel with the highest possible intensity. To converge to this minimum, it must descend the gradient describing its error's partial derivative. The points' cumulative minimum then corresponds to the correct heading change, a necessary variable to estimate the sensor's yaw rate. Therefore, the partial derivative of a point's error with respect to the sensor's heading change can be expressed using the chain rule as seen in Equation 5.19:

[8]Gaussian blurring is also critical for mitigating the noisy high-frequency artifacts common in real RADAR images. Specifically, the utilized polar image must provide a smooth and continuous surface for differentiation.

$$\frac{\partial e_{w,j}}{\partial \delta\psi} = \frac{\partial e_{w,j}}{\partial \theta}\frac{\partial \theta}{\partial \delta\psi} \tag{5.19}$$

The partial derivative of a point's error with respect to azimuthal angle can be calculated by taking the partial derivative of Equation 5.18 to yield Equation 5.20:

$$\frac{\partial e_{w,j}}{\partial \theta} = \frac{-1.0}{I_{max,k}}\left(\frac{\partial I(r_j, \hat{\theta}_j)}{\partial \theta}\right) \tag{5.20}$$

The partial derivative of the target's angle with respect to the sensor's heading change can be calculated by taking the partial derivative of Equation 5.5 to yield Equation 5.21:

$$\frac{\partial \theta}{\partial \delta\psi} \approx \frac{\frac{\partial \theta}{\partial t}}{\frac{\partial \psi}{\partial t}} = \frac{\partial \dot{\theta}}{\partial \omega_z} = -1.0 \tag{5.21}$$

Finally, combining Equations 5.19, 5.20, and 5.21 yields the partial derivative of a point's error with respect to the sensor's heading change, as seen in Equation 5.22:

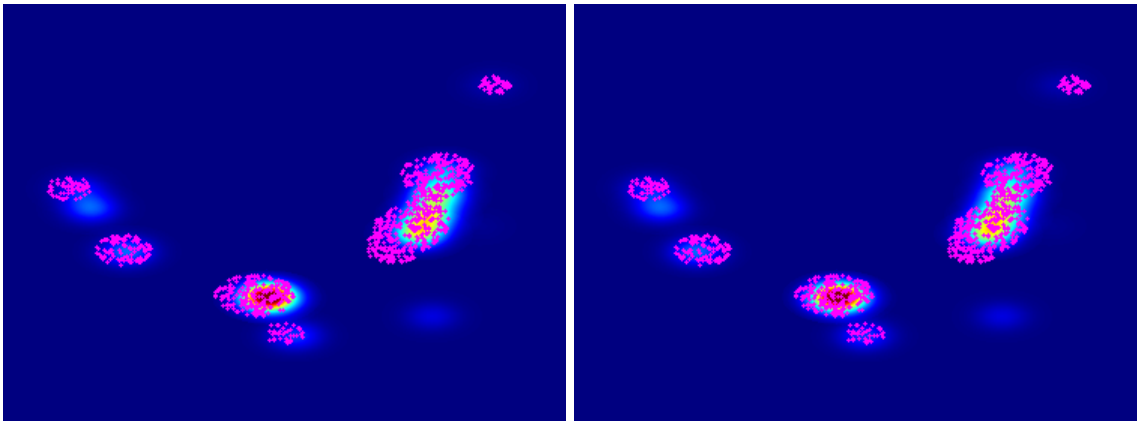
$$\frac{\partial e_{w,j}}{\partial \delta\psi} = \frac{1.0}{I_{max,k}}\left(\frac{\partial I(r_j, \hat{\theta}_j)}{\partial \theta}\right) \tag{5.22}$$

where $\partial I(r_j, \hat{\theta}_j)/\partial\theta$ describes the partial derivative of the RADAR image's pixel intensity with respect to target angle. Since it is based on a discrete measurement, it does not have an analytical derivative. Rather, $\partial I(r_j, \hat{\theta}_j)/\partial\theta$ is a numerical (spatial) derivative. This derivative can be efficiently calculated with computer vision techniques. Specifically, this derivative was calculated by convolving a symmetric Sobel derivative kernel over the polar RADAR image.

For this chapter's implementation, the error metric and its partial derivative was integrated into g^2o [119] as well. Specifically, each extracted point contributed its own constraint to estimate the sensor's heading change. Furthermore, each constraint was weighted according to Equation 5.17. These weights were critical since the error metric makes points "compete" to maximize their projected intensities. Yet, ego-motion estimation depends on consecutive measurements that are relatively consistent. Thus, a high SNR region (i.e. target) from the previous frame is expected to correspond with a similarly-high SNR region in the current frame. Consequently, to correctly estimate heading change, it is imperative that points project onto pixels with similar intensities. So, these weights ensure that highly-weighted points yield more influence as they are more likely to correspond to higher intensity regions.

Along with the heading change estimate, this framework also calculated the estimate's expected uncertainty. This value most notably depends on the uncertainty captured in the RADAR image and the structure of the observed environment.

An example of the optimization's input and resulting heading change estimate can be visualized in Figure 5.5. Note how an unconstrained convergence may have attempted to incorrectly reduce the points' ranges in addition to its azimuthal angles. However, instead, RADARODO's decoupled estimation and constrained optimization produced the accurate estimate.



(a) Post-transformation pixel points before rotational velocity estimate.

(b) Pixel points after convergence to the correct rotational velocity estimate.

Figure 5.5: Cropped polar RADAR image overlaid with the previous image's extracted pixel points (purple) before and after converging to the correct rotational ego-motion estimate. Enlarged sigma values used for visualization purposes.

5.4 Validation

RADARODO was validated using real data collected from an instrumented semi-truck. Specifically, the utilized RADAR data was supplied by a front-mounted Delphi ESR 2.5 RADAR. This RADAR is a multimode electronically scanning RADAR that simultaneously provides a mid-range and long-range scan [117].[9] As previously stated, the RADAR images were emulated but were based on real measurements. For use as a reference, GNSS Doppler velocity measurements were provided by an onboard Hemisphere A325 GNSS receiver [120]. Therefore, the truck's longitudinal velocities were calculated to be its measured global velocities' vector norms. This was deemed acceptable as minimal side-slip could be assumed from the testing conditions. Furthermore, the truck's headings were estimated from the global velocity vectors. Thus, the reference yaw rates were estimated by filtering the discrete time derivatives of these heading estimates. This validation analyzed an urban route with speeds up to 20 m/s (72 kph).

5.4.1 Estimate Accuracy

It is noteworthy that, up until this point, RADARODO has only been used to estimate the *sensor's* velocities. However, the true goal is to estimate the ego-vehicle's motion. The ego-vehicle's motion can be estimated by transforming the estimated sensor motion if the rigid transformation between the two are known. For simplicity, their axes are assumed to be aligned to yield the planar transformation in Equation 5.23:

$$\underbrace{\begin{bmatrix} v_{x,V} \\ v_{y,V} \\ \omega_{z,V} \end{bmatrix}}_{Vehicle\ Motion} = \begin{bmatrix} 1 & 0 & +y_{S/V} \\ 0 & 1 & -x_{S/V} \\ 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} v_{x,S} \\ v_{y,S} \\ \omega_{z,S} \end{bmatrix}}_{Sensor\ Motion} \tag{5.23}$$

where the vehicle's ego-motion is denoted by V subscripts, the sensor's motion is denoted by S subscripts, and $x_{S/V}$, $y_{S/V}$ describe the sensor's offset from the vehicle's reference frame along the $x-$ and $y-$ axes, respectively. Note that the vehicle's and sensor's yaw rates, ω_z, are equivalent.

[9] The mid-range scan has a max. range of 60 m and a $\pm 45°$ field-of-view. The long-range scan has a max. range of 174 m and a $\pm 10°$ field-of-view. These scans are provided simultaneously every 50 ms [117].

Equation 5.23 can be used to transform the sensor's longitudinal motion into the vehicle's longitudinal motion. In this case, their longitudinal velocities were equal since their axes were aligned and $y_{S/V} = 0$. Therefore, the RADARODO and reference estimates are plotted in Figure 5.6 for comparison. Figure 5.6 demonstrates that RADARODO accurately measures longitudinal velocities at all observed speeds. While it is difficult to see at this scale, RADARODO's estimates were corrupted by less noise than their GNSS Doppler counterparts.

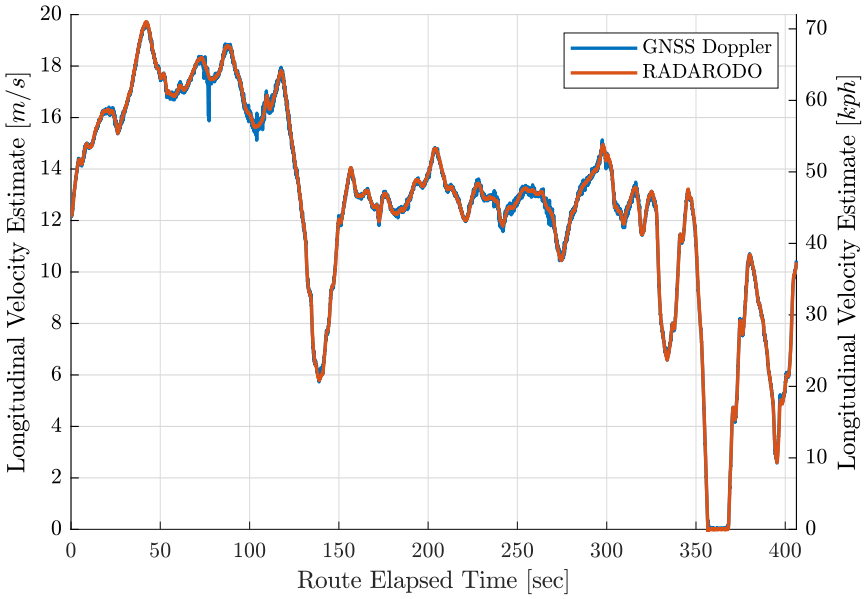


Figure 5.6: RADARODO and GNSS Doppler longitudinal velocity estimates

For a better understanding of the expected uncertainty, RADARODO's errors with respect to the GNSS Doppler measurements are plotted in Figure 5.7. Figure 5.7 demonstrates that RADARODO's longitudinal velocity estimates were very accurate, with a median expected standard deviation of 0.02 m/s (≈ 0.07 kph). The translucent red region illustrates this expected uncertainty. As a further validation, RADARODO's expected longitudinal velocity uncertainty is consistent with the results reported by many similar methods [73, 74, 81]. Therefore, at these low expected uncertainties, the plotted errors appear to be largely due to the reference GNSS Doppler measurement's noise.

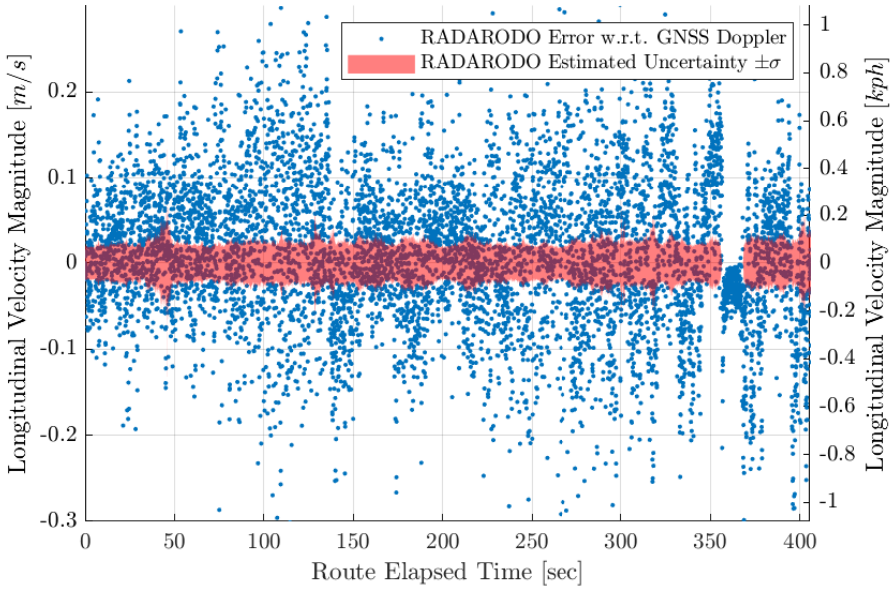


Figure 5.7: RADARODO longitudinal velocity estimate error with respect to GNSS Doppler and expected uncertainty

Since the sensor and vehicle both share a rigid body, they also share their rotational motion. Thus, RADARODO directly estimates the vehicle's yaw rate. For comparison, the GNSS Doppler-based and RADARODO yaw rate estimates are plotted in Figure 5.8. This figure shows that RADARODO successfully detects and estimates the yaw rate. However, RADARODO's noise is significantly affected by the consistency and structure of the observed environment. This is better demonstrated by plotting RADARODO's error with respect to the GNSS Doppler-based estimates and expected uncertainty, as seen in Figure 5.9.

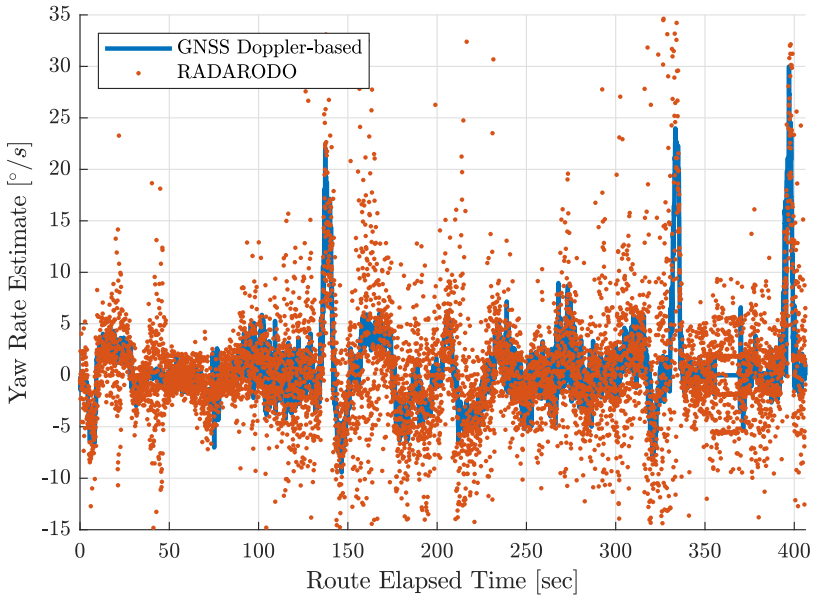


Figure 5.8: Yaw rate estimates for RADARODO and GNSS Doppler-based measurements

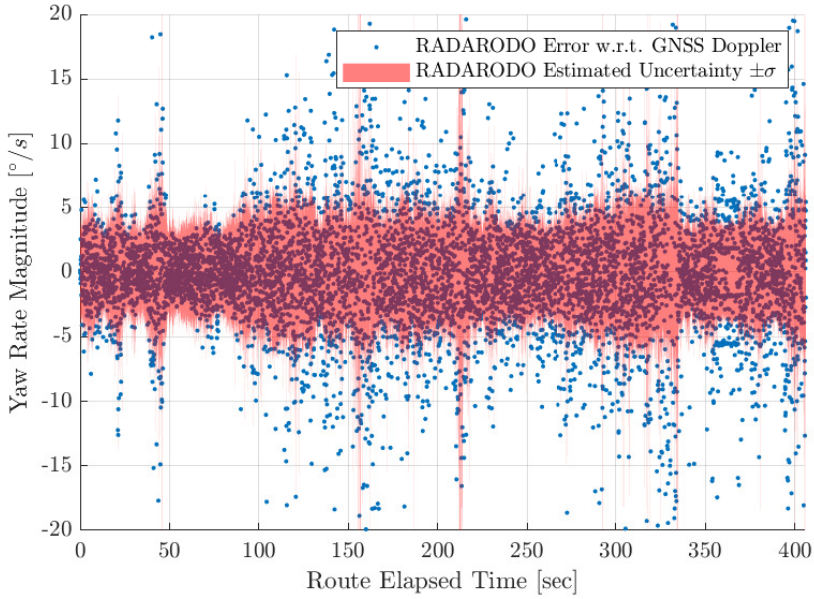


Figure 5.9: Yaw Rate Error and Standard Deviation with respect to GPS Doppler

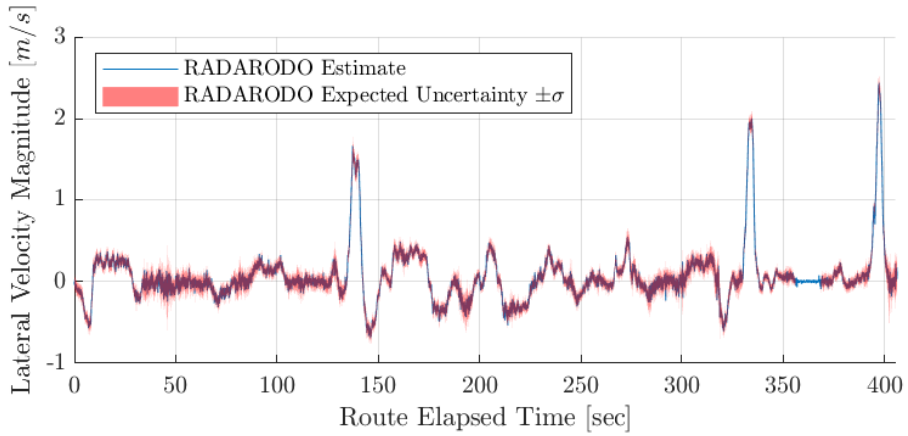
RADARODO's median expected yaw rate uncertainty was 4.8 $^\circ/s$, which may seem alarming at first. Yaw rates are essentially estimated by taking the discrete time derivatives of target angles, a spatial measurement that is one of RADAR's characteristic weaknesses. Therefore, as an added check, RADARODO's uncertainty can be compared to its theoretical counterpart. Assuming that the RADAR's azimuthal angle uncertainty is the only source of error, the theoretical uncertainty of purely spatial rotational motion estimates, $\sigma_{\hat{\omega}_z}$, can

be calculated via Equation 5.24:

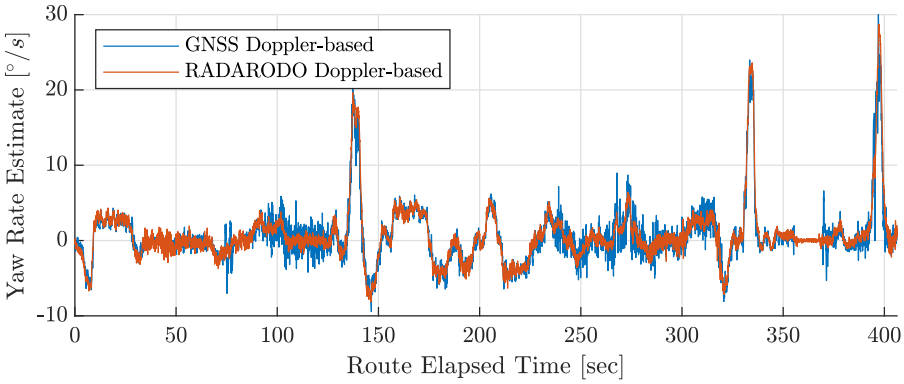
$$\sigma_{\hat{\omega}_z} = \sqrt{\frac{2}{N}} \frac{\sigma_\theta}{\delta t} \tag{5.24}$$

The target centroids' azimuthal angle standard deviation, σ_θ, was reported by [116] to be 1.0°. For the given time between measurements ($\delta t = 0.05$ *sec.*) and the mean observed number of inlier targets ($N = 35$), the expected yaw rate uncertainty is approximately 4.8 $^\circ/s$. This is consistent with the validation results. Thus, this spatial approach appears to be near its sensor-dependent limits. Theoretically, however, *directly* analyzing the RADAR images would increase accuracy as it avoids the errors induced by centroid extraction.

Currently, a direct yaw rate estimate comparison with Rapp et al.'s method [73, 74] is infeasible for several reasons. First, they did not report errors when estimating full planar motion from a single RADAR. However, when multiple RADARs were used, they reported a lower uncertainty of $0.6 - 0.7$ $^\circ/s$ [73, 74]. Secondly, unlike RADARODO, they fuse their Doppler and spatial data *internally* within their algorithm, most likely explaining their lower uncertainties. In a front-mounted configuration, fused estimates depend on the way the *sensor's* lateral Doppler-based translational velocity measurements are utilized. RADARODO's estimates can be seen in Figure 5.10a.



(a) RADARODO's sensor lateral velocity estimates and expected uncertainty



(b) RADARODO Doppler-based and GPS Doppler-based yaw rate estimates

Figure 5.10: Sensor lateral velocity estimates and its possible use for yaw rate estimates, given the "zero side-slip" motion assumption

For this single front-mounted RADAR, if and only if zero side-slip can be assumed, Equation 5.23 could be used to *also* estimate yaw rate from RADARODO's sensor lateral velocity measurements. These Doppler-

based yaw rate estimates are shown in Figure 5.10b. Their expected uncertainties can also be calculated via Equation 5.25:

$$\sigma_{\hat{\omega}_z} = \sigma_{\hat{v}_{y,s}} / x_{S/V} \tag{5.25}$$

Equation 5.25 demonstrates that these uncertainties are proportional to the uncertainties of the sensor's highly-accurate lateral velocity estimates, $\sigma_{\hat{v}_{y,s}}$. Consequently, the Doppler-based yaw rate estimates usually correspond to uncertainties factors lower than their spatial counterparts. Thus, for the related methods that depend on a joint metric, the relative uncertainties dictate that estimates are largely driven by Doppler measurements anyway.

Regardless, when these foundational assumptions are no longer valid, spatial data plays a crucial role. Equation 5.25 demonstrates that Doppler-based yaw rate estimates can be highly uncertain at smaller lever-arm offsets. More importantly, spatial data is critical for estimating motion in all planar degrees of freedom when side-slip is possible. Therefore, Equation 5.23 can be used with RADARODO's spatial yaw rate estimates to estimate *vehicle* ego-motion in all planar degrees of freedom from a single RADAR sensor without depending on a lever-arm offset. Despite this, unfortunately, Equation 5.23 also demonstrates that this will propagate the substantial yaw rate estimate uncertainties in a manner proportional to the lever-arm offset. For this reason, even though RADARODO can provide unique *vehicle* ego-motion estimates, its measurements are better utilized in a "tightly-coupled" sensor fusion framework. This will be discussed further in Section 5.5.

5.4.2 Computational Performance

To evaluate RADARODO's computational performance, its subprocesses' runtimes were plotted in Figure 5.11. These runtimes were based on a Robotic Operating System (ROS) [121] C++ node running on an Intel Core i7-7500U 64-bit CPU (2.70 GHz). Overall, the RADARODO process had a mean runtime of approximately 13 ms ($\approx$ 77 Hz). If real RADAR images were available, image emulation would not be required, reducing the runtime even further. This low cost easily permits real-time operation. Furthermore, RADARODO's measurement frequency of 77 Hz is significantly higher than the $11.7 - 17$ Hz reported for Rapp et al.'s method [73, 74].[10] RADARODO's significantly lower runtime can be largely attributed to the fact that its spatial-based registration has only one degree of freedom while Rapp et al.'s has three.

[10]It should be noted that Rapp et al. validated their methods in a different processing environment than this chapter's RADARODO validation.

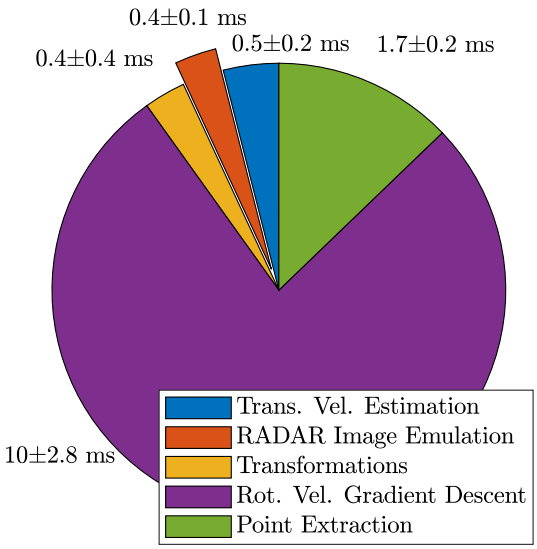


Figure 5.11: Runtimes for RADARODO subprocesses.

It should be noted that all subprocesses except for "Translational Velocity Estimation" exist solely to support spatial rotational velocity estimation. Strikingly, the Doppler-based translational velocity estimation incurs a minimal computational cost relative to its spatial rotational velocity counterpart.

5.5 Implementation Discussion

This section discusses how RADARODO could be implemented in practice. As previously discussed, Rapp et al.'s methods internally fuse Doppler measurements, spatial measurements, and the vehicle-sensor transformation to yield a unique *vehicle* ego-motion estimate [73, 74]. Yet, Section 5.4 demonstrates that, in terms of accuracy and computational performance, there is a wide gap between Doppler- and spatial-based techniques. That is why, conversely, RADARODO decoupled Doppler- and spatial-data to yield the full *sensor* motion estimate. This was specifically done to better utilize the low-level data of decoupled motion estimates and uncertainties. Overall, the data is still fused later, but RADARODO calculates and provides it in a way that is better suited for "tightly-coupled" sensor fusion frameworks.

This section details two proposed implementations for this data: velocity estimation and pose estimation. For both, using *a priori* estimates from the overarching sensor fusion framework resolves the ambiguities of purely Doppler-based methods without depending on motion assumptions. Furthermore, it only requires one RADAR, although the framework greatly (and easily) benefits by its extension to multiple. Essentially, a "tightly-coupled" framework depends on measurements of *sensor's* motion, not the ego-vehicle's. Although it does require a complementary sensor, e.g. an inertial measurement unit (IMU), many suitable sensors are guaranteed on today's vehicles. Thus, the very accurate Doppler-based measurements can be a powerful and complementary asset to conventional sensor suites. This implementation alone can meet the estimation needs for a large majority of vehicles.

Therefore, what is the purpose of spatial rotational motion estimates, particularly since the "tightly-coupled" solution can already resolve ambiguities to correct rotational motion? Despite being associated with a relatively larger uncertainty, its inclusion still reduces the overall framework's uncertainties. Furthermore, the uncertainty gap between spatial- and Doppler-based yaw rate estimates is reduced for smaller lever-arm

offsets. Combined, an implementation's computational budget determines whether the spatial rotational motion estimates have a benefit-to-cost ratio that supports its implementation. To that end, RADARODO aims to increase the benefit-to-cost ratio by reducing its computational expense.

5.5.1 Velocity Estimation

The aforementioned "tightly-coupled" framework would utilize the residuals between the sensor motion estimates and the "expected" sensor motion given the *a priori* vehicle motion estimate [82]. The expected sensor motion can be calculated via Equation 5.26:

$$\underbrace{\begin{bmatrix} v_{x,S} \\ v_{y,S} \\ \omega_{z,S} \end{bmatrix}}_{Sensor\ Motion} = \begin{bmatrix} 1 & 0 & -y_{S/V} \\ 0 & 1 & +x_{S/V} \\ 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} v_{x,V} \\ v_{y,V} \\ \omega_{z,V} \end{bmatrix}}_{Vehicle\ Motion} \tag{5.26}$$

Note how the coupling present in the transformation matrix enables yaw rate estimate corrections. This methodology is very similar to Rapp et al.'s "expected Doppler" weighting [73, 74]. However, in this implementation, the framework's *a priori* uncertainties may be less than the RADAR's spatial data can provide, enabling the very accurate Doppler measurements to be better utilized.

In a Kalman Filter implementation, an IMU can be very complementary to RADAR because RADAR measurements can help estimate the IMU's walking biases. An actively calibrated IMU has the potential to provide both fast and accurate ego-motion estimates. Even the spatial yaw rates measurements are useful because their inclusion, over time, help accelerate convergence to these quasi-static biases.

5.5.2 Pose Estimation

Pose graphs are a popular localization framework for calculating the "most likely" estimates of vehicle pose. So, to effectively utilize this framework, RADAR pose nodes could be created over time using its *sensor* motion estimates. Each current RADAR pose node would then be connected to its previous one via an edge constraint. These frequent "odometry" constraints would be determined by its Doppler-based translations, but be unconstrained with regards to heading.

Furthermore, each RADAR pose node would be fully constrained to its corresponding vehicle pose node by its lever-arm offset. Thus, this tight coupling allows other measurements of the vehicle's heading (or yaw rate) to also correct the RADAR's headings. Notably, a measurement of a map landmark may determine the most likely combination of yaw or side-slip that caused the RADAR's lateral motion. More impressively, even sporadic landmark measurements can retroactively correct previous pose changes.

In fact, RADAR's spatial data may be best utilized within this pose graph framework. As their name implies, these measurements capture a pose in space, independent of the motion that brought them there. Thus, a pose graph could utilize these slower spatial registrations for their less frequent "back-end" optimizations. Specifically, the efficient algorithm described in Section 5.3.3 fundamentally estimates heading changes. So, RADARODO's algorithm could be deployed on non-concurrent measurements to restrain heading drift, particularly when it exceeds the RADAR's angular resolution.

5.6 RADARODO Conclusion

This chapter presented RADARODO, a novel real-time RADAR-based ego-motion estimation algorithm. RADARODO is differentiated from similar techniques because it decouples translational and rotational motion estimation and directly analyzes RADAR images. Its validation results on real data produced longitudinal velocity estimates with a low expected uncertainty of 0.02 m/s ($\approx$ 0.07 kph) and yaw rate estimates with an expected uncertainty of 4.8 $^\circ/s$. These uncertainties were consistent with similar methods and/or near the RADAR's sensor-dependent limits. Overall, RADARODO's methodology is particularly suited for integration within "tightly-coupled" sensor fusion frameworks.

Chapter 6

MonoRAD: Monocular Camera and RADAR-based Ego-Motion Estimation

Abstract – Ego-motion estimation provides a critical foundation for many autonomous vehicles' localization frameworks, particularly when wheel odometry and/or global pose measurements are unreliable. This chapter demonstrates how monocular camera and RADAR integration can be well-suited for ego-motion estimation due to their complementary strengths and weaknesses. Specifically, this chapter proposes MonoRAD, a novel decoupled ego-motion estimation algorithm that integrates these popular and complementary sensors. This approach depends on a single RADAR's Doppler measurements for translational ego-motion estimation and monocular camera images for rotational ego-motion estimation. While MonoRAD's implementation is based on widely accepted techniques, it is novel in its integration of select submodules from those techniques. The chapter's real-world validation demonstrated that MonoRAD estimated longitudinal velocities with an expected standard deviation of 0.02 m/s (0.07 kph), estimated yaw rates with an expected standard deviation of 0.66 $deg./s$, and yielded a mean dead reckoning percent error of 2.78% at a mean runtime of 36 ms.

6.1 Introduction to MonoRAD

Many autonomous vehicle modules are critically dependent on fast and accurate pose estimates. Since Global Navigation Satellite System (GNSS) estimates have severe accuracy, frequency, and reliability limitations, robust autonomous vehicles typically localize by matching observed landmarks with those stored in an *a priori* map [4,107]. Unfortunately, many environments lack distinct and consistent landmarks. Furthermore, pose graph optimization frameworks often rely on velocity estimates to impose smoothness constraints and for fast *a priori* estimates [11, 12]. Consequently, ego-motion estimation is fundamentally important to the localization framework [3].

While wheel odometry offers a natural solution, its estimates are, unfortunately, very susceptible to tire slip [108, 109]. The inertial measurement unit (IMU) also provides a solution, yet IMU measurements are generally corrupted by walking biases [10]. To circumvent these errors, camera-based "Visual Odometry" (VO) [7, 17, 23, 110] and/or LIDAR-based odometry [46, 122] methods can be deployed. Due to their complementary nature, they are often combined within a sensor fusion framework [108].

At the time of this writing, this chapter is under review for publication in the IEEE Transactions on Intelligent Vehicles (T-IV).

Even though cameras and LIDARs dominate ego-motion estimation research, RADARs offer a unique advantage: the ability to "instantaneously" measure relative radial velocities via Doppler shift. Doppler-based velocity estimation can be very accurate [19, 20, 115]. Unlike other methods, it is not dependent on (possibly inaccurate) estimates of depth. Furthermore, it does not require computationally-intensive feature tracking [81, 123].

More specifically, Doppler-based methods directly estimate *translational* ego-motion. This is because Doppler sensors measure range rates over their field of view, yielding relative rotational motion unobservable. Consequently, Doppler-based methods cannot directly measure rotational ego-motion [82]. There are strategies to produce rotational estimates, yet they require stringent vehicle motion assumptions, sensors placed at lever-arm offsets from the reference frame, and/or multiple distributed sensors [80, 81]. For non-holonomic vehicles, "zero side-slip" is the vehicle motion assumption. If these conditions are not met, the vehicle's full motion state is either unobservable or ambiguous from Doppler data [82]. In these scenarios, the RADAR's spatial data (i.e. range and azimuth of observed targets) can be utilized to resolve ambiguities [72–74]. Specifically, observing the bearing changes of RADAR targets over time is critical for extracting the sensor's rotational motion.

Unfortunately, however, one of RADAR's characteristic weaknesses is its poor angular resolution [19]. Therefore, this inaccuracy propagates to the resulting spatial-based ego-motion estimates, particularly in regards to rotational ego-motion. Fortunately, many RADAR-equipped automotive vehicles also possess a sensor whose angular resolution is its characteristic strength: the monocular camera [19].

At its core, the monocular camera is a bearing sensor. Consequently, Monocular Visual Odometry methods cannot immediately estimate the depth of points in its environment. Thus, without this scaling reference, its translational velocities can only be estimated up to an unknown scale factor. While methods like Visual Inertial Odometry (VIO) [40–42] and Visual Simultaneous Localization and Mapping (V-SLAM) [28, 29, 43] can simultaneously estimate these depths and scaling factors, they require an IMU and/or feature tracking over many frames. In automotive environments populated with many dynamic agents, this can present a very challenging data association problem. Unfortunately, for data with unknown associations, robustness comes at a computational cost. Conversely, however, rotational motion can be estimated by Monocular Visual Odometry with just two consecutive images [17].

These insights reveal that monocular cameras and RADARs are complementary in nature with regards to ego-motion estimation. To demonstrate the integration of these sensor capabilities, this chapter proposes MonoRAD, a decoupled ego-motion algorithm that depends on a single RADAR's Doppler measurements for translational ego-motion estimation and monocular camera images for rotational ego-motion estimation. While MonoRAD is based on widely accepted algorithms, it is novel in its integration of select submodules from those techniques. The results of this integration show that MonoRAD can estimate the vehicle's full planar ego-motion from sensors currently available on many affordable production vehicles.

6.2 Related Work

Due to their complementary nature, monocular camera and RADAR measurements have long been used in conjunction in advanced driver assistance systems (ADAS). Interestingly, however, research on their fusion has mostly focused on perceiving the ego-vehicle's surrounding environment, not for estimating the vehicle's own ego-motion [109]. Some authors have used RADAR or SONAR sensors in conjunction with Monocular VO. However, these sensors were only used to estimate the vehicle's altitude, enabling VO estimates in absolute scale [124, 125]. Unlike these methods (and many ADAS algorithms), MonoRAD does not establish correspondences between RADAR and camera measurements. Furthermore, MonoRAD does not need to estimate the depth of points in its environment.

While MonoRAD is novel in its integration of submodules from RADAR- and camera-based methods, MonoRAD's rotational ego-motion estimation submodule itself is based on current Monocular VO techniques. It utilizes the optimized algorithms in OpenCV, the open-source computer vision library [126]. Specifically, corner features are extracted using the methodology by Shi et al. [31], tracked using Lucas-Kanade optical flow [35–37], and analyzed by the five-point algorithm to estimate the essential matrix within a RANSAC

scheme [127, 128]. However, unlike V-SLAM, MonoRAD only focuses on estimating current ego-motion, so does not store features or poses from more than two images [17, 43]. Consequently, MonoRAD also differs because it does not estimate translational ego-motion from its images.

For this reason, MonoRAD's translational ego-motion submodule is based on current RADAR-based techniques. To generate an initial estimate, MonoRAD emulates the Kellner et al.'s "Sinefit" algorithm by calculating the least-squares fit of the RADAR's measured targets' radial velocities and azimuthal angles [80]. MonoRAD similarly utilizes a RANSAC scheme to classify targets as static or dynamic. Furthermore, MonoRAD also emulates Kellner et al.'s subsequent techniques for refining this initial estimate via non-linear optimization [81, 118].

When zero side-slip cannot be assumed, the vehicle's ego-motion can be ambiguous from a single RADAR's Doppler measurements. Therefore, like MonoRAD, extensions of Kellner et al.'s research attempted to resolve this ambiguity. Specifically, Barjenbruch et al. and Rapp et al. utilized the RADAR's spatial data [72–74]. While this spatial data provides redundant information regarding the *RADAR's* translation, it uniquely provides initial estimates of its rotation. Conversely, MonoRAD does not analyze the RADAR's spatial data, but instead utilizes the rotational estimates from its camera images. Comparing techniques for correcting the *vehicle's* ego-motion from sensor estimates is worthy of the more nuanced discussion in Section 6.4.

6.3 Algorithm Overview

This section details the MonoRAD algorithm via its subprocesses. MonoRAD's C++ source code is available online at `https://bitbucket.org/chrisdmonaco/monorad`.

6.3.1 Rotational Ego-Motion Estimation

MonoRAD estimates the monocular camera's rotational ego-motion by tracking image features over time. To do so, it first extracts so-called "good features to track" for each new image [31]. To maintain computational consistency, the number of extracted features are capped by a pre-set maximum. Therefore, to ensure accurate ego-motion estimation, it is essential that these corner features are utilized effectively.

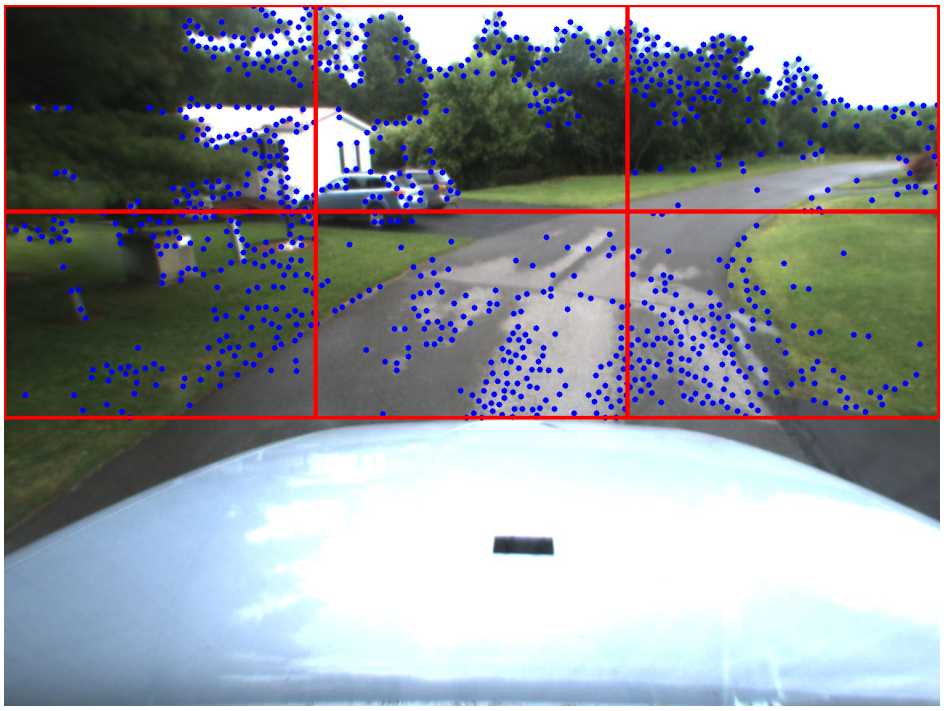


Figure 6.1: Extracted corner features (blue). To promote their even distribution throughout the image, the extraction process was executed separately for each image grid section (red). The grid was designed to exclude the rigidly connected vehicle hood.

Figure 6.2: The corner features (blue) and corresponding optical flow tracks (green) utilized to estimate the monocular camera's rotational ego-motion. Only static inliers shown.

Consequently, it is critical that these features are evenly distributed throughout the image. Features should be extracted so that they produce diverse optical flow vectors in all directions surrounding the focus of expansion. Otherwise, if only local regions of nearly homogeneous optical flow vectors are available, the camera's ego-motion may be ambiguous. To that end, MonoRAD promotes distribution at two different scales. First, MonoRAD divides the image by turning it into an equally-spaced grid. Corner features are then independently extracted within each grid section up to its allotted share. However, even within these sections, features can be extracted as dense clusters, demanding computational resources while offering limited new information. Therefore, MonoRAD ensures that a feature is not extracted if it is within a preset distance to a stronger one. Figure 6.1 shows these extracted features (and their grid sections) for one of the captured images.

When the next image arrives, the previously extracted features must be matched or tracked in the next image. V-SLAM methods often opt to extract ORB features for feature matching [28, 29, 34]. This is necessary to robustly establish feature correspondences from many distributed poses over a long period of time. However, ego-motion estimation has the advantage in that its features cannot feasibly move far or change their appearance between image measurements. Therefore, particularly in this implementation, the search radius and consistency requirements are much more relaxed [32]. For this reason, MonoRAD utilizes the Lucas-Kanade optical flow algorithm for feature tracking [35–37].

Once feature correspondences have been established, the essential matrix can be estimated. The essential matrix contains the relative transformation between its two (and in this case, consecutive) camera poses. The essential matrix is derived from epipolar geometry, i.e. the geometric relationship that the three vectors connecting two camera poses and their mutually observable world point form an "epipolar" plane. This geometric relationship is visualized in Figure 6.3.

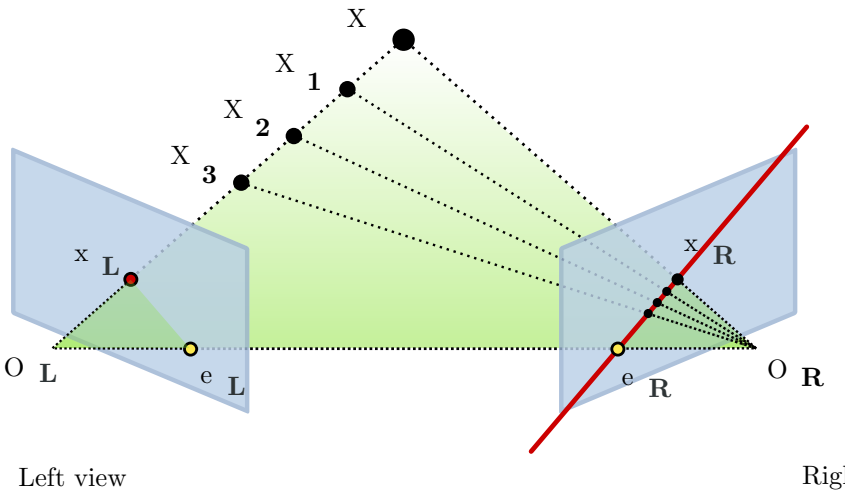


Figure 6.3: The epipolar plane that relates two camera poses and their mutually observable world point [2].

This relationship is mathematically defined in Equation 6.1:

$$\tilde{p}_k^T \boldsymbol{E} \tilde{p}_{k-1} = 0 \tag{6.1}$$

where $\boldsymbol{E}$ is the essential matrix and˜denotes normalized coordinates. Specifically, $\tilde{p}$ contains the homogeneous normalized image coordinates of the same world point in the k^{th} and $(k-1)^{th}$ images. This is defined in Equation 6.2:

$$\tilde{p} = [\tilde{u}, \tilde{v}, 1]^T = \boldsymbol{K}^{-1}[u, v, 1]^T \tag{6.2}$$

where u and v are the world point's image coordinates and $\boldsymbol{K}$ is the monocular camera's intrinsic matrix. The essential matrix is mathematically defined by Equation 6.3:

$$\boldsymbol{E} \simeq [\boldsymbol{t}]_\times \boldsymbol{R} \tag{6.3}$$

where $\boldsymbol{R}$ represents the rotation matrix for the $k-1$ to k camera pose transformation. The $\simeq$ in Equation 6.3 denotes equivalence up to a (unknown) nonzero scale factor. $[\boldsymbol{t}]_\times$ is a skew-symmetric cross-product matrix corresponding to the translation between the $k-1$ and k camera poses. It is defined in Equation 6.4:

$$[\boldsymbol{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{6.4}$$

where t_x, t_y, and t_z are the individual components of the camera's translation along their respective axes [12, 17].

The six components of the relative pose can then be extracted from the estimated essential matrix. However, Equation 6.3 states that the translation vector can only be recovered up to an unknown scale. Therefore, there are actually only five degrees of freedom. Consequently, to handle the dynamic environment, MonoRAD utilized the five-point algorithm within a RANSAC scheme to estimate the essential matrix [127, 128]. Specifically, the inlier threshold was set as a pre-set maximum pixel distance between the feature and its estimated epipolar line, i.e. the image projection of the epipolar plane.

While methods for recovering the relative pose components from the essential matrix are beyond the scope of this chapter,[2] each essential matrix could conceivably correspond to four possible relative poses (two possible rotations and two possible unit translation vectors). Often, the correct relative pose is determined

[2]For further details, the author points the reader to [127].

via the "cheirality check," i.e. checking that all triangulated 3D points reside in front of both cameras [127]. Unfortunately, this triangulation fails when the camera's translation is negligible, e.g. when the camera is stationary.

Fortunately, in this implementation, yaw changes between images are limited due to vehicle motion constraints. Therefore, MonoRAD extracts both rotation matrices and decomposes them into their equivalent Euler angle rotation vectors. Once these vectors are transformed to align with the vehicle's coordinate system, MonoRAD only considers the vector with the smallest relative yaw magnitude to be valid. Dividing this vector by the time between images, δt, yields the rotational ego-motion estimate, $\vec{\Omega} = [\omega_x, \omega_y, \omega_z]$.

6.3.2 Translational Ego-Motion Estimation

MonoRAD estimates the RADAR's translational ego-motion via its Doppler range rate measurements. These range rate measurements can be analytically described for planar motion by first taking the time derivative of a target's range, as seen in Equation 6.5:

$$\frac{\partial}{\partial t}\left(r_i^2\right) = \frac{\partial}{\partial t}\left(x_i^2 + y_i^2\right) \Rightarrow \dot{r}_i = \frac{x_i}{r_i}\dot{x}_i + \frac{y_i}{r_i}\dot{y}_i \tag{6.5}$$

where r_i is the range of the i^{th} target from the RADAR in its xy-plane, x_i is the position of the target along the RADAR's longitudinal x-axis, y_i is the position of the target along the RADAR's lateral y-axis, $\dot{r}_i$ is the relative velocity of the target with respect to the RADAR along its radial axis (i.e. range rate), and $\dot{x}_i$, $\dot{y}_i$ are the time derivatives of the target's position in the RADAR's Cartesian coordinate system. If the target is static, the time derivative of its relative position is defined by the RADAR's planar ego-motion, as described by Equation 6.6:

$$\begin{aligned} \dot{x} &= -v_x + y_i\omega_z \\ \dot{y} &= -v_y - x_i\omega_z \end{aligned} \tag{6.6}$$

where ω_z is the RADAR's rotational velocity about its vertical z-axis (i.e. yaw rate) and v_x, v_y are the RADAR's translational velocities along its longitudinal and lateral axes, respectively. Since $x_i = r_i \cos\theta_i$ and $y_i = r_i \sin\theta_i$, combining Equations 6.5 and 6.6 yields the relationship between the target's range rate measurement and the RADAR's ego-motion, as shown in Equation 6.7:

$$\dot{r}_i = -\cos\theta_i\, v_x - \sin\theta_i\, v_y \tag{6.7}$$

where θ is the azimuthal angle of a target from the RADAR's x-axis. Note how all terms related to rotational ego-motion, ω_z, have cancelled each other out. This analytically demonstrates that Doppler measurements cannot directly measure rotational ego-motion. The linear system in Equation 6.7 can be represented as a matrix for all targets as shown in Equation 6.8:

$$\begin{bmatrix} -\cos\theta_1 & -\sin\theta_1 \\ -\cos\theta_2 & -\sin\theta_2 \\ \vdots & \vdots \\ -\cos\theta_N & -\sin\theta_N \end{bmatrix} \underbrace{\begin{bmatrix} v_x \\ v_y \end{bmatrix}}_{\vec{v}_S} = \begin{bmatrix} \dot{r}_1 \\ \dot{r}_2 \\ \vdots \\ \dot{r}_N \end{bmatrix} \tag{6.8}$$

where 1, 2, ... N refers to the RADAR's individual extracted targets and $\vec{v}_S$ denotes the sensor's (RADAR's) translational ego-motion vector.

Similar to [80], MonoRAD solves this linear system using data from extracted RADAR targets. Since automotive environments are highly dynamic, MonoRAD similarly deploys a RANSAC scheme to classify targets as static or dynamic. Specifically, outliers are rejected if their measured range rates disagree with the translational ego-motion consensus. Note that at least two range rate measurements from different azimuths are required to generate a valid estimate. Consequently, MonoRAD deploys a "two-point" RANSAC scheme.

Once the static inliers have been identified, the RADAR's translational ego-motion is estimated via Least Squares Regression based on Equation 6.8.

While this initial estimate is often sufficient, its calculation is based on an incorrect implicit assumption. Specifically, it assumes that the RADAR has perfect measurements of its target's azimuthal angles. Therefore, these estimates can be further refined with a non-linear optimization framework that takes this uncertainty into account. This framework would converge to the optimal estimate by cumulatively minimizing the inlier targets' range rate and azimuthal errors. Overall, this optimization process is very similar to the Orthogonal Distance Regression deployed by [118] and [81] and the non-linear optimization deployed by [72–74]. However, these works do not explicitly provide the necessary underlying equations. Consequently, the rest of this section defines these equations so they are made accessible to the reader.

The range rate error metric is described in Equation 6.9 for the i^{th} point target:

$$e_{\dot{r},i} = \dot{r}_i - \hat{\dot{r}}_i \tag{6.9}$$

where the $\hat{}$ superscripts denote predicted or estimated values here and for the remainder of this chapter. Thus, $\hat{\dot{r}}_i$ can be calculated via Equation 6.7 with $\hat{v}_x$, $\hat{v}_y$, and θ_i. Furthermore, this non-linear optimization introduces the azimuthal angle error metric[3], as described in Equation 6.10:

$$e_{\theta,i} = \theta_i - \hat{\theta}_i = \theta_i - \left(\hat{\alpha}_\theta \pm \hat{\beta}_i\right) \tag{6.10}$$

where α_θ is the azimuthal angle of the RADAR's translational motion vector. It is mathematically defined in Equation 6.11:

$$\alpha_\theta = \operatorname{atan2}\left(v_y, v_x\right) \tag{6.11}$$

Theoretically, range rates measured in the direction of the RADAR's translational motion vector should be equal and opposite to its vector norm. Thus, the range rate's magnitude should decrease as the target deviates from this direction. Therefore, β_i is defined as the angle between the RADAR's translational motion vector and the vector from the RADAR to the i^{th} target. It is mathematically defined in Equation 6.12:

$$\beta_i = \arccos\left(-\dot{r}_i / \left\|\vec{v}_S\right\|_2\right) \tag{6.12}$$

where $\left\|\vec{v}_S\right\|_2$ denotes the Euclidean norm of the RADAR's translational ego-motion vector. Consequently, the $\pm\hat{\beta}_i$ in Equation 6.10 accounts for the fact that, given the sensor motion, each range rate measurement could have originated from two possible azimuthal angles. Specifically, it is ambiguous if $\hat{\theta}_i$ should reside to the left or right of $\hat{\alpha}_\theta$. This ambiguity can be resolved by comparing both predicted angles to the actual azimuthal angle measurement.

For the framework to understand how it should adjust its next iteration's estimate, the partial derivatives of each measurement's errors with respect to the vehicle's ego-motion must be defined. Without proof, this Jacobian, $\boldsymbol{J}_{\vec{e}_i,\vec{v}_S}$, is defined in Equations 6.13 and 6.14:

$$\boldsymbol{J}_{\vec{e}_i,\vec{v}_S} = \begin{bmatrix} \frac{\partial e_{\dot{r},i}}{\partial \hat{v}_x} & \frac{\partial e_{\dot{r},i}}{\partial \hat{v}_y} \\ \frac{\partial e_{\theta,i}}{\partial \hat{v}_x} & \frac{\partial e_{\theta,i}}{\partial \hat{v}_y} \end{bmatrix} = \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_x} \mp \frac{\partial \hat{\beta}_i}{\partial \hat{v}_x} & -\frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_y} \mp \frac{\partial \hat{\beta}_i}{\partial \hat{v}_y} \end{bmatrix} \tag{6.13}$$

where

$$\begin{bmatrix} \frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_x} & \frac{\partial \hat{\alpha}_\theta}{\partial \hat{v}_y} \\ \frac{\partial \hat{\beta}_i}{\partial \hat{v}_x} & \frac{\partial \hat{\beta}_i}{\partial \hat{v}_y} \end{bmatrix} = \begin{bmatrix} \frac{-\hat{v}_y}{\hat{v}_x^2+\hat{v}_y^2} & \frac{\hat{v}_x}{\hat{v}_x^2+\hat{v}_y^2} \\ \frac{-\hat{v}_x\dot{r}_i}{(\hat{v}_x^2+\hat{v}_y^2)^{3/2}\sqrt{1-\frac{\dot{r}_i^2}{\hat{v}_x^2+\hat{v}_y^2}}} & \frac{-\hat{v}_y\dot{r}_i}{(\hat{v}_x^2+\hat{v}_y^2)^{3/2}\sqrt{1-\frac{\dot{r}_i^2}{\hat{v}_x^2+\hat{v}_y^2}}} \end{bmatrix} \tag{6.14}$$

[3]For practical implementations, the azimuthal error metric, $e_{\theta,i}$ should be scaled by $\operatorname{erf}\left(\sqrt{\hat{v}_x^2+\hat{v}_y^2}\right)$, where $\operatorname{erf}(x)$ is the Gauss error function. This scaling factor smoothly eliminates this error metric's influence only for translational velocities near zero. This is necessary because the expected azimuthal angles are grossly unreliable for these negligible velocities.

However, due to a variety of factors, each Doppler measurement, $\vec{z}_i$, constrains the vehicle ego-motion estimate differently. Therefore, $\mathbf{\Sigma}_{\hat{\vec{v}}_S,\vec{z}_i}$, the translational ego-motion covariance matrix corresponding to each Doppler measurement, must be defined. This can be calculated if the measurement's expected range rate and azimuthal uncertainties ($\sigma_{\dot{r}}$, σ_θ) are provided by the RADAR's datasheet. These measurement uncertainties can then be propagated to the vehicle ego-motion estimate's uncertainties as shown in Equations 6.15 and 6.16:

$$\mathbf{\Sigma}_{\hat{\vec{v}}_S,\vec{z}_i} = \begin{bmatrix} \sigma^2_{\hat{v}_x} & \sigma_{\hat{v}_x\hat{v}_y} \\ \sigma_{\hat{v}_y\hat{v}_x} & \sigma^2_{\hat{v}_y} \end{bmatrix}_{\vec{z}_i} = \boldsymbol{J}_{\hat{\vec{v}}_S,\vec{z}_i} \begin{bmatrix} \sigma^2_{\dot{r}} & 0 \\ 0 & \sigma^2_{\theta} \end{bmatrix} \boldsymbol{J}^T_{\hat{\vec{v}}_S,\vec{z}_i} \tag{6.15}$$

where

$$\boldsymbol{J}_{\hat{\vec{v}}_S,\vec{z}_i} = \begin{bmatrix} \frac{\partial \hat{v}_x}{\partial \dot{r}_i} & \frac{\partial \hat{v}_x}{\partial \theta_i} \\ \frac{\partial \hat{v}_y}{\partial \dot{r}_i} & \frac{\partial \hat{v}_y}{\partial \theta_i} \end{bmatrix} = \begin{bmatrix} -1/\cos\theta_i & \frac{-\dot{r}_i \sin\theta_i - \hat{v}_y}{\cos^2\theta_i} \\ -1/\sin\theta_i & \frac{\dot{r}_i \cos\theta_i + \hat{v}_x}{\sin^2\theta_i} \end{bmatrix} \tag{6.16}$$

For this chapter's implementation, the aforementioned error metrics, Jacobians, and covariances[4] were directly integrated into the g^2o non-linear optimization framework [119].

6.4 Correcting the Vehicle Ego-motion Estimates

While MonoRAD focuses on estimating the *sensor's* ego-motion, the end objective is to correct the *vehicle's* ego-motion. There are several ways to implement this; however, they all depend on the vehicle and its sensors sharing the same rigid body. This is critical because, if the full motion state of one point on the rigid body is known, the full motion state of any other point on that body can be calculated given the lever-arm offset between the two. This relationship is mathematically defined in Equation 6.17:

$$\vec{v}_A = \vec{v}_B + \vec{\Omega} \times \vec{r}_{A/B} \tag{6.17}$$

where $\vec{v}_A$ and $\vec{v}_B$ are the translational velocities of points A and B on the rigid body, respectively. The vector $\vec{r}_{A/B}$ is the lever-arm offset of point A with respect to point B and $\vec{\Omega}$ is the rotational velocity vector of the rigid body, which is shared by all points. For brevity, the coordinate frames of A and B are assumed to be aligned. Note how the lever-arm offset and rotational velocities combine to yield translational velocity differences between points A and B. These translational velocity differences can be observed by the Doppler measurements, permitting their indirect estimates of rotational ego-motion.

Therefore, for ego-motion estimation, these points typically represent the sensor's and vehicle's reference frame. Technically, the vehicle's reference frame can be arbitrarily chosen; however, certain reference frames are more convenient than others. For example, for low-speed vehicles where zero side-slip can be assumed, the vehicle reference frame is typically the rear axle's midpoint. At this location, the lateral velocity can be assumed to be zero in the vehicle's coordinate frame [109], greatly simplifying Equation 6.17. Therefore, if a sensor at another location measures a velocity that projects to the vehicle's lateral axis, it must be only due to the vehicle's yaw rate. These integrated vehicle assumptions were critical for the "Sinefit" method [80] and greatly aids the "2DOF" variants of [72–74, 81].

However, when zero side-slip cannot be assumed, the vehicle's ego-motion state can be ambiguous from a single RADAR's Doppler measurements. Specifically, Equation 6.17 demonstrates that the full motion state of the vehicle cannot be uniquely determined with only the sensor's translational velocity estimates. Fortunately, this ambiguity can be resolved if shared rotational velocities can be estimated from other measurements. Then, the vehicle's ego-motion can be corrected in several ways within the spectrum of loosely- to tightly-coupled sensor fusion frameworks.

[4] More accurately, the covariance matrices' inverses (i.e. information matrices) were utilized.

6.4.1 Loosely-Coupled Framework

In this context, loosely-coupled frameworks correct *a priori* vehicle ego-motion estimates through a direct comparison with *vehicle* ego-motion measurements [4, 109]. Specifically, modifying Equation 6.17, the vehicle's ego-motion can be calculated via Equation 6.18:

$$\vec{v}_V = \vec{v}_S - \vec{\Omega} \times \vec{r}_{S/V} \tag{6.18}$$

where $\vec{v}_V$ and $\vec{v}_S$ represent the vehicle's and sensor's translational velocities, respectively, and $\vec{r}_{S/V}$ is the lever-arm offset of the sensor with respect to the vehicle's reference frame. The sensor and vehicle share their rotational motion, $\vec{\Omega}$.

Note that a unique vehicle motion estimate can be calculated only if the full sensor motion state ($\vec{v}_S$, $\vec{\Omega}$) can be measured. Some sensor modalities can estimate the full motion state, e.g. Stereo VO and methods utilizing multiple RADARs [17, 81]. Others resolve ambiguities by depending on complementary sensor modalities for different state estimates. For example, inertial or odometry measurements can estimate the scale of Monocular VO's translational velocities [129]. Also, IMU measurements or spatial-based RADAR techniques can estimate rotational ego-motion while Doppler measurements can estimate translational ego-motion [82].

6.4.2 Tightly-Coupled Framework

While the loosely-coupled framework's extraction is intuitive, it loses valuable information in the process. Therefore, more tightly-coupled frameworks can be deployed for improved navigation accuracy [82]. Tightly-Coupled frameworks incorporate measurements in a more raw state for processing within the fusion framework itself. Specifically, instead of comparing vehicle ego-motion estimates, they compare actual and expected *sensor* measurements [4, 109]. In this context, such a framework could use the *a priori* vehicle ego-motion estimates to calculate what the sensor measurements are expected to be. For translational velocities, this relationship can be extracted from Equation 6.18 to yield Equation 6.19:

$$\vec{v}_S = \vec{v}_V + \vec{\Omega} \times \vec{r}_{S/V} \tag{6.19}$$

Tightly-coupled frameworks are often deployed when the vehicle motion corresponds to a unique sensor measurement, but not vice versa [4, 109]. This is precisely the case for the Doppler measurements.

The caveat here is that the full *a priori* estimate must come from another source or sensor modality. Often, this is provided by an odometry-based motion model or IMU measurements. Consequently, variants of tightly-coupled frameworks exist for Visual Inertial Odometry, e.g. comparing observed feature locations with those expected from inertial navigation [129]. Furthermore, they are present in the internal processes of joint RADAR ego-motion estimation techniques, e.g. comparing Doppler measurements with those expected from the initial spatial data-based ego-motion estimates [72–74].

It may be best to think of tightly-coupled frameworks as not passing through vehicle motion estimates, but the vehicle motion's probability distributions. Since sensor measurements can be ambiguous with regards to vehicle ego-motion, the *a priori* vehicle ego-motion uncertainties are critical for resolving these ambiguities. For this reason, due to the coupling present in Equation 6.19, tightly-coupled frameworks are actually able to correct *both* rotational and translational vehicle ego-motion estimates from $\vec{v}_S$ measurements [82]. For MonoRAD, these rotational ego-motion corrections can work in conjunction with their image-based counterparts.

It is worth mentioning that tightly-coupled frameworks exist within a spectrum. A framework is said to be more tightly coupled if more of the measurements' processing takes place within the framework itself. Thus, a more tightly coupled framework receives the measurements in a more raw, unprocessed state [4, 109]. This is typically advantageous because it reduces the probability distribution inaccuracies induced by extraction.

6.5 Validation

While many Visual Odometry datasets exist, unfortunately, the author is unaware of any publicly-available datasets with RADAR Doppler data. Therefore, MonoRAD was validated using real data collected from an instrumented semi-truck driving a 805 m suburban route for speeds up to 27 kph. This route, shown in Figure 6.8, was specifically chosen for the frequency and magnitude of its turns. Furthermore, this route was on a public road, so MonoRAD had to detect and reject dynamic agents in the semi-truck's environment.

MonoRAD utilized the data from a front-mounted Delphi ESR 2.5 RADAR [117] and a windshield-mounted FLIR Blackfly monocular camera [130]. For use as a reference, global position measurements were provided by an onboard Hemisphere A325 GNSS receiver [120]. Reference global velocities were provided by its GNSS Doppler measurements. The vehicle's reference rotational velocities were provided by an onboard ADIS 16407 IMU [131].

While the reference sensors were utilized due to their relative accuracy, they themselves have non-negligible uncertainties, so they cannot be considered true "ground truth." Therefore, this validation can only estimate MonoRAD's uncertainties up to the limits set by the reference sensors' uncertainties. Even with these limitations, this practical validation with real-world data and environments was viewed as more valuable than a simulation-based validation with ground truth.

6.5.1 Real-World Results

To promote intuitive results, the aforementioned sensor fusion frameworks were not implemented for this validation. Thus, MonoRAD's presented camera- and RADAR-based estimates are independent from any other sensors.

Several factors prevent a direct comparison of MonoRAD's translational estimates with the reference GNSS Doppler measurements. Due to the side-slip present at the GNSS receiver's reference frame, the GNSS receiver experiences a lateral velocity while the vehicle is turning. Furthermore, while MonoRAD is able to estimate its ego-motion about the vehicle's longitudinal axis, GNSS Doppler measurements are not vehicle-aligned. Without an accurate heading measurement, it can be practical to simply calculate the GNSS velocity vector's Euclidean norm. However, this incorporates any "side-slip" velocities that align with the vehicle's lateral axis, yielding an inflated value during turns.

Therefore, to permit a direct comparison, the GNSS receiver's lateral velocities were estimated using Equation 6.17. Specifically, this calculation utilized the IMU's reported yaw rates and the GNSS receiver's lever-arm offset from the rear axles' midpoint, i.e. the "zero side-slip" reference frame that could be assumed to experience no lateral motion for this route. Then, the lateral velocity component was removed from the Euclidean norm to produce a pseudo-GNSS longitudinal velocity estimate. Thus, more accurately, MonoRAD's longitudinal velocity estimates were compared to reference GNSS+IMU Doppler measurements.

These estimates were plotted in Figure 6.4 for the course of the route. As the figure demonstrates, MonoRAD's estimates closely mirror the reference GNSS Doppler measurements and even appear to be corrupted by significantly less noise. MonoRAD's longitudinal velocity estimate errors with respect to the GNSS Doppler measurements were calculated to be 0.00 ± 0.06 m/s (0.00 ± 0.22 kph) and are visualized in Figure 6.5. However, this uncertainty is a combination of MonoRAD measurements' and the reference GNSS+IMU measurements' uncertainties. For comparison, the GNSS+IMU measurement uncertainty was calculated to be ± 0.057 m/s for a period when the vehicle was stationary right before this route traversal. Thus, removing this influence, MonoRAD's *expected* longitudinal velocity estimate errors were calculated to be 0.00 ± 0.02 m/s (0.00 ± 0.07 kph). This is consistent with the results of Kellner et al.'s similar method for multiple RADARs [81].

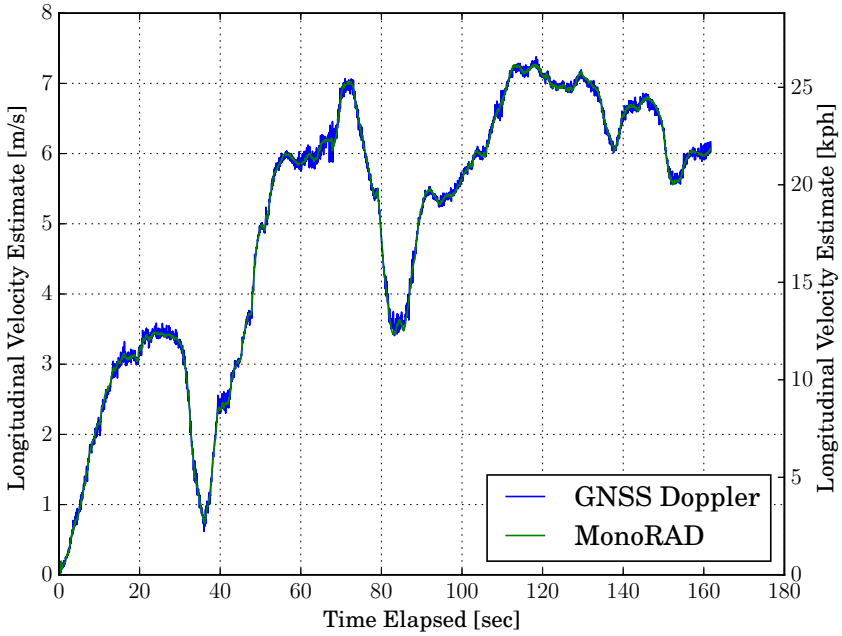


Figure 6.4: Longitudinal velocity estimates from GNSS Doppler and MonoRAD measurements for the course of the route.

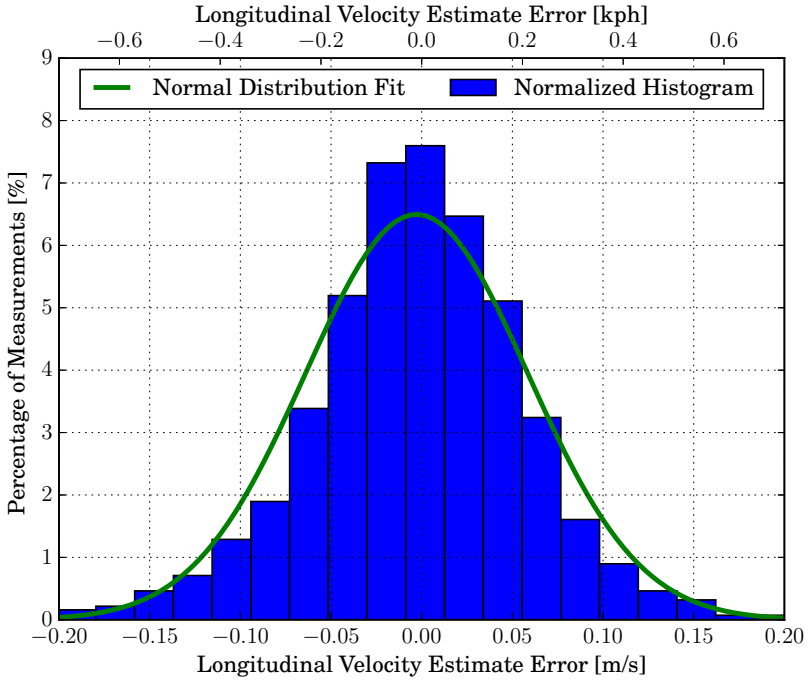


Figure 6.5: MonoRAD's longitudinal velocity estimate errors with respect to GNSS Doppler measurements for the entire route.

Fortunately, since a rigid body shares its rotational motion, MonoRAD's yaw rates estimates can technically be directly compared to the IMU's. However, IMUs are notorious for their walking biases. For comparison, the IMU errors were calculated to be -0.08 ± 0.44 *deg./s* for a period when the vehicle was stationary right before this route traversal. This bias was then removed from the reference IMU's measurements. Therefore, the (calibrated) IMU and MonoRAD yaw rate estimates for the course of the route were plotted in Figure 6.6. As the figure demonstrates, MonoRAD's estimates closely mirror the IMU's. MonoRAD's yaw rate estimate errors with respect to the (calibrated) IMU measurements were calculated to be 0.09 ± 0.79 *deg./s* and are visualized in Figure 6.7. This uncertainty is a combination of both MonoRAD's and the IMU's uncertainties. Therefore, removing the IMU's uncertainty influence, MonoRAD's *expected* yaw rate error is 0.09 ± 0.66 *deg./s*. This validation was unable to determine whether this slight bias was due to MonoRAD's implementation or from other factors.

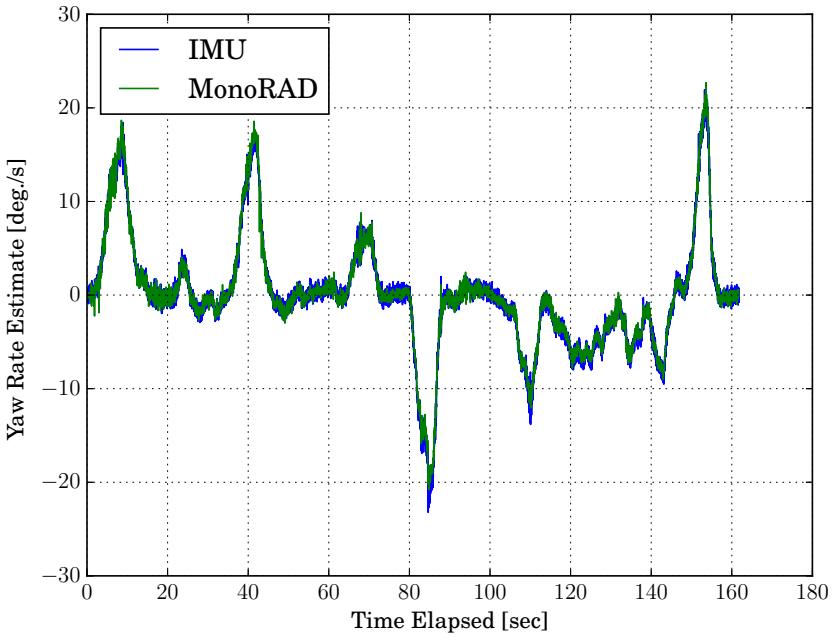


Figure 6.6: Yaw rate estimates from calibrated IMU and MonoRAD measurements for the course of the route.

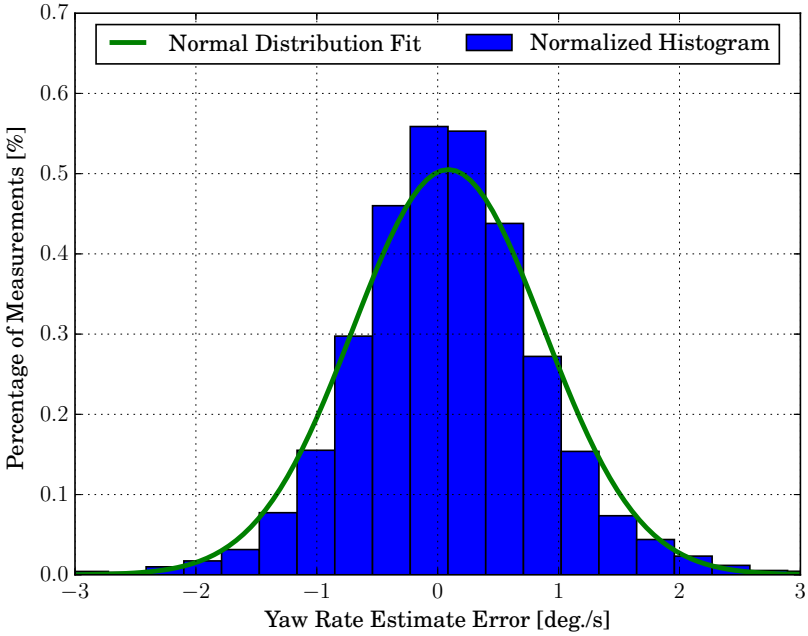


Figure 6.7: MonoRAD's yaw rate estimate errors with respect to calibrated IMU measurements for the course of the route.

Since MonoRAD provides the complete planar ego-motion estimates, they alone can be utilized for dead-reckoning navigation. To illustrate this, MonoRAD's dead reckoning navigation estimate was plotted in Figure 6.8 alongside the reference GNSS position measurement for the route. Due to the accurate ego-motion estimates shown in Figures 6.4 and 6.6, MonoRAD's dead reckoning estimate closely mirrors the reference GNSS measurement.

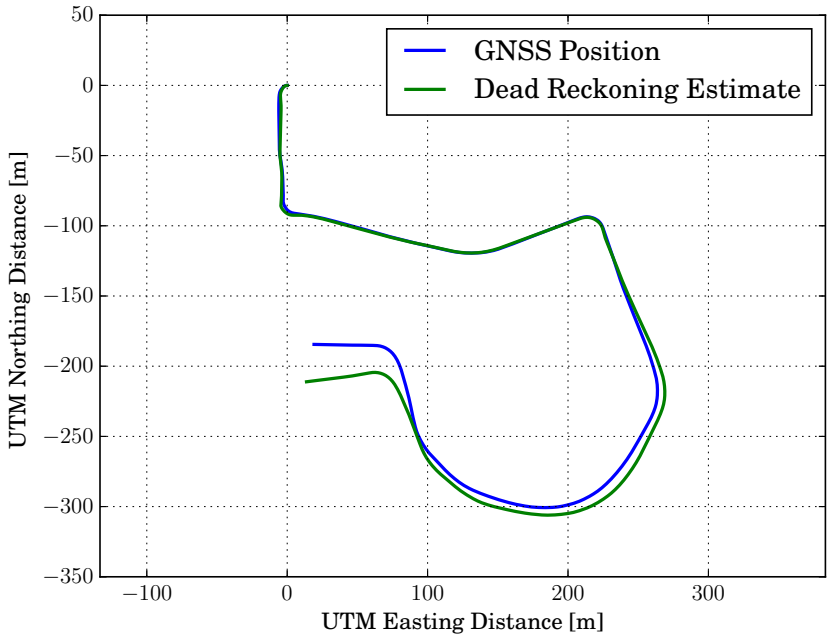


Figure 6.8: MonoRAD's dead reckoning navigation estimates and the GNSS position measurements for the chosen route.

As expected, MonoRAD's estimates exhibit "odometry drift" from the reference measurements over time. For the 805 m route, the final error was 27 m, yielding a percent error of 3.37%. However, it is a known limitation that dead reckoning percent error is very sensitive to the length of the trajectory and how early rotational errors occurred within that trajectory. Therefore, the popular KITTI Odometry Benchmark Evaluation offered a standard for how translational errors can be reported. For KITTI evaluations, routes are split into all possible sub-sequences of [100 m, 200 m, 300 m, ..., 800 m] path lengths. Then, the odometry estimate's dead reckoning percent error is calculated for each possible sub-sequence. The mean of all percent errors thus yields the reported value [132]. Applying this methodology for this route, MonoRAD's mean dead reckoning percent error was 2.78%. To illustrate how this value changes with respect to sub-sequence length, the mean translation percent error for each possible sub-sequence length was plotted in Figure 6.9.

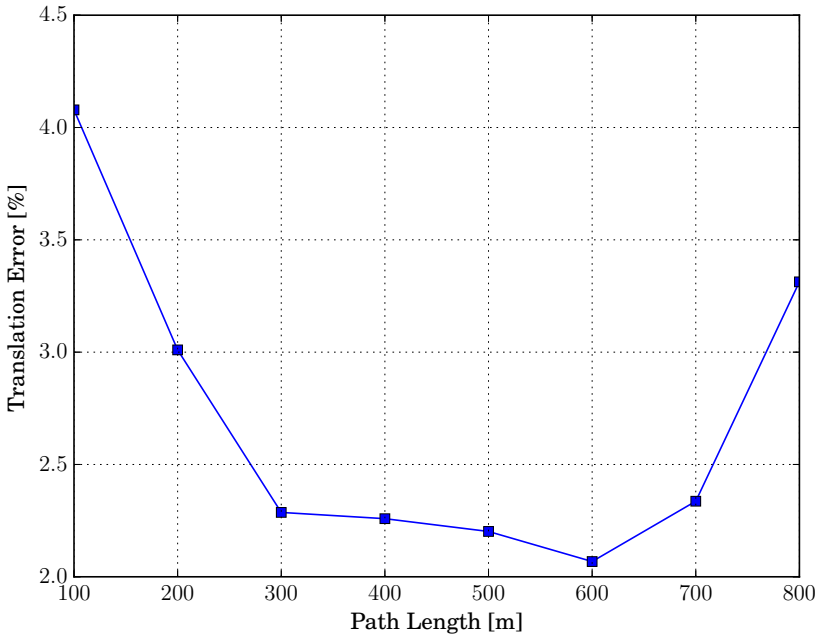


Figure 6.9: MonoRAD's mean dead reckoning translation percent error for each possible route path length sub-sequence.

Collectively, these validation results demonstrate that MonoRAD's estimates accumulate only small translation errors during short-term dead reckoning navigation.

6.5.2 Computational Performance

For this validation, MonoRAD was executed as a Robotic Operating System (ROS) [121] C++ node running on an Intel Core i7-7500U 64-bit CPU (2.70 GHz). In this environment, MonoRAD's mean runtime for the aforementioned route was approximately 36 ms. Since this corresponds to a measurement frequency of almost 28 Hz, MonoRAD can be considered real-time for most automotive scenarios. It is noteworthy that MonoRAD's RADAR-based subprocess had a mean runtime of only 0.3 ms, less than 1% of MonoRAD's entire mean runtime.

6.6 Extension For More Tightly-Coupled Frameworks

Section 6.4 briefly discussed both loosely-coupled and tightly-coupled sensor fusion frameworks in the context of MonoRAD's estimates. As previously discussed, tightly-coupled frameworks often offer accuracy advantages over their loosely-coupled counterparts. Furthermore, a tightly-coupled framework can utilize the sensor's translational ego-motion estimates to estimate the vehicle's rotational *and* translational ego-motion [82]. Consequently, MonoRAD's estimates are arguably better suited for a tightly-coupled framework.

In general, tightly-coupled frameworks are characterized by the fact that they process measurements within the framework itself. Therefore, tightly-coupled frameworks exist within a spectrum that is defined by how much processing is handled by the framework. Naturally, it follows that, if tightly-coupled frameworks offer accuracy advantages over loosely-coupled frameworks, even more tightly-coupled frameworks offer further improvements. Therefore, there are benefits to integrating MonoRAD's methodology into a framework more tightly coupled than the one presented in Section 6.4.2. This section discusses that extension.

The caveat, however, is that frameworks that are more tightly coupled are often less intuitive. By definition, it is often infeasible to decouple their sub-systems and/or sensor estimates. Therefore, such a framework deserves a comprehensive nuanced discussion that, unfortunately, is outside the scope of this chapter. Despite this, this section presents their fundamental equations so that they are made accessible to the reader.

In the context of MonoRAD, an even more tightly coupled framework would incorporate the translational ego-motion estimation process (detailed in Section 6.3.2) so that it resides within the framework itself. In other words, such a framework would take the raw range rate measurements as inputs in lieu of the processed RADAR translational ego-motion estimates.

Consequently, for the framework to correct the *vehicle's* ego-motion, the relationship between the range rate measurements and the vehicle's ego-motion must be defined. This relationship is best defined by generalizing the equations in Section 6.3.2 for all three dimensions and combining them with the equations in Section 6.4.

First, the i^{th} static target's translational motion with respect to the sensor's (RADAR's) frame, $\vec{v}_{T/S,i}$, must be defined. This is defined by modifying Equation 6.17 to instead describe their relative motion, as shown in Equation 6.20:

$$\begin{aligned} \vec{v}_{T/S,i} &= -\left(\vec{v}_S + \vec{\Omega} \times \vec{r}_{T/S,i}\right) \\ &= -\vec{v}_S - \vec{\Omega} \times \vec{r}_{T/S,i} \end{aligned} \tag{6.20}$$

where $\vec{r}_{T/S,i}$ is the vector describing the i_{th} target's position in RADAR's Cartesian coordinate system. That target's range rate is then the scalar projection of $\vec{v}_{T/S,i}$ onto that target's unique radial axis. This radial axis is described by the unit vector $\tilde{r}_{T/S,i}$, which is defined in Equation 6.21:

$$\tilde{r}_{T/S,i} = \vec{r}_{T/S,i}/r_i = \begin{bmatrix} \cos\varphi_i \cos\theta_i & \cos\varphi_i \sin\theta_i & \sin\varphi_i \end{bmatrix} \tag{6.21}$$

Note that $\tilde{r}_{T/S,i}$ can be calculated by normalizing $\vec{r}_{T/S,i}$ by the target's RADAR range, r_i. However, this Cartesian vector can also be calculated with just the target's azimuthal angle, θ_i, and elevation angle, φ_i, in the RADAR's coordinate system. In this context, a target's elevation angle refers to its angle from the RADAR's xy plane.[5] This coordinate system convention is visualized in Figure 6.10.

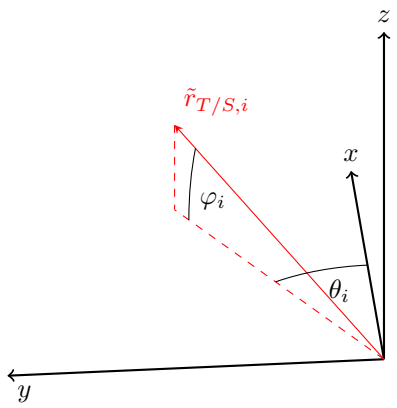


Figure 6.10: Coordinate system convention used to describe the i^{th} RADAR target's radial axis unit vector, $\tilde{r}_{T/S,i}$.

Consequently, the target's range rate can be calculated using Equation 6.20 and the properties of vector algebra, as shown in Equation 6.22:

[5]This is referred to as the depression angle if the $+z$ axis points downward.

$$
\begin{aligned}
\dot{r}_i &= \vec{v}_{T/S,i} \cdot \tilde{r}_{T/S,i} \\
&= \left(-\vec{v}_S - \vec{\Omega} \times \vec{r}_{T/S,i} \right) \cdot \tilde{r}_{T/S,i} \\
&= -\vec{v}_S \cdot \tilde{r}_{T/S,i} - \left(\vec{\Omega} \times \vec{r}_{T/S,i} \right) \cdot \tilde{r}_{T/S,i} \\
&= -\vec{v}_S \cdot \tilde{r}_{T/S,i} - \vec{\Omega} \cdot \Big(\underbrace{\vec{r}_{T/S,i} \times \tilde{r}_{T/S,i}}_{=\vec{0}} \Big) \\
&= -\vec{v}_S \cdot \tilde{r}_{T/S,i}
\end{aligned}
\tag{6.22}
$$

Mirroring the derivation of Equation 6.7, rotational ego-motion was eliminated from Equation 6.22. For this reason, Doppler measurements cannot estimate the sensor's rotational ego-motion *in isolation*. Therefore, Equations 6.19 and 6.22 can be combined to relate range rates to the vehicle's ego-motion, as shown in Equation 6.23:

$$
\begin{aligned}
\dot{r}_i &= -\Big(\vec{v}_V + \vec{\Omega} \times \vec{r}_{S/V} \Big) \cdot \tilde{r}_{T/S,i} \\
&= -\vec{v}_V \cdot \tilde{r}_{T/S,i} - \left(\vec{\Omega} \times \vec{r}_{S/V} \right) \cdot \tilde{r}_{T/S,i} \\
&= -\vec{v}_V \cdot \tilde{r}_{T/S,i} - \vec{\Omega} \cdot \left(\vec{r}_{S/V} \times \tilde{r}_{T/S,i} \right)
\end{aligned}
\tag{6.23}
$$

This equation exposes several important phenomena. By relating range rates to the *vehicle's* ego-motion, range rates can now help estimate rotational ego-motion. However, this is only true when $\vec{r}_{S/V} \neq \vec{0}$ and $\vec{r}_{S/V} \times \tilde{r}_{T/S,i} \neq \vec{0}$. Consequently, it is imperative that the RADAR is at a lever-arm offset from the vehicle's reference frame. Furthermore, targets whose radial axes are nearly aligned with that lever-arm provide negligible information about rotational motion. Thus, front-mounted RADARs must have a wide horizontal field-of-view for their Doppler measurements to aid rotational ego-motion estimation.

Equation 6.23 is critical for this more tightly-coupled framework. It can be used (with $\hat{\vec{v}}_V$ and $\hat{\vec{\Omega}}$) to calculate $\hat{\dot{r}}_i$, the target's expected range rate measurement. This estimate is required for the range rate error metric, as defined in Equation 6.9. Similarly, this framework requires the azimuthal error metric, as defined in Equation 6.10. Furthermore, a three-dimensional implementation additionally requires an error metric defined by the targets' elevation angles.[6] This is mathematically defined in Equation 6.24:

$$
e_{\varphi,i} = \varphi_i - \hat{\varphi}_i = \varphi_i - \left(\hat{\alpha}_\varphi \pm \hat{\beta}_i \right) \tag{6.24}
$$

where α_φ represents the elevation angle of the RADAR's translational motion vector, as defined by Equation 6.25:

$$
\alpha_\varphi = \arcsin\left(v_z / \left\| \vec{v}_S \right\|_2 \right) \tag{6.25}
$$

where, for this three-dimensional analysis, $\vec{v}_S = [v_x,\, v_y,\, v_z]^T$. Lastly, the Jacobians and covariance matrices in Section 6.3.2 need to be rederived for this framework's more tightly coupled implementation.

However, it is important to keep in mind that MonoRAD's estimates are limited by the sensors utilized. Since automotive RADARs typically cannot measure their targets' elevation angles, their Doppler measurements cannot be used to estimate the RADAR's vertical ego-motion, v_z. Thus, in these scenarios, MonoRAD may only be able to estimate five of the six ego-motion states. This may cause implementation issues for a six degree-of-freedom loosely-coupled framework. Conversely, any of the previously discussed tightly-coupled frameworks are well suited for this limitation.

Regardless, vehicle motion constraints and the RADAR's narrow vertical field-of-view often make a planar analysis sufficient for translational ego-motion. For this reason, Section 6.3.2 focused on a planar analysis. Furthermore, if a planar analysis can be assumed, $v_z = 0$ and $\varphi_i = 0$, simplifying Equation 6.22 to Equation 6.7. Similarly, Equation 6.23 can be simplified significantly for planar implementations.

[6] For similar reasons as the azimuthal error metric, it is advisable to scale the elevation angle error metric, $e_{\varphi,i}$, by $\operatorname{erf}(\hat{v}_z)$.

6.7 MonoRAD Conclusion

This chapter presented MonoRAD, a novel ego-motion estimation algorithm that integrated two popular and complementary sensors: a monocular camera and a single RADAR. The presented real-world validation results demonstrated that MonoRAD offers fast and accurate ego-motion estimates. Specifically, they demonstrated that MonoRAD estimated longitudinal velocities with an expected standard deviation of 0.02 m/s (0.07 kph), estimated yaw rates with an expected standard deviation of 0.66 $deg./s$, and yielded a mean dead reckoning percent error of 2.78% at a mean runtime of 36 ms.

The presented implementation used submodules from widely accepted camera- and RADAR-based techniques. However, the essence of MonoRAD's novel concept is not in its implementation details, but in its decoupled integration of camera images and RADAR Doppler measurements for rotational and translational ego-motion estimates, respectively. Therefore, these submodules could easily be refined with more state-of-the-art techniques while adhering to MonoRAD's core concept. In all, this chapter demonstrates that the integration of these complementary sensors is highly promising as an ego-motion estimation solution and for future research.

Chapter 7

SONARODO: Motion Estimation from Doppler and Spatial Data in SONAR Images

Abstract – Motion estimation is critical for the localization of Autonomous Underwater Vehicles. Current SONAR-based techniques exclusively utilize either Doppler or spatial measurements. However, these measurement domains are complementary to each other; Doppler measurements directly measure radial motion while spatial measurements uniquely observe angular motion. Therefore, this chapter presents SONARODO (SONAR Odometry), a novel real-time motion estimation algorithm for 2D Forward-Looking SONARs. It depends on a largely decoupled motion estimation process that better utilizes each measurement domain for their respective strengths. Specifically, it estimates translational motion from Doppler-Azimuth images and rotational motion from Range-Azimuth images. This method was designed to ensure robustness to relatively featureless seafloor environments and low-resolution SONAR images. The chapter's validation with high-fidelity simulation data demonstrated that SONARODO offers accuracy and computational cost advantages over related motion estimation techniques.

7.1 Introduction to SONARODO

Autonomous Underwater Vehicle (AUV) localization poses a particularly challenging problem. Global position measurements are often unavailable due to signal attenuation underwater or the lack of equivalent long-distance acoustic networks. For attitude, magnetic heading sensor inaccuracies have been shown to be a principal source of navigation error [5, 6]. While matching with an *a priori* map is an option, the necessary high-resolution data is rarely available [9].

Therefore, many AUVs utilize local techniques based on dead-reckoning navigation instead. For instance, methods that estimate motion relative to the water column can support dead-reckoning navigation. However, these navigation estimates are easily corrupted by unknown currents [8, 9]. Due to these environmental factors, the inertial measurement unit (IMU) is often the foundation of an AUV's localization framework [9, 18]. Unfortunately, IMUs are notorious for walking biases that quickly corrupt their navigation estimate over time [10].

To improve navigation accuracy, robust AUVs also depend on perception-based motion estimation. Perception-based motion estimation analyzes the perceived motion of the environment from an onboard sensor to estimate the vehicle's own motion within that environment. Since perception-based motion estimation has different error sources than the aforementioned sensors, it can serve as a complementary measurement,

At the time of this writing, this chapter is under review for publication in the IEEE Journal of Oceanic Engineering.

thus constraining odometry drift [3]. For AUVs, the required environmental perception is largely provided by acoustic sensors, i.e. SONAR, due to its relatively low signal attenuation underwater.

A specialized SONAR, the Doppler Velocity Log (DVL), is ubiquitous among AUVs. The DVL is characterized by its narrow beamwidths and its use of Doppler shift to measure relative motion [8, 79]. While these Doppler measurements can be very accurate [18], they are only able to measure relative *radial* motion, rendering angular motion unobservable. Thus, a DVL with "bottom-lock" can measure its translational motion but not its rotational motion. For this reason, the DVL is often placed at the AUV's Center of Buoyancy (CB) to directly measure the vehicle's translational motion.

While there are DVL-based techniques to correct the vehicle's rotational motion, they require that the DVL is mounted at a lever-arm offset from the vehicle's CB.[2] Specifically, its relative observability is a function of this lever-arm's magnitude [82]. Consequently, the limited size of many AUVs often make practical offsets infeasible.

Without a practical DVL offset, rotational motion estimation must depend on *spatial* measurements. These often come in the form of acoustic images (i.e. SONAR images) from a 2D Forward-Looking SONAR (FLS). Unlike Doppler measurements, SONAR images also capture rotational motion. Thus, many motion estimation techniques exist for high-resolution SONARs based on extracting prominent features or edge points [52, 68–71, 134]. However, these techniques are ill-suited for featureless environments and/or SONAR images with a poor resolution and/or signal-to-noise ratio. Fortunately, Fourier-based techniques are better suited for these scenarios [53, 77]. While these techniques have been successful, their translational motion estimates depend on high-resolution range measurements instead of characteristically accurate Doppler measurements.

Consequently, this chapter presents SONARODO (SONAR Odometry), a novel real-time motion estimation algorithm for 2D Forward-Looking SONARs. This algorithm depends on a largely decoupled motion estimation process that better utilizes each measurement domain for their respective strengths. Specifically, it estimates translational motion from Doppler-Azimuth images and rotational motion from Range-Azimuth images. This method was designed to ensure robustness to featureless seafloor environments and low-resolution SONAR images. Furthermore, the presented validation results demonstrate that it offers a fast and accurate motion estimation solution.

7.2 Related Work

SONARODO is related to several current techniques. For translational motion estimation, it analyzes Doppler-Azimuth images. Thus, its analysis of range rates over the entire azimuthal field-of-view most closely resembles Kellner et al.'s RADAR-based *Sinefit* algorithm [80]. However, *Sinefit* does not directly analyze Doppler images. Interestingly, SONAR Doppler images are usually used for environmental perception, but not for estimating its own vehicle's states. This is most likely due to the success of commercial sensors specialized for translational motion estimation, e.g. Doppler Velocity Logs and Correlation Velocity Logs. While these sensors often offer practical solutions, our method enables both environmental perception and motion estimation from a single SONAR.

For rotational motion estimation, SONARODO's sub-process is closely related to Hurtós et al.'s comprehensive body of research and algorithms [53, 75–78]. Like our method, Hurtós et al.'s motion estimation algorithms use Fourier-based registration for 2D FLS images. Our method is particularly similar to their most recent variant, as this method directly analyzes polar images [76–78]. While Hurtós et al.'s research focused on image mosaicing, our method is more focused on motion estimation. Therefore, SONARODO's methodology differs in two important ways. First, it utilizes Doppler data for translational motion estimation. Secondly, for this reason, it estimates motion states in a reversed order.

Essentially, Hurtós et al.'s algorithm deploys a "rotation then translation" estimation methodology. First, it uses Fourier-based registration on consecutive polar images to estimate their relative rotation. Since

[2] This is also the case for related sensors that only provide translational motion estimates, e.g. Correlation Velocity Logs (CVLs) [8, 133].

these images also capture translational motion, it is assumed that these effects are minimal. However, due to this assumption, the authors state that, "its performance can decrease when the image rotation is combined with large translations and therefore other techniques may be considered in this situation" [77]. Then, Hurtós et al. remap one image to compensate for the estimated rotation so that only translational differences remain. Finally, Fourier-based registration is again utilized to estimate translational motion from the Cartesian SONAR images [76, 78]. Since these estimates are based on Cartesian offsets, the authors state that, "Resolution cannot be sacrificed past a certain point if we want to guarantee a good translation estimation" [77].

Conversely, SONARODO deploys a "translation then rotation" methodology. First, it takes advantage of Doppler measurements by directly analyzing Doppler-Azimuth images to estimate translational motion. This order is important because, as will be demonstrated in Section 7.3.1, Doppler measurements are analytically decoupled from rotational motion. Then, the previous polar image is remapped according to the estimated translation so that only rotational differences remain. Finally, Fourier-based registration is employed on these polar images to estimate rotational motion.

7.3 Algorithm Overview

This section details the SONARODO algorithm process, as visualized in Figure 7.1. It mainly consists of two largely decoupled sub-processes: translational and rotational motion estimation, respectively. Its open-source code can be found at `https://bitbucket.org/chrisdmonaco/sonarodo`.

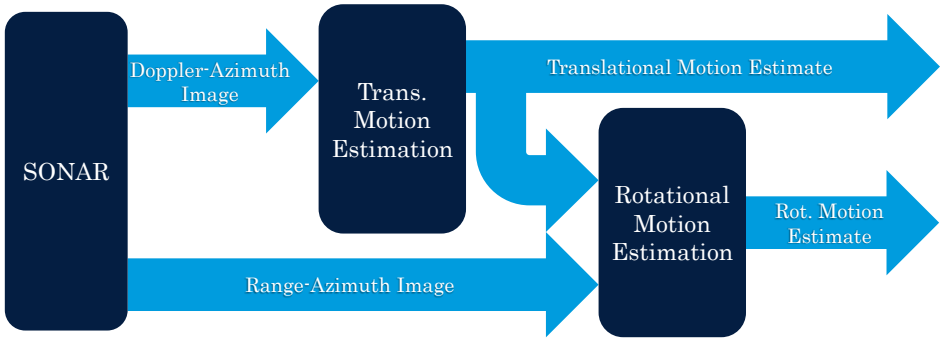


Figure 7.1: SONARODO Algorithm Overview

7.3.1 Translational Motion Estimation

SONARODO's translational motion estimation sub-process depends on its SONAR's Doppler measurements. These measurements capture the relative radial motion of its targets, thus, if the target belongs to the static environment (e.g. the seafloor), its relative motion can be described by Equation 7.1:

$$\begin{aligned} \vec{v}_{T/S,i} &= -(\vec{v}_S + \vec{\Omega} \times \vec{r}_i) \\ &= -\vec{v}_S - \vec{\Omega} \times \vec{r}_i \end{aligned} \tag{7.1}$$

where $\vec{v}_{T/S,i}$ is the vector describing the i^{th} target's translational velocity with respect to the SONAR, $\vec{v}_S$ is the SONAR's translational motion vector (i.e. $[v_x \ v_y \ v_z]^T$), $\vec{\Omega}$ is the vector describing the SONAR's rotational motion (i.e. $[\omega_x \ \omega_y \ \omega_z]^T$), and $\vec{r}_i$ is the vector describing the i^{th} target's position in the SONAR's Cartesian coordinate system. The measured range rates are then the scalar projection of $\vec{v}_{T/S,i}$ onto that target's radial axis. This axis is analytically described in Equation 7.2:

$$\tilde{r}_i = \vec{r}_i / r_i = \begin{bmatrix} \cos\theta_i \cos\phi_i & \sin\theta_i \cos\phi_i & \sin\phi_i \end{bmatrix} \tag{7.2}$$

where $\tilde{r}_i$ is the unit vector that describes the i^{th} target's radial axis in the SONAR's coordinate system. It can be derived via normalizing the target's $\vec{r}_i$ by its SONAR range, r_i. More practically, it can be derived by the target's azimuthal angle, θ_i, and depression angle, ϕ_i, in the SONAR's spherical coordinate system. Note that, in this context, the azimuthal angle refers to the target's angle from the x-axis in the SONAR's xy-plane. Furthermore, the depression angle refers to the target's angle from the SONAR's xy-plane. This coordinate system convention is visualized in Figure 7.2.

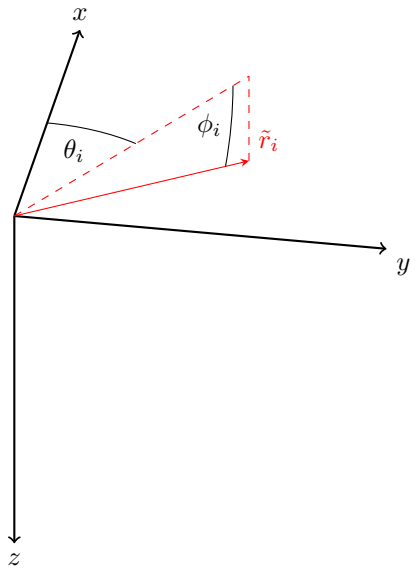


Figure 7.2: Coordinate system convention used to describe the i^{th} SONAR target's radial axis unit vector, $\tilde{r}_i$.

Therefore, combining Equations 7.1 and 7.2 yields the relationship in Equation 7.3 between a target's range rate and the SONAR's motion:

$$\begin{aligned}
\dot{r}_i &= \vec{v}_{T/S,i} \cdot \tilde{r}_i \\
&= \left(-\vec{v}_S - \vec{\Omega} \times \vec{r}_i \right) \cdot \tilde{r}_i \\
&= -\vec{v}_S \cdot \tilde{r}_i - \left(\vec{\Omega} \times \vec{r}_i \right) \cdot \tilde{r}_i \\
&= -\vec{v}_S \cdot \tilde{r}_i - \vec{\Omega} \cdot \left(\underbrace{\vec{r}_i \times \tilde{r}_i}_{=\vec{0}} \right) \\
&= -\vec{v}_S \cdot \tilde{r}_i
\end{aligned} \tag{7.3}$$

Equation 7.3 exposes an important phenomenon; all terms related to the SONAR's rotational motion have cancelled each other out. This analytically demonstrates that Doppler measurements are decoupled from the sensor's rotational motion. In fact, this is a foundational rationale behind the algorithm's decoupled estimation methodology.

By design, DVL transducers have a very narrow beamwidth. Thus, it is valid to assume that their returns were reflected from a point along its transducer's main response axis (MRA). Therefore, with $\tilde{r}_i$ well defined, it is relatively straightforward to apply their range rate measurements to estimate $\vec{v}_S$. Specifically, if all of the DVL's MRAs intersect at the DVL's reference frame, Equations 7.2 and 7.3 can be combined to form the linear system in Equation 7.4:

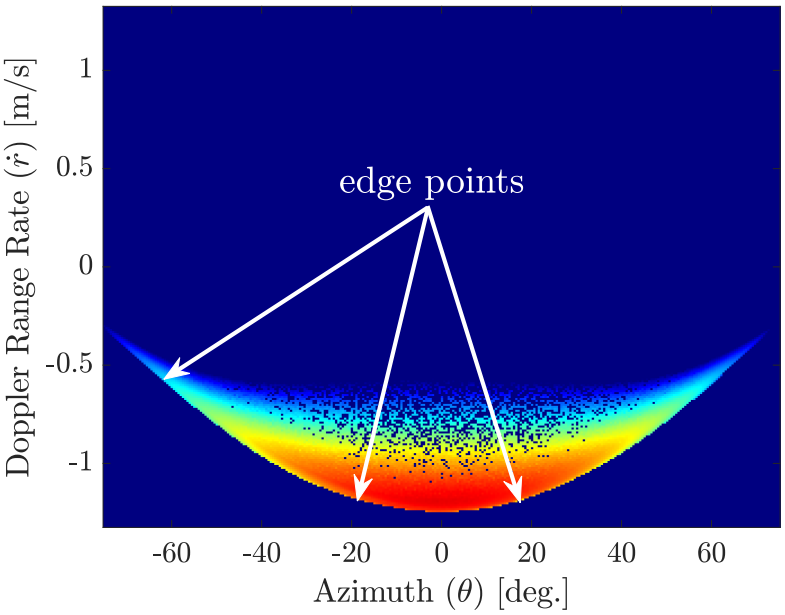
$$\begin{bmatrix} -\cos\theta_1\cos\phi_1 & -\sin\theta_1\cos\phi_1 & -\sin\phi_1 \\ -\cos\theta_2\cos\phi_2 & -\sin\theta_2\cos\phi_2 & -\sin\phi_2 \\ \vdots & \vdots & \vdots \\ -\cos\theta_N\cos\phi_N & -\sin\theta_N\cos\phi_N & -\sin\phi_N \end{bmatrix} \underbrace{\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}}_{\vec{v}_S} = \begin{bmatrix} \dot{r}_1 \\ \dot{r}_2 \\ \vdots \\ \dot{r}_N \end{bmatrix} \tag{7.4}$$

where N refers to the number of targets (and, for DVLs, corresponding transducers).

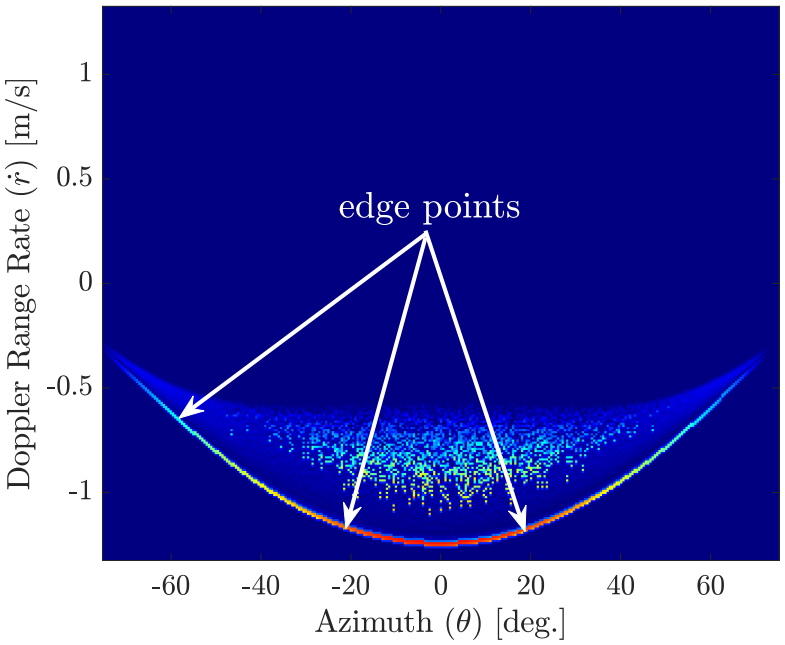
Conversely, Forward Looking SONARs have a much larger beamwidth to capture a large field-of-view for perception. Many 2D SONARs have a horizontal array of transducers to determine their targets' azimuthal angles. Unfortunately, however, their targets' depression angles remain ambiguous. Furthermore, a closer inspection of Equation 7.4 reveals that targets within the same azimuth "slice" likely have varying depression angles, leading to a distribution of range rates. Thus, these Doppler measurements can be provided as Doppler-Azimuth images, as shown in Figure 7.3a. Each image column represents a different azimuth "slice" with bins for all range rate measurements. They are only differentiated by their signals' return intensities, represented in the image as pixel intensities.

At first glance, the high signal intensity region in Figure 7.3a appears to correspond to the main response axis, where $\phi = 0$. Unfortunately, this assumption is quickly invalidated by environmental factors, particularly signal attenuation. In practice, the MRA's range rates are indistinguishable within this continuous distribution. Consequently, with the correspondences between range rates and depression angles remaining unknown, motion initially appears unobservable.

However, Equation 7.3 indicates that this distribution does possess a distinct boundary that can help establish correspondences. Specifically, it states that $|\dot{r}_i| \leq \|\vec{v}_S\|_2$, where $\|\vec{v}_S\|_2$ represents the Euclidean norm of the SONAR's translational velocity vector. If the SONAR's beamwidth encompasses its translational velocity vector, there is a corresponding distinct drop-off at this boundary limit. Similarly, if the SONAR's translational velocity vector resides outside of its vertical beamwidth, this edge corresponds to the range rates at that beamwidth's nearest edge. Therefore, the distinctness of these edge points could be utilized to estimate the SONAR's translational motion.



(a) Doppler-Azimuth image with frequency shifts converted to range rates. Image was generated by the simulation environment described in Section 7.4.



(b) Vertical smoothed derivative magnitudes of the Doppler-Azimuth image.

Figure 7.3: Simulated Doppler-Azimuth image (top) and its vertical smoothed derivative (bottom). Blue regions correspond to low magnitudes; red regions correspond to high magnitudes. Arrows point to some Doppler-Azimuth image "edge points." Image was generated by the simulation environment described in Section 7.4.

Extracting Doppler-Azimuth Image Edge Points

As seen in Figure 7.3a, the Doppler-Azimuth image edge corresponds to a large signal intensity change. Thus, to extract these edge points, the algorithm convolves this image with a vertical derivative kernel. Specifically, this kernel contains the derivative of a Gaussian bell curve. This simultaneously calculates the image derivative while smoothing the result to mitigate the effects of SONAR image "speckle" and data gaps. As visualized in Figure 7.3b, Figure 7.3a's edge is now prominent.

Therefore, an "edge point" is extracted for each azimuth by finding the maximum intensity point within each image column. These edge points correspond to individual range rate and azimuthal angle measurements. However, not all measurements should wield equal influence. Edge points that correspond to low Doppler-Azimuth signal intensities should have a minimal effect on the resulting motion estimate. Thus, the algorithm also extracts their pixel intensities to serve as their weights, i.e. $w_i = I(\dot{r}_i, \theta_i)$.

Compensating for the Depression Angle

Due to their inclusion in Equation 7.4, depression angles must also be known to estimate the motion vector. This is relatively straightforward if depression angles are directly measured. However, for the reference 2D SONAR and its lack thereof, estimation is more limited, complex, and nuanced.

Essentially, the depression angles must then be estimated from other measurements. Specifically, if the SONAR's translational velocity vector, $\vec{v}_S$, resides outside of its vertical beamwidth, the Doppler-Azimuth edge points correspond to that beamwidth's nearest edge. Fortunately, there is a correspondence between the beamwidth edges in the Doppler-Azimuth image and the Range-Azimuth image. An example of a Range-Azimuth image is shown in Figure 7.5. Since SONAR is an active sensor, Range-Azimuth bins with reasonable signal intensities can be assumed to reside within the SONAR's beamwidth. Thus, the algorithm analyzes the Range-Azimuth image to extract the beamwidth's minimum and maximum captured ranges.[3]

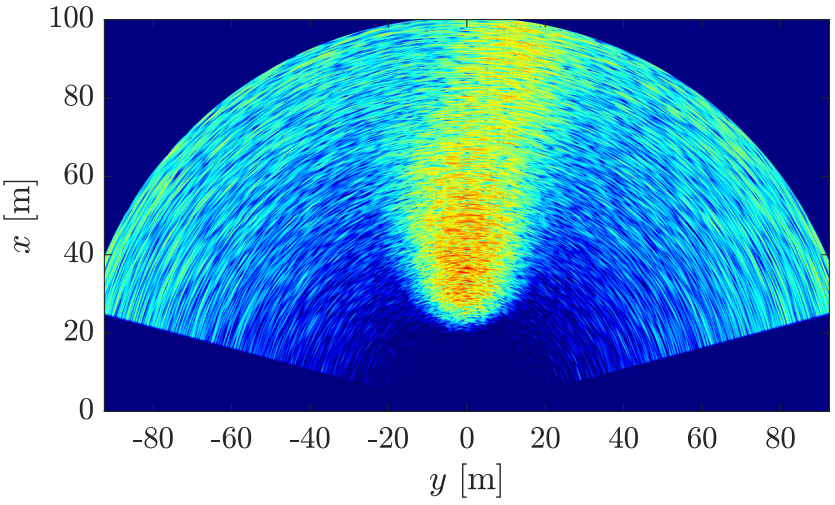


Figure 7.4: Simulated Cartesian SONAR image for a typical seafloor environment. Blue regions correspond to low signal intensities; red regions correspond to high signal intensities. Image was generated by the simulation environment described in Section 7.4.

[3]These ranges can typically be extracted by analyzing the image column that corresponds to the zero azimuth.

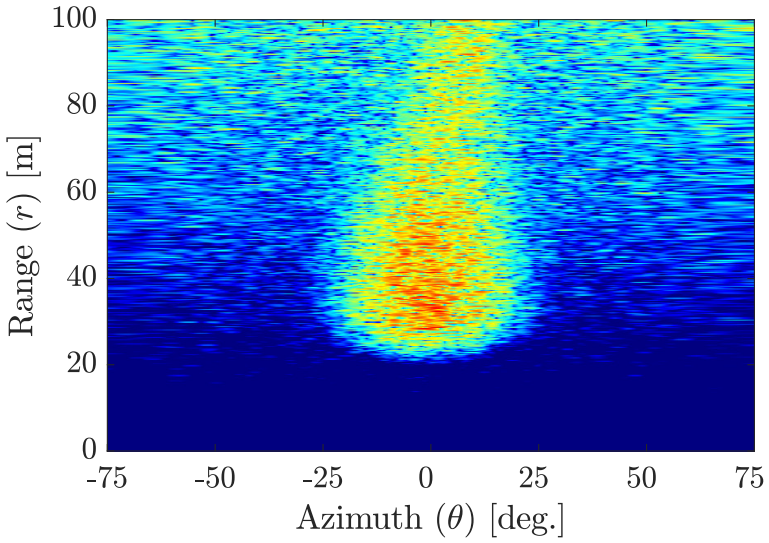


Figure 7.5: Simulated Range-Azimuth SONAR image that corresponds to the Cartesian image in Figure 7.4. Blue regions correspond to low signal intensities; red regions correspond to high signal intensities. Image was generated by the simulation environment described in Section 7.4.

Then, these ranges' depression angles can be determined. Yet this requires *a prori* estimates and an additional sensor measurement to maintain robustness. In particular, a single-beam echo sounder (or equivalent) is required to provide seafloor distance measurements. Furthermore, this method requires estimates of the vehicle's pitch and roll. Together, they help relate the SONAR's range measurements to the inertial frame, i.e. the seafloor. For brevity, the SONAR's and vehicle's reference frames will be assumed to be roughly co-located here and for the rest of this chapter. Thus, their reference frames will only differ by the SONAR's depression angle with respect to the vehicle, $\phi_{S/V}$. The vehicle's pitch angle, Θ, is its depression angle from its seafloor horizon. Therefore, the SONAR's depression angle with respect to its seafloor horizon, $\phi_{S/H}$, can be estimated via Equation 7.5:

$$\phi_{S/H} = \phi_{S/V} + \Theta \tag{7.5}$$

Combined, the SONAR/vehicle, the echo sounder's seafloor target, and the SONAR's "edge" seafloor target form three corners of a triangle. This triangle can be projected according to the vehicle's roll to yield the triangle in Figure 7.6:

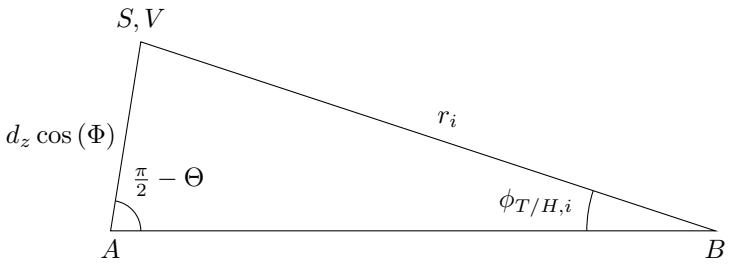


Figure 7.6: Geometry relating the SONAR/vehicle (S, V), the projected single-beam echo sounder seafloor target (A), and the SONAR's seafloor target (B) for a relatively flat seafloor.

where d_z represents the distance to the seafloor from the vehicle, V, along the *vehicle*'s vertical z-axis. The

vehicle's roll angle is represented by Φ. The i^{th} target's range from the SONAR (as extracted from the Range-Azimuth image) is again represented by r_i; its depression angle from the SONAR's seafloor horizon is represented by $\phi_{T/H,i}$. The underlying assumption here is that the seafloor is relatively planar. Consequently, $\phi_{T/H,i}$ can be estimated using the law of sines, as shown in Equation 7.6:

$$\begin{aligned} \frac{\sin\left(\phi_{T/H,i}\right)}{d_z \cos\left(\Phi\right)} &= \frac{\sin\left(\frac{\pi}{2} - \Theta\right)}{r_i} = \frac{\cos\left(\Theta\right)}{r_i} \\ \Rightarrow \phi_{T/H,i} &= \arcsin\left(\cos\left(\Phi\right)\cos\left(\Theta\right)\frac{d_z}{r_i}\right) \end{aligned} \tag{7.6}$$

Using this equation, the depression angles of the SONAR's beamwidth edges can be calculated using the aforementioned minimum and maximum ranges. However, only one of these $\phi_{T/H,i}$ values are useful since only one corresponds to the edge in the Doppler-Azimuth image. This depression angle is the one closest to the SONAR's *a priori* translational velocity vector. Thus, only this "edge" depression angle (represented by $\phi_{E/H}$) will be utilized from this point forward.

This depression angle can be transformed into its equivalent in the SONAR's coordinate system, $\phi_{E/S}$. At first glance, this would appear to provide all information needed to apply Equation 7.4 to estimate $\vec{v}_S$. However, this only establishes angle and range rate correspondences for one depression angle. Consequently, from this information alone, the translational motion estimate is still unconstrained in one degree of freedom.[4]

Therefore, to permit valid estimates, one degree of freedom must be eliminated. A possibility is to only consider targets in the SONAR's xy-plane since, in that plane, v_z is eliminated from the range rate equation.[5] This means that the SONAR's v_x and v_y can be estimated given range rates in its xy-plane. Fortunately, these range rates can be estimated from the range rates in the edge points' plane. As implied in Equation 7.4, the range rate magnitude is maximized in the direction of the SONAR's translational velocity vector, $\vec{v}_S$. For a given azimuthal angle, the range rate then decreases according to the cosine of its depression angle deviation from $\vec{v}_S$. Consequently, the range rates in the SONAR's xy-plane can be estimated via the "scaling factor" in Equation 7.7:

$$\dot{r}_{xy,i} = \frac{\cos\phi_{S/\vec{v}_S}}{\cos\phi_{E/\vec{v}_S}}\dot{r}_{E,i} \tag{7.7}$$

where $\dot{r}_{xy,i}$ represents the i^{th} estimated range rate in the SONAR's xy-plane, $\phi_{S/\vec{v}_S}$ represents the depression angle of the SONAR with respect to $\vec{v}_S$, $\phi_{E/\vec{v}_S}$ represents the depression angle of the edge points with respect to $\vec{v}_S$, and $\dot{r}_{E,i}$ represents the i^{th} edge point's range rate. Thus, the algorithm extracts $\dot{r}_{xy,i}$ estimates using this methodology.

Conversely, if $\vec{v}_S$ resides *within* the SONAR's vertical beamwidth, the resulting estimation is much more limited. In these cases, the Doppler-Azimuth edge's depression angle can correspond to *any* depression angle within the beamwidth. Therefore, the aforementioned methodology cannot be used to resolve this ambiguity. In these cases, the algorithm must assume $\dot{r}_{xy,i} = \dot{r}_{E,i}$ to proceed. This assumption can introduce significant errors regarding the resulting estimate vector's *directionality*. Thus, for these cases, it may be better to only utilize the estimate's *magnitude* within the overarching sensor fusion framework. These cases will not be explored further in this chapter.

Estimating via Iteratively Reweighted Least Squares

Given corresponding range rates, azimuthal angles, depression angles, and weights, Equation 7.4 can be used within a Weighted Least Squares (WLS) framework to estimate translational motion. As previously discussed in Section 7.3.1, SONARODO utilizes range rate measurements in the SONAR's xy-plane where $\phi = 0$.

[4]Specifically, it is unconstrained in the direction orthogonal to the edge points' depression angle with respect to the SONAR.

[5]The depression angles in Equation 7.4, ϕ_i, describe the targets' depression angles with respect to the SONAR. Thus, in that context, $\phi_i = 0$ for targets in the SONAR's xy-plane.

While the WLS estimate is often sufficient, it can be corrupted by erroneous outliers in the Doppler-Azimuth image. Therefore, our method employs the more robust Iteratively Reweighted Least Squares (IRLS) framework instead. This framework iteratively downweighs points with larger range rate residuals, as they are more likely to be outliers. However, these weights should not quickly diverge from their initial signal intensity-based weights. Consequently, in our implementation, the weights for each iteration are calculated according to Equation 7.8:

$$\begin{aligned} w_{i,0} &= \eta_0 \, I(\dot{r}_i, \theta_i) \\ w_{i,k} &= \eta_k \, w_{i,k-1} / |e_{\dot{r},i,k}| \end{aligned} \tag{7.8}$$

where η_k is the k^{th} iteration's normalization factor[6] and $e_{\dot{r},i,k}$ is the i^{th} point's range rate residual, i.e. its range rate error.[7]

7.3.2 Rotational Motion Estimation

SONARODO's rotational motion estimation sub-process depends on spatial measurements; specifically, the Range-Azimuth SONAR images. Essentially, the algorithm detects the azimuthal changes of the seafloor environment to estimate the SONAR's own yaw rate. However, translational motion induces changes in both azimuth and range. Therefore, this must be compensated for beforehand. This is accomplished by remapping the Range-Azimuth images according to Section 7.3.1's translational motion estimates.[8]

Remapping the Previous Range-Azimuth Image

The core concept of the image remapping is that, once transformed, only azimuthal differences should remain between consecutive images. The relationship between azimuthal motion and the sensor's motion is presented without proof in Equation 7.9:

$$\dot{\theta}_i = \begin{bmatrix} \sin\theta_i / (r_i \cos\phi_i) \\ -\cos\theta_i / (r_i \cos\phi_i) \\ 0 \\ \tan\phi_i \cos\theta_i \\ \tan\phi_i \sin\theta_i \\ -1 \end{bmatrix}^T \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{7.9}$$

Consequently, using Equation 7.4 (for range) and Equation 7.9 (for azimuth) with the time between images, δt, the algorithm remaps the previous Range-Azimuth SONAR image to compensate for the current translational motion estimate. Note that ω_z is assumed to be zero for this transformation since it is the variable to be estimated. Furthermore, our implementation assumes $\phi_i \approx 0$ since its influence is negligible for image remapping. This significantly simplifies Equation 7.9 because it now only requires v_x and v_y estimates from the preceding translational motion subprocess. An image remapping example is shown in Figure 7.7.

[6] This is necessary to avoid numerical instabilities.
[7] This error is saturated to a minimum magnitude to avoid division by zero.
[8] These estimates are the only link between the two sub-processes.

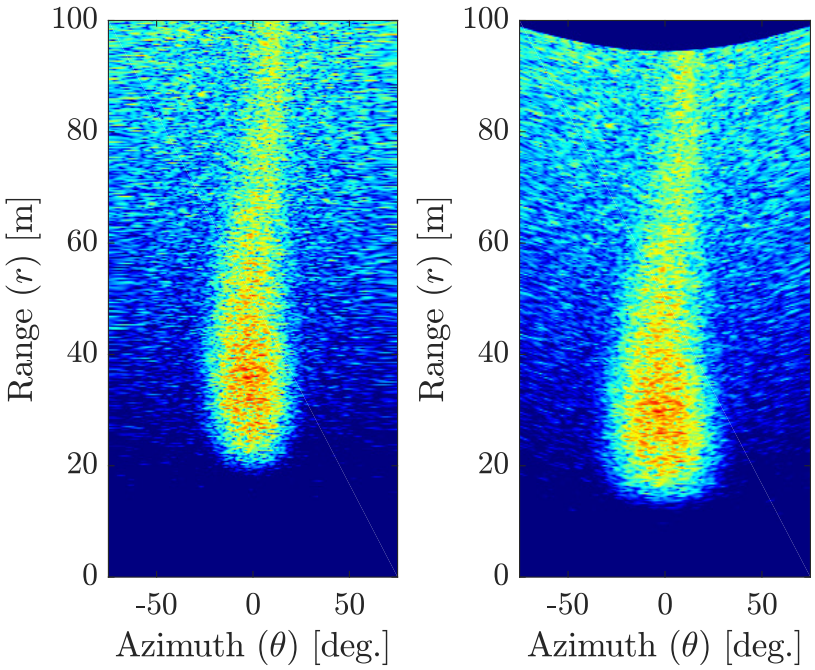


Figure 7.7: Remapping the previous Range-Azimuth image (left) to compensate for the current forward translational motion. Consequently, only azimuthal differences should remain between the resulting image (right) and the current image. Motion was scaled by a factor of 5 for visualization purposes.

Fourier-based Registration via Phase Correlation

After the previous image was remapped, corresponding targets in the current image should appear in the same range row. The azimuthal offset that aligns these images should then correspond to the relative azimuthal change, and thus, the SONAR's yaw rate.

For computational efficiency, SONARODO accomplishes this via Fourier-based phase correlation. This is because images are dense two-dimensional signals. Typically, phase correlation is employed to find translational offsets between noisy camera images by estimating their signals' phase offsets. Therefore, for SONAR images, our method uses remapped polar images to represent the azimuthal offset as an image translation in only one dimension.

Consequently, the algorithm applies the Discrete Fourier Transform (DFT) on the Range-Azimuth images to yield its frequency-domain representation. Essentially, the DFT decomposes the SONAR images into their equivalent weighted summation of sinusoidal functions. Therefore, due to this foundation, discrete image edges introduce challenging artifacts. Thus, the algorithm smoothes the SONAR image with a Hann function before its DFT is taken. This function smoothly preserves the image center while reducing its intensities near the image edges, as shown in Figure 7.8. Furthermore, the algorithm specifically shapes this Hann function to attenuate the beamwidth's nearest edge at its minimum range. Once the SONAR images have been prepped,[9] the algorithm performs their DFTs. This process is analytically described in Equation 7.10:

$$\begin{aligned} \mathbf{G}_{k-1,r} &= \mathscr{F}\{g_{k-1,r}\} \\ \mathbf{G}_{k} &= \mathscr{F}\{g_{k}\} \end{aligned} \tag{7.10}$$

where $\mathscr{F}\{\}$ represents the Discrete Fourier Transform operation, g_k represents the current Range-Azimuth image, $g_{k-1,r}$ represents the remapped previous Range-Azimuth image, and $\mathbf{G}_k$ and $\mathbf{G}_{k-1,r}$ represents their

[9]This preparation process also includes zero-padding the images to their nearest so-called "optimal" DFT size.

respective frequency-domain representations.

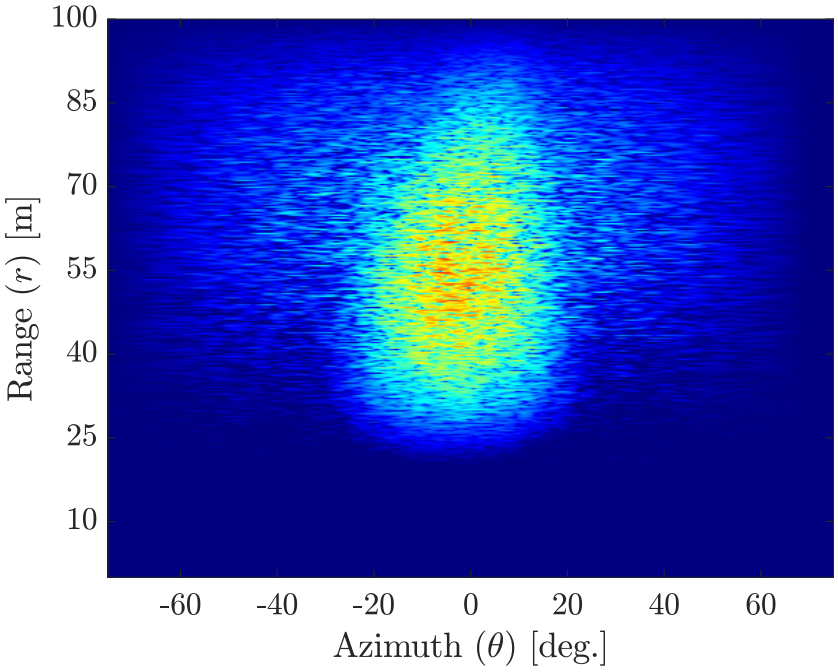


Figure 7.8: The SONAR Range-Azimuth image from Figure 7.5 after being smoothed by a Hann function.

Then, the algorithm calculates the images' normalized cross-power spectrum, $\mathbf{R}$, as shown in Equation 7.11:

$$\mathbf{R} = \frac{\mathbf{G}_{k-1,r} \circ \mathbf{G}_k^*}{|\mathbf{G}_{k-1,r} \circ \mathbf{G}_k^*|} \tag{7.11}$$

where $\circ$ represents the Hadamard product (i.e. entry-wise product) and * represents the result of complex-conjugation. The resulting $\mathbf{R}$ image represents the images' shared frequencies. Thus, this includes the high-frequencies of SONAR image speckle. As this does not correspond to the seafloor's relative motion, its influence should be minimized. Thus, the algorithm smoothes $\mathbf{R}$ by another Hann function to attenuate these high-frequency signals.[10] The algorithm then takes the Inverse Discrete Fourier Transform (IDFT) of its result, as analytically described in Equation 7.12:

$$r_a = \mathscr{F}^{-1}\{\mathbf{R}_a\} \tag{7.12}$$

where $\mathscr{F}^{-1}\{\}$ represents the Inverse Discrete Fourier Transform operation, $\mathbf{R}_a$ is the attenuated $\mathbf{R}$, and r_a represents the resulting phase correlation matrix.

Thus, the pixel location of phase correlation matrix's maximum corresponds to the estimated image translation. Due to the aforementioned polar image remapping, only azimuthal offsets are valid. Consequently, since the first row corresponds to a zero range offset, the algorithm *only* searches this row for its maximum. This unique one-dimensional search is critical for avoiding erroneous estimates.

[10] In lieu of shifting the cross-power spectrum so that its zero-frequency components are in the center of the image, the Hann function is shifted instead to ensure that zero-frequency components are passed through.

Estimating the Yaw Rate with Sub-pixel Resolution

For improved accuracy, the phase correlation maximum can be estimated in sub-pixel resolution via peak refinement techniques. To do so, the algorithm extracts the maximum pixel along with its two neighbors to form a three-element vector.[11] The vector's pixel intensities are then used to calculate their pixel coordinates' weighted centroid. This centroid is again the location of the correlation maximum, but now with sub-pixel resolution.

The algorithm then converts the correlation maximum pixel location to its equivalent azimuthal angle offset, $\delta\theta$. Since this process assumes cyclic signals, pixel locations on the right side of the image actually correspond to negative offsets. The SONAR's yaw rate is then calculated by transforming Equation 7.9 to the simplified and discretized variant shown in Equation 7.13:

$$\omega_z = -\delta\theta/\delta t \tag{7.13}$$

7.4 Validation

SONARODO was validated using simulated data generated by Penn State Applied Research Laboratory's point-based SONAR scattering model [135,136]. The SONAR images previously shown, most notably Figures 7.4, 7.3a, and 7.5, were all generated by this simulation framework.

The high-fidelity Range-Azimuth images were generated by simulating signal returns and then processing those returns, as described in [135, 136]. Conversely, the Doppler-Azimuth images were generated by a simulation framework with a much lower fidelity. Specifically, they were generated by simulating signal return intensities as described by [135, 136] but then calculating their corresponding ground truth range rates in lieu of signal processing. This methodology was considered to be sufficient for validating this algorithm's methodology. However, validating this method's Translational Motion Estimation sub-process based on higher fidelity Doppler-Azimuth images remains a possible extension of this research.

7.4.1 Simulation Characteristics

The Doppler-Azimuth and Range-Azimuth images were both provided by a simulated 2D Forward-Looking SONAR with the specifications in Table 7.1. The sensing trade-offs associated with using Doppler-Azimuth images are discussed in Section 7.5.2.

Field-of-View	150° Horizontal, 20° Vertical
Max. Range	100 m
Range Resolution	0.05 m
Azimuth Resolution	0.5°
Range Rate Resolution	0.01 m/s
Update Rate	1 Hz
Frequency	900 kHz
Doppler-Azimuth Image Size	301×265 $px.$
Range-Azimuth Image Size	301×2000 $px.$

Table 7.1: Simulated 2D SONAR Specifications

An Autonomous Underwater Vehicle (AUV) was simulated traversing a path while equipped with the aforementioned SONAR. This Forward-Looking SONAR was mounted near the vehicle's center of buoyancy (CB) at a depression angle of 15°. The vehicle traversed the 148 m path in 157 $sec.$ at an average speed of

[11] Since this process assumes cyclic signals, the left edge pixel's neighbor is actually the right edge pixel, and vice versa.

0.94 m/s (1.8 $knots$). The vehicle traveled at varying yaw rates up to 9.1 $^\circ/s$ and at a constant depth that was, on average, 12.3 m above the seafloor.[12]

The vehicle's environment (and its path) are visualized in Figure 7.9. As the figure shows, the simulated environment was a seafloor characterized by seamounts, but lacking features with distinct edges. The seafloor was modeled to be consistent with High Frequency Environmental Acoustics (HFEVA) model for "medium sand" [137]. Within the 250×250 m environment, the seafloor altitude's standard deviation was 1.5 m.

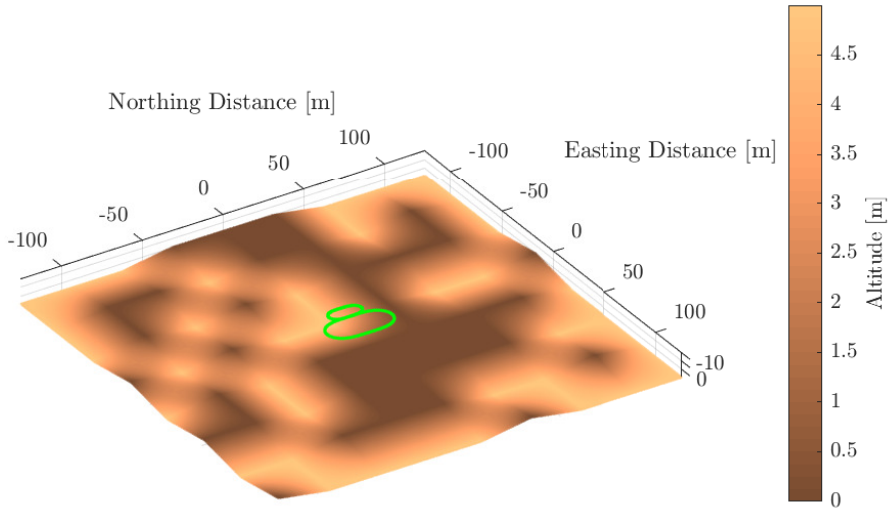


Figure 7.9: Path of the simulated AUV (green) and its seafloor environment. See Figure 7.13 for a clearer visualization of the path itself.

7.4.2 Estimate Accuracy

SONARODO's estimates of the SONAR's longitudinal velocities for the aforementioned path were plotted in Figure 7.10 alongside ground truth. Similarly, the SONAR lateral velocity estimates were plotted in Figure 7.11. As the figures demonstrate, both longitudinal and lateral velocity estimates closely mirrored ground truth, as they each yielded an error of 0.000 ± 0.002 m/s. The SONAR yaw rate estimates were plotted in Figure 7.12. Again, estimates closely matched ground truth, yielding an error of 0.04 ± 0.24 $^\circ/s$.

[12]For this simulation, the AUV did not exhibit roll or pitch.

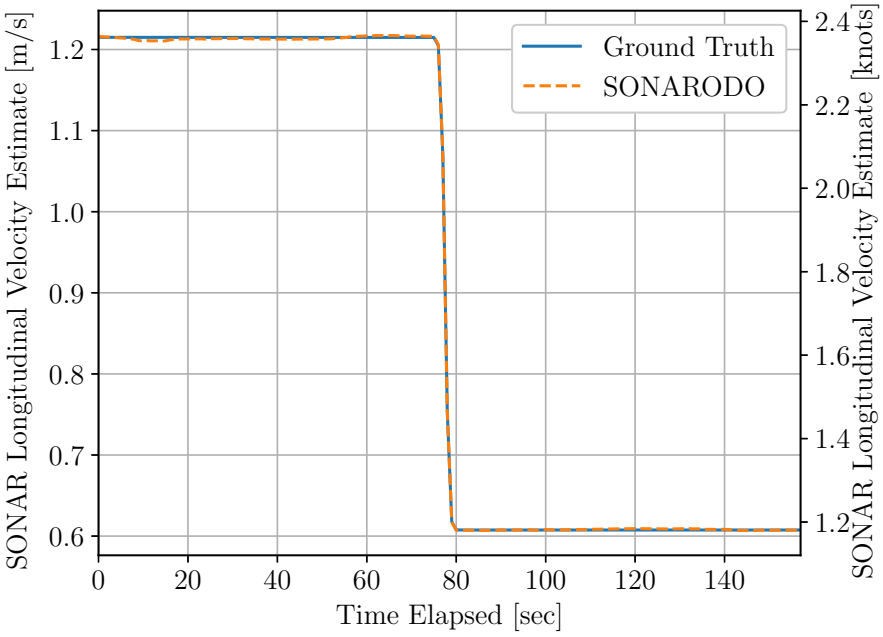


Figure 7.10: Ground truth and SONARODO estimates of the SONAR's longitudinal velocity for the simulation path.

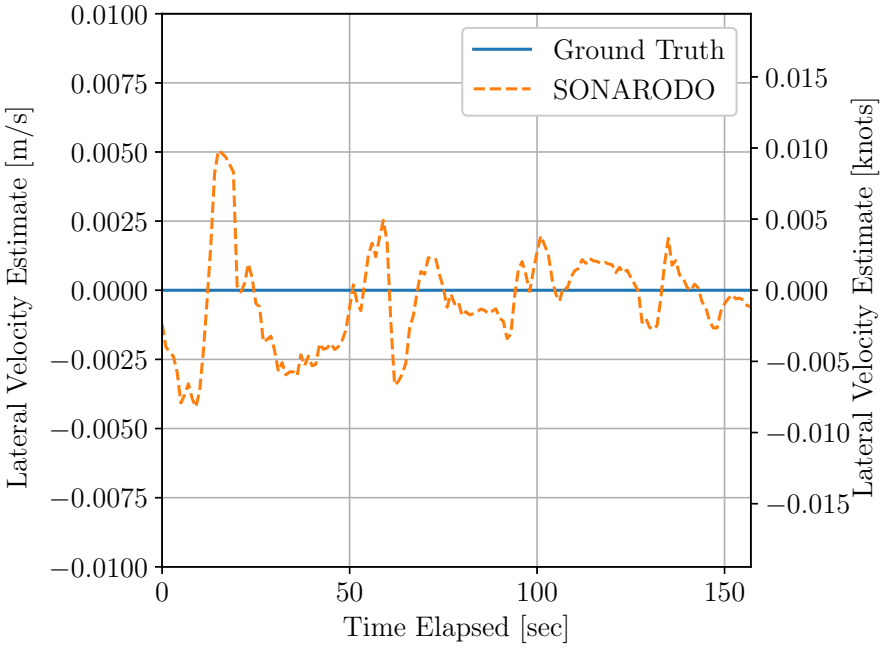


Figure 7.11: Ground truth and SONARODO estimates of the SONAR's lateral velocity for the simulation path.

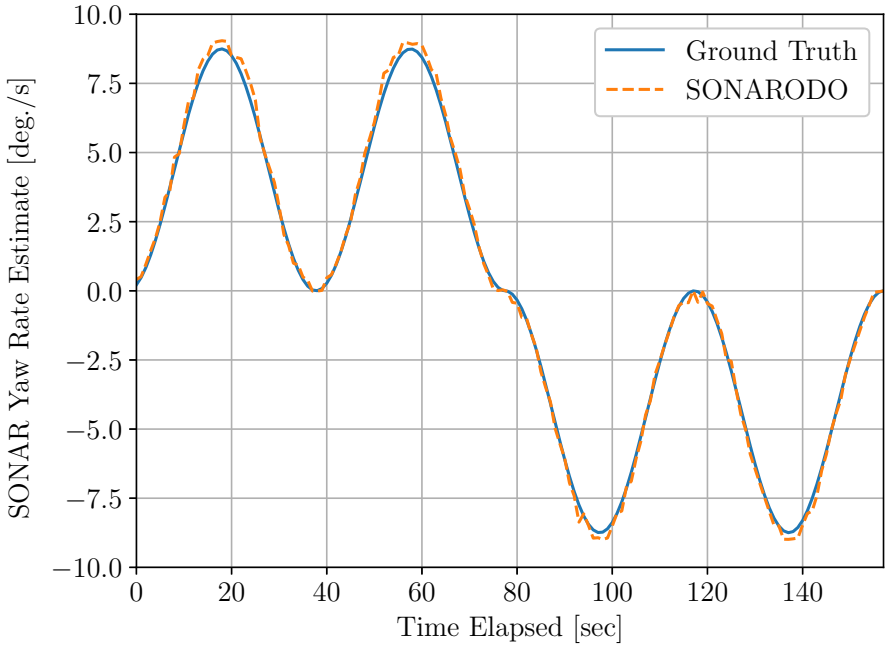


Figure 7.12: Ground truth and SONARODO estimates of the SONAR's yaw rate for the simulation path.

A direct comparison with Hurtós et al.'s related method is complicated because Hurtós et al. used different environments and three widely varying SONARs. For unit equivalency, velocities were transformed into equivalent translations and rotations. For reference, Hurtós et al.'s mean longitudinal and lateral translation errors were 0.11 m and 0.08 m, respectively.[13] Thus, SONARODO achieved significantly greater translation accuracy with its translation errors of 0.000 ± 0.002 m. For rotation, Hurtós et al.'s method yielded a mean error of 0.36° [78].[14] Again, SONARODO achieved greater accuracy with its rotational errors of 0.04 ± 0.24 °. It is worth repeating that our method depends on Doppler measurements while Hurtós et al.'s method does not. Consequently, SONARODO is perhaps better suited for motion estimation while Hurtós et al.'s method is better suited for its intended use case of image mosaicing.

SONARODO's estimates were furthermore utilized for dead-reckoning navigation. However, as previously described, its foundational 2D SONAR could only yield the SONAR's $[v_x\ v_y\ \omega_z]$ estimates. Without estimates in all six degrees-of-freedom, it could not be directly transformed into the vehicle's motion. Consequently, to permit a reasonable comparison, the *missing* states were populated by their ground truth values. Therefore, its dead-reckoning navigation estimates are plotted alongside ground truth in Figure 7.13. As the figure shows, the estimates emulate the general ground truth path, with a final dead-reckoning percent error of 0.4 %. It is noteworthy that an overestimated yaw rate during the first turn results in an early small heading offset, incorrectly rotating the rest of its path estimates. Fortunately, the impact of these estimate outliers would be minimized within an overarching sensor fusion framework.

[13] This was calculated by averaging the "mean errors" reported from all three of its datasets.
[14] This was calculated by averaging the "mean errors" reported from all three of its datasets.

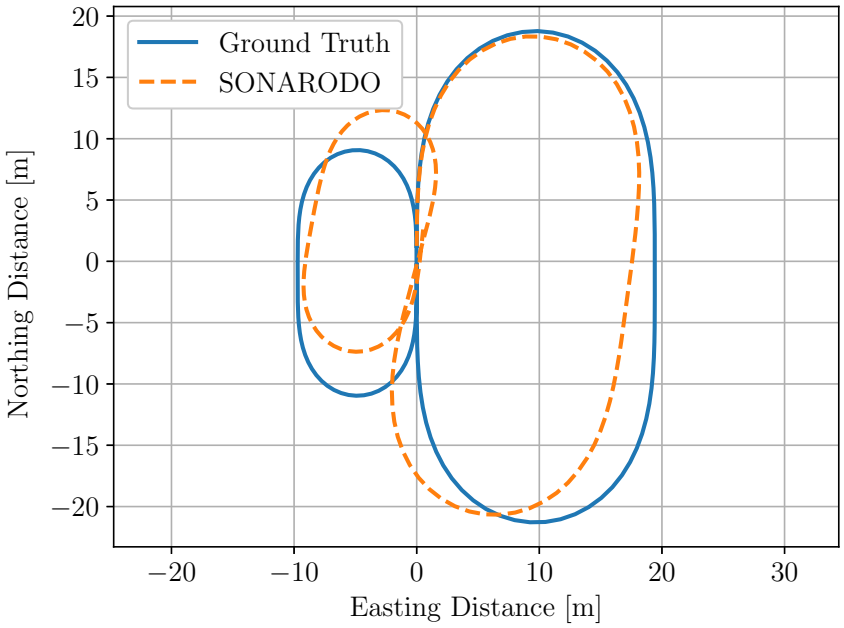


Figure 7.13: Ground truth and SONARODO-based dead-reckoning navigation estimates of the AUV's path.

7.4.3 Computational Performance

For this validation, the Doppler-Azimuth and Range-Azimuth images were generated offline using the aforementioned simulation framework. Then, SONARODO processed these recorded images in real-time. Specifically, it executed as a Robotic Operating System (ROS) [121] C++ node running on an Intel Core i7-7500U 64-bit CPU (2.70 GHz). It primarily utilized the open-source computer vision library, OpenCV [126]. In this environment, SONARODO's mean runtime was $59 \pm 2\ ms$, corresponding to an average measurement frequency of 17 Hz. This can be considered real-time for many AUVs and is a fraction of practical sound wave propagation times.

This is similar to Hurtós et al.'s typical registration runtime of 60 ms. However, for this validation, the equivalent Cartesian images would have been $3,864 \times 2,000\ px$. This is much larger than Hurtós et al.'s Cartesian images, as they ranged from $350 \times 274\ px$. to $1,526 \times 848\ px$. [76,78]. Thus, our method's reduced computational cost is most likely attributed to it only requiring one Fourier-based registration while Hurtós et al.'s method requires two. These registrations are costly; SONARODO's rotational sub-process alone occupied the vast majority (97%) of its total runtime.

7.5 Implementation Discussion

This section discusses how SONARODO could be implemented in practice.

7.5.1 Integration within a Sensor Fusion Framework

As previously noted, SONARODO's use of a 2D SONAR limits its estimates to the SONAR's v_x, v_y, and ω_z motion. If the SONAR and vehicle have different reference frames, the limited SONAR motion states cannot be directly transformed into vehicle motion. Thus, this may result in increased complexity when integrating this method's estimates within an overarching sensor fusion framework. If its estimates were

integrated within a loosely-coupled framework, *a priori* SONAR motion estimates would be required to fill in the missing states. Consequently, a tightly-coupled framework may be a more suitable implementation. These frameworks are better suited for "partial" sensor measurements and often yield improved navigation accuracy [82].

7.5.2 Sensing Trade-offs for Translational Motion Estimation

Section 7.3.1 detailed the translational motion estimation sub-process. Due to the 2D SONAR's lack of depression angle measurements, this technique also required single-beam echo sounder measurements and *a priori* measurements. These measurements are critical for calculating the "scaling factor" in Equation 7.7. Fortunately, due to the cosine terms that comprise the scaling factor, small *a priori* measurement inaccuracies have only a minimal effect.

Regardless, this phenomenon exposes the fundamental trade-off of the SONAR's vertical field-of-view. To avoid angle ambiguities, DVLs are designed to have a very focused beamwidth to estimate translational motion. Conversely, however, SONARODO's rotational motion estimation (and environmental perception tasks) depend on sizeable beamwidths to provide informative SONAR images. Furthermore, SONARs are subject to hardware and algorithmic tradeoffs that dictate their Doppler and range measurement resolutions. Thus, there may be applications in which it is better to use DVL measurements in lieu of this method's translational motion estimation. At its core, this agrees with SONARODO's methodology: utilize measurements for their respective strengths. Therefore, SONARODO is perfectly compatible with this implementation; its rotational motion estimation is invariant to the source of v_x and v_y.

Thus, the presented translational motion estimation sub-process was based on Doppler-Azimuth images to offer implementation flexibility. Specifically, SONARODO was designed to promote using the SONAR sensor for both environmental perception and motion estimation without a separate DVL. Consequently, for the end user, the optimal implementation depends largely on its AUV's sensor suite.

7.5.3 Integration within a Pose Graph Framework

SONARODO offers fast and accurate motion estimates that are critical for constraining "odometry" drift. Despite this, even minimal odometry errors accumulate over time. Thus, this drift can be constrained even further by integrating pose estimates and "keyframe" SONAR images within a pose graph framework.[15]

Keyframe-based optimization is a variant of bundled-window optimization with a focus on real-time operation. In this context, it constrains distant non-consecutive poses with their corresponding "keyframes." Thus, the constraint between these two poses restrain the accumulated odometry drift from all measurements in between. Fortunately, thid method's *rotational* motion estimation sub-process is well-suited for these rotational constraints. In practice, it can utilize the framework's translation estimates for image remapping. This methodology is particularly useful for SONAR images with poor angular resolutions since they can only detect large rotations.

7.6 Conclusion to SONARODO

This chapter presented SONARODO, a novel motion estimation algorithm for 2D Forward-Looking SONARs. The presented validation results with high-fidelity simulation data demonstrated that it offers fast and accurate motion estimates. Specifically, it estimated translational velocities and yaw rates with standard deviations of 0.002 m/s and 0.24 $^\circ/s$, respectively, at a mean runtime of 59 ms.

This method was designed to offer a solution specifically tailored for motion estimation by better utilizing the strengths of different measurement domains. Furthermore, it was designed to be robust to featureless en-

[15]This methodology resides on a spectrum that is characterized on one end by Simultaneous Localization and Mapping (SLAM). However, since SONARODO focuses on featureless environments and does not assume loop closures, SLAM is outside the scope of this chapter.

vironments, low resolution SONAR images, and the source of its Doppler measurements. In all, SONARODO offers a promising motion estimation solution for a wide variety of applications.

Chapter 8

Conclusions

The previous chapters detailed several contributions that independently offered valuable insights. However, these contributions followed a natural progression over the course of this dissertation's research. Thus, their insights are perhaps best viewed holistically. To that end, this chapter details the collective insights, extensions, and applications of this research.

8.1 Main Insights

Collectively, this research yields the following five main insights:

1. **Doppler measurements offer fast and accurate estimates of relative radial motion without requiring point correspondences.** As discussed in Chapter 2, ego-motion estimation is a very robust and mature research field. However, most ego-motion estimation techniques are based on camera images and/or LIDAR measurements. Conversely, this research focused mainly on Doppler measurements, placing it in a niche field. This is likely due to many factors, one of which being that Doppler sensors only provide relative *radial* measurements. Thus, relative angular motion is unobservable, severely limiting Doppler sensors' applicability for ego-motion estimation.

   Despite these limitations, range rate measurements are readily-available from widely-utilized Doppler sensors for both underwater and automotive vehicles. Furthermore, these measurements are very accurate and do not require point/target correspondences to be established over time. This offers several benefits regarding accuracy, computational cost, and robustness.

   Consequently, the algorithms developed for this research focused on utilizing Doppler measurements' strengths while mitigating its weaknesses.

2. **A Doppler sensor can directly estimate its translational ego-motion; rotational ego-motion can be estimated only under certain circumstances.** A Doppler sensor's translational ego-motion can be estimated from its range rate measurements. Conversely, its rotational ego-motion cannot be directly estimated because relative angular motion is unobservable from Doppler measurements. However, Chapter 3 demonstrated that rotational ego-motion can be *indirectly* estimated if the Doppler sensor is mounted at a lever-arm offset from the vehicle's reference frame.

   Furthermore, as briefly discussed in Chapters 5 and 6, direct rotational ego-motion estimation is possible with *multiple* distributed Doppler sensors. For instance, two distributed Doppler sensors can estimate motion in all degrees of freedom except rotation about the axis connecting the two sensors. This is particularly useful for autonomous automotive vehicles that have more than one RADAR.

3. **For many applications, spatial measurements provide the best, if not only, option for rotational ego-motion estimation.** The ability to estimate (or correct) rotational ego-motion from Doppler measurements depends on several factors: the number of Doppler sensors, vehicle motion assumptions, the lever-arm offset of the sensor(s), and the sensor fusion framework's uncertainties.

Depending on the application, these factors could yield posterior estimates with impractically high uncertainties. In these cases, spatial measurements may provide a better option.

First, *single* Doppler sensors will be discussed. Rotational ego-motion can be uniquely estimated from a single Doppler sensor only under strict vehicle motion assumptions. As discussed in Chapters 3, 5, and 6, these assumptions are often violated for automotive vehicles and simply inapplicable for underwater vehicles. Without these assumptions, vehicle ego-motion is ambiguous.

Therefore, since these measurements do not correspond to a unique vehicle ego-motion estimate, they must rely on their overarching sensor fusion framework to resolve their ambiguities. For instance, *a priori* estimates could be used to resolve ambiguities within a loosely-coupled framework. However, Chapter 3 demonstrated that tightly-coupled frameworks are better suited for these "partial" measurements and thus yield improved navigation accuracy.

Ambiguities are quantitatively described by the framework estimates' uncertainties. In this case, the magnitude of the Doppler sensor's lever-arm offset dictates the relative observability of its ego-motion states. Thus, larger lever-arms reduce the vehicle's rotational ego-motion uncertainty while sacrificing its translational ego-motion uncertainty. Yet, Chapter 3 demonstrated that rotational ego-motion accuracy is more important for navigation accuracy due to the nature of dead-reckoning navigation.

Unfortunately, many AUVs are simply not large enough for their lever-arm offsets to permit rotational ego-motion estimates with low uncertainties. Even if their lever-arms were sufficiently large, like they are for automotive RADARs, rotational ego-motion uncertainties would still have a lower limit. A tightly-coupled framework can only resolve ambiguities within the bounds of their *a priori* estimates' uncertainties. Thus, the *posterior* ego-motion estimate uncertainties may still be too high. It is for these common applications that spatial measurements provide the best, if not only, option for rotational ego-motion estimation. Unlike Doppler measurements, spatial measurements can directly observe both translational and rotational ego-motion. Consequently, these common applications were the focus of this dissertation's research.

Conversely, for *multiple* Doppler sensors, spatial measurements may only be the best option for small lever-arm offsets.

4. **Correspondence-based spatial techniques are ill-suited for RADAR and SONAR images. Thus, depending on whether the image contains discontinuities, either correspondence-free iterative methods or global searches are better suited.** Visual Odometry depends on reliably establishing correspondences between distinct features in camera images. Once these correspondences have been established, there is a closed-form solution for ego-motion. Thus, Chapter 6 used this methodology on camera images. Conversely, SONAR and RADAR images have characteristically low resolutions and signal-to-noise ratios, making them ill-suited for feature-based registration techniques.

   Thus, in these scenarios, correspondence-free techniques are better suited. For instance, as shown in Chapter 4, Geometric Dense Visual Odometry techniques depend on the discrete derivatives of range images. These derivatives are used within a rigid "coarse-to-fine" estimation framework. However, these methods fundamentally assume that feature movement (caused by ego-motion) is limited between frames. Therefore, violations of this constraint can unknowingly corrupt the ego-motion estimation solution.

   These violations are most troublesome at discrete edges since their calculated derivatives do not correspond to the true ego-motion. Conversely, if the image represents a continuous surface, the calculated derivatives should at least serve as a fair approximation for an iterative process. Consequently, iterative approaches may be better suited if its source images are continuous. Fortunately, RADAR/SONAR images are relatively continuous due to inherent uncertainties and the nature of their processing. For this reason, Chapter 5 used an iterative optimization to register RADAR images.

   Despite this, image "speckle" can corrupt continuity, particularly in SONAR images. While image blurring is an option, it comes at a steep accuracy and computational cost. Therefore, in these cases,

global techniques like Fourier-based registration may be better suited. In fact, Chapter 7 demonstrated its successful use on SONAR images.

5. **This research's developed algorithms largely depend on analytical decouplings to improve the benefit-to-cost ratio of accuracy and computational cost.** The algorithms presented in Chapters 5, 6, and 7 all rely on the same premise: decoupling ego-motion estimation to utilize different measurement domains for their respective strengths. Specifically, they utilize Doppler and spatial measurements for translational and rotational ego-motion estimation, respectively.

   This concept was largely founded on the fact that angular motion is unobservable from Doppler measurements. Consequently, Doppler measurements were used for translational ego-motion estimation due to its invariance to rotational ego-motion. Spatial data was used for rotational ego-motion estimation due to spatial data's unique ability to observe it. However, this required removing the translational ego-motion's influence from the spatial data. Both the algorithms presented in Chapters 5 and 7 accomplished this by "remapping" RADAR/SONAR images accordingly. Chapter 6's algorithm applied a different approach for camera images. It decomposed the estimated Essential Matrix into its individual ego-motion states.

   This decoupling provided two main benefits. First, it improved accuracy by utilizing characteristically accurate Doppler measurements. It also greatly reduced or eliminated the influence of inaccurate range measurements/estimates from SONAR, RADAR, or camera images. Secondly, the computational cost was reduced by eliminating the processes required for spatial translational ego-motion estimation. For Chapter 5's algorithm, two degrees of freedom were eliminated from the optimization. For Chapter 6's algorithm, feature tracking and depth estimation were no longer required. For Chapter 7's algorithm, one Fourier-based registration was eliminated.

   As previously stated, spatial measurements provide redundant information about translation. Therefore, for accuracy, a truly "optimal" solution incorporates all possible measurements. However, it is often the case that Doppler-based translational ego-motion estimates are much more accurate than their spatial counterparts. Thus, a fusion of the two would be dominated by the Doppler-based estimate anyway. For this reason, spatial-based estimates are often not worth their corresponding computational cost. As mentioned in Chapter 1, this is critical for practical ego-motion estimation since it maximizes the benefit-to-cost ratio of accuracy and computational cost.

8.2 Possible Extensions

This section discusses possible extensions to this research that were deemed to be outside of this dissertation's scope.

- *Validation with real RADAR images:* As mentioned in Chapter 5, the author was unaware of any commercially-available automotive-grade RADARs that publish its internal RADAR images in a practically accessible manner. Instead, RADAR images were emulated using real extracted point targets. Therefore, a possible extension for this research is a RADARODO validation for real RADAR images.

- *Comparison with Fourier-based registration for RADAR images:* Chapter 5's RADARODO algorithm used iterative non-linear optimization techniques to register RADAR images. This approach was considered to be appropriate because RADAR images usually exhibit relatively high signal-to-noise ratios. Furthermore, their images are usually populated by either the high signal intensities of metallic vehicle/poles or the negligible signal intensities of empty space. Conversely, Chapter 7's SONARODO algorithm used Fourier-based registration for SONAR images. This approach was specifically deployed to handle the low signal-to-noise ratios of SONAR images.

  Since this research lacked real RADAR images, Fourier-based registration techniques were not deployed for RADAR images. Therefore, it remains an open question whether Fourier-based registration techniques are better suited for RADAR images. Fourier-based registration is generally better suited

for very large images and/or images with low signal-to-noise ratios [138]. Consequently, this possible extension could yield yet another novel RADAR-based algorithm.

- *Attenuate only the frequencies that correspond to SONAR image speckle:* For Chapter 7's SONARODO algorithm, a Hann function was used to attenuate high frequencies in the cross-power spectrum image. This was done primarily to mitigate the influence of SONAR image speckle. However, due to the Hann function's gradual nature, it also attenuated high-frequency components of the seafloor, albeit to a lesser extent. Yet, these high-frequency components could correspond to the distinct edges of man-made objects on the seafloor. Thus, these objects could provide valuable pose information that should not be rejected by attenuation.

  Therefore, if the frequencies corresponding the SONAR's image speckle are known, they can be attenuated more precisely. This could be accomplished via a specifically-tailored Tukey function. Consequently, this is a possible extension for this research.

- *Comprehensive covariance estimation:* As mentioned in Chapter 1, the methods utilized in Chapters 5, 6, and 7 were only considered if they could support corresponding covariance estimation. For this dissertation, covariance estimation was either explicitly defined or already established in literature.

  RADARODO's non-linear optimization framework naturally calculates its pose estimates' uncertainties. For rotational ego-motion, this process is explicitly detailed in Chapter 5. Since RADARODO and MonoRAD share the same translational ego-motion estimation process, their covariance estimation process was detailed in Chapters 5 and 6. Conversely, MonoRAD's rotational ego-motion estimation depends on camera images and the epipolar geometry. Fortunately, its corresponding covariance estimation process is detailed in [139, 140]. For Chapter 7's SONARODO, translational ego-motion estimation is based on solving a linear system, thus its covariance estimation is straightforward. For SONARODO's Fourier-based rotational ego-motion estimation, the corresponding covariance estimation process is detailed in [78]. Consequently, a possible extension would be to incorporate the missing covariance estimation processes into this research's algorithms.

- *Comparison with the use of multiple distributed Doppler sensors:* The benefit-to-cost ratio dichotomy was striking between the Doppler-based and spatial-based sub-processes in Chapters 5, 6, and 7. Doppler-based processes consistently offered very accurate estimates for a very low computational cost. For this reason, purely Doppler-based techniques should always be explored first. Yet, for many applications with a single Doppler sensor, uncertainty requirements demand the aid of spatial techniques. Therefore, this was this research's focus.

  Consequently, *multiple distributed* Doppler sensors were outside of this research's scope. However, multiple Doppler sensors are much more likely to be able to justify a purely Doppler-based solution. For underwater vehicles, this can be accomplished with multiple DVLs or individual distributed transducers in lieu of single DVL unit. The crossover point that dictates whether spatial data should be utilized largely depends on the sensors' lever-arm offsets. Thus, this comprehensive comparison remains a possible extension for this research.

- *Integration within a pose graph framework:* As mentioned in Chapter 1, ego-motion estimation exists on one end of a spectrum that is characterized on the other end by Simultaneous Localization and Mapping (SLAM). While ego-motion estimation focuses on real-time performance, SLAM focuses more on global consistency. In between these two ends, there are methods like "windowed bundle adjustment" that depend on constraints between sparse keyframes. Pose graph frameworks, like *g2o* [119], are well-suited to perform the optimizations necessary for windowed bundle adjustment and SLAM.

  Unfortunately, the Doppler-based translational ego-motion estimation sub-processes in Chapters 5, 6, and 7 are currently ill-suited for these optimization approaches. This research's utilization of Doppler measurements was largely responsible for its algorithms' reduced computational costs. However, since pose graph frameworks are typically suited for poses and spatial measurements, their optimization is better suited for spatial-based approaches.

Fortunately, the spatial-based rotational ego-motion estimation sub-processes in Chapters 5, 6, and 7 are well-suited for these approaches. Their inclusion in the pose graph framework can help further constrain heading drift.

A truly comprehensive pose graph framework utilizes different techniques for their respective strengths to maximize *both* speed and accuracy. Specifically, it operates asynchronously; utilizing fast techniques for its more frequent front-end processes and slower techniques for its less frequent but more consistent back-end optimizations. Inertial measurements, odometry, and the ego-motion estimation algorithms presented in this research could serve as its faster processes. Conversely, the spatial-based methods detailed in [28, 29, 43, 73, 74, 76, 78] are better suited for more slowly optimizing all degrees of freedom. Consequently, the integration of this dissertation's algorithms into a pose graph framework remains a possible extension.

8.3 Possible Applications

Since ego-motion estimation resides at such a fundamental level of an autonomous vehicle's localization framework, many processes can benefit from ego-motion estimate improvements. For instance, improved algorithms can provide faster and/or more accurate *a priori* estimates for global map-matching methods. Conversely, in the absence of map landmarks and/or reliable global measurements, these algorithms can help restrain odometry drift.

Specifically, for automotive vehicles, MonoRAD was designed to estimate ego-motion using a popular sensor suite: a monocular camera and RADAR sensor. In the event of inclement weather, that same vehicle can turn to RADARODO since it estimates ego-motion using only a single RADAR. For underwater vehicles, Chapter 3 detailed how ego-motion can be estimated from a single DVL. It even discusses how its insights could lead to a more optimal AUV design. In the scenario that purely DVL-based techniques are not sufficient, an underwater vehicle can deploy SONARODO. SONARODO was designed to utilize only a Forward-Looking SONAR or a combination of a FLS SONAR and a DVL.

Therefore, in all, this dissertation presented comprehensive research on ego-motion estimation for automotive and underwater vehicles. It also presented novel ego-motion estimation algorithms that are individually and collectively robust to a wide variety of scenarios. In conclusion, while some open questions remain, this dissertation supports ego-motion estimation's practical application to automotive and underwater vehicles.

Bibliography

[1] S. Tech, "TELEDYNE BlueView," 2019.

[2] A. Nordman, "Epipolar geometry," 2006.

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

[4] A. Eskandarian, *Handbook of Intelligent Vehicles*, vol. 2. London: Springer-Verlag London, 2012.

[5] L. Whitcomb, D. Yoerger, and H. Singh, "Advances in Doppler-based navigation of underwater robotic vehicles," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 1, pp. 399–406, 1999.

[6] J. C. Kinsey and L. L. Whitcomb, "Preliminary field experience with the DVLNAV integrated navigation system for oceanographic submersibles," *Control Engineering Practice*, vol. 12, no. 12 SPEC. ISS., pp. 1541–1549, 2004.

[7] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the Mars Exploration Rovers," *Journal of Field Robotics*, 2007.

[8] P. N. Denbigh, "Ship velocity determination by Doppler and correlation techniques," *IEE Proceedings F: Communications Radar and Signal Processing*, vol. 131, pp. 315–326, jun 1984.

[9] J. J. Leonard and A. Bahr, "Autonomous underwater vehicle navigation," in *Springer Handbook of Ocean Engineering*, pp. 341–357, Springer International Publishing, jan 2016.

[10] O. J. Woodman, "Introduction to Inertial Navigation," tech. rep., University of Cambridge, 2007.

[11] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, and B. Huhnke, "Junior: The Stanford Entry in the Urban Challenge," *Journal of Field Robotics*, vol. 25, no. 9, 2008.

[12] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer Berlin Heidelberg, 2nd ed., 2017.

[13] D. L. Simon, *Optimal state estimation: Kalman, H [infinity] and nonlinear approaches*. Hoboken, NJ: Wiley-Interscience, 2006.

[14] F. Raudies and H. Neumann, "A review and evaluation of methods estimating ego-motion," *Computer Vision and Image Understanding*, vol. 116, no. 5, pp. 606–633, 2012.

[15] N. H. Khan and A. Adnan, "Ego-motion estimation concepts, algorithms and challenges: an overview," *Multimedia Tools and Applications*, 2017.

[16] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intelligent Transportation Systems Magazine*, 2010.

[17] D. Scaramuzza and F. Fraundorfer, "Visual Odometry: Part I: The First 30 Years and Fundamentals," *IEEE Robotics and Automation Magazine*, vol. 18, pp. 80–92, dec 2011.

[18] J. C. Kinsey, R. Eustice, and L. L. Whitcomb, "A Survey of Underwater Vehicle Navigation: Recent Advances and New Challenges," *7th Conference on Manoeuvring and Control of Marine Craft (MCMC'2006)*, pp. 1–12, 2006.

[19] R. H. Rasshofer and K. Gresser, "Automotive radar and lidar systems for next generation driver assistance functions," *Advances in Radio Science*, 2005.

[20] J. Steinbaeck, C. Steger, G. Holweg, and N. Druml, "Next generation radar sensors in automotive sensor fusion systems," in *2017 Symposium on Sensor Data Fusion: Trends, Solutions, Applications, SDF 2017*, 2017.

[21] J. Engel, V. Koltun, and D. Cremers, "Direct Sparse Odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[22] H. P. Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," tech. rep., Stanford University, Stanford, CA, 1980.

[23] D. Nistér, O. Naroditsky, and J. Bergen, "Visual Odometry," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[24] A. Geiger, J. Ziegler, and C. Stiller, "StereoScan: Dense 3d reconstruction in real-time," in *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 963–968, 2011.

[25] C. Kerl, J. Sturm, and D. Cremers, "Robust Odometry Estimation for RGB-D Cameras," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3748–3754, 2013.

[26] Z. Fang and Y. Zhang, "Experimental evaluation of RGB-D visual odometry methods," *International Journal of Advanced Robotic Systems*, vol. 12, pp. 1–16, 2014.

[27] Z. Fang and S. Scherer, "Experimental study of odometry estimation methods using RGB-D cameras," in *IEEE International Conference on Intelligent Robots and Systems*, 2014.

[28] R. Mur-Artal, J. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, pp. 1147–1163, oct 2015.

[29] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, pp. 1255–1262, oct 2017.

[30] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *Springer Tracts in Advanced Robotics*, vol. 100, pp. 235–252, Springer Verlag, 2017.

[31] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR*, pp. 593–600, IEEE Comput. Soc. Press, 1994.

[32] D. Scaramuzza and F. Fraundorfer, "Visual Odometry: Part II: Matching, Robustness, Optimization, and Applications," *IEEE Robotics and Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[33] D. G. Low, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, pp. 91–110, 2004.

[34] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision (ICCV)*, 2011.

[35] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," in *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 674–679, 1981.

[36] C. Tomasi and T. Kanade, "Detection and Tracking of Point Features," tech. rep., Carnegie Mellon University, Pittsburgh, PA, 1991.

[37] J.-Y. Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm," tech. rep., Intel Corporation, 2000.

[38] D. Scaramuzza, F. Fraundorfer, and R. Siegwart, "Real-time monocular visual odometry for on-road vehicles with 1-point RANSAC," in *2009 IEEE International Conference on Robotics and Automation*, pp. 4293–4299, 2009.

[39] B. Kitt, A. Chambers, H. Lategahn, S. Singh, and C. Systems, "Monocular visual odometry using a planar road model to solve scale ambiguity," in *European Conference on Mobile Robots (ECMR)*, 2011.

[40] M. Li and A. I. Mourikis, "Improving the accuracy of EKF-based visual-inertial odometry," in *IEEE International Conference on Robotics and Automation*, pp. 828–835, 2012.

[41] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *The International Journal of Robotics Research*, vol. 32, pp. 690–711, may 2013.

[42] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *International Journal of Robotics Research*, vol. 34, pp. 314–334, mar 2015.

[43] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2012.

[44] G. Arbeiter, S. Fuchs, R. Bormann, J. Fischer, and A. Verl, "Evaluation of 3D feature descriptors for classification of surface geometries in point clouds," in *IEEE International Conference on Intelligent Robots and Systems*, 2012.

[45] A. Aldoma, Z. C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. Rusu, S. Gedikli, and M. Vincze, "Tutorial: Point cloud library: Three-dimensional object recognition and 6 DOF pose estimation," *IEEE Robotics and Automation Magazine*, 2012.

[46] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Robotics: Science and Systems Conference (RSS)*, 2014.

[47] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke, "Registration with the point cloud library: A modular framework for aligning in 3-D," in *IEEE Robotics and Automation Magazine*, 2015.

[48] K. Kim, N. Neretti, and N. Intrator, "Mosaicing of acoustic camera images," *IEE Proceedings: Radar, Sonar and Navigation*, vol. 152, no. 4, pp. 263–270, 2005.

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

[50] S. Negahdaripour, H. Assalih, Y. Petillot, and L. N. Brisson, "Performance and accuracy in visual motion computation from FS sonar video sequences," in *MTS/IEEE Seattle, OCEANS 2010*, 2010.

[51] S. Negahdaripour, M. D. Aykin, and S. Sinnarajah, "Dynamic scene analysis and mosaicing of benthic habitats by FS sonar imaging-Issues and complexities," in *OCEANS'11 - MTS/IEEE Kona, Program Book*, 2011.

[52] D. Ribas, P. Ridao, J. D. Tardós, and J. Neira, "Underwater SLAM in man-made structured environments," *Journal of Field Robotics*, 2008.

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

[54] F. Steinbrücker, "Real-Time Visual Odometry from Dense RGB-D Images," *IEEE International Conference on Computer Vision Workshops*, pp. 719–722, 2011.

[55] G. A. Jones, "Accurate and Computationally-inexpensive Recovery of Ego-Motion using Optical Flow and Range Flow with Extended Temporal Support," *Procedings of the British Machine Vision Conference*, pp. 75.1–75.11, 2013.

[56] H. Spies, B. Jähne, and J. L. Barron, "Range Flow Estimation," *Computer Vision and Image Understanding*, vol. 85, pp. 209–231, 2002.

[57] M. Jaimez and J. Gonzalez-Jimenez, "Fast Visual Odometry for 3-D Range Sensors," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 809–822, 2015.

[58] A. V. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," *Robotics: Science and Systems*, vol. 2, no. 4, 2009.

[59] P. Biber and W. Straaer, "The Normal Distributions Transform: A New Approach to Laser Scan Matching," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[60] M. Magnusson, *The Three-Dimensional Normal-Distributions Transform – an Efficient Representation for Registration, Surface Analysis, and Loop Detection.* PhD thesis, Örebro, 2009.

[61] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal, "Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations," *International Journal of Robotics Research*, 2012.

[62] T. Stoyanov, M. Magnusson, and A. J. Lilienthal, "Point set registration through minimization of the L 2 distance between 3D-NDT models," in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5196–5201, Institute of Electrical and Electronics Engineers Inc., 2012.

[63] M. Magnusson, A. Nüchter, C. Lörken, A. J. Lilienthal, and J. Hertzberg, "Evaluation of 3D registration reliability and speed-A comparison of ICP and NDT," in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3907–3912, 2009.

[64] A. Das and S. L. Waslander, "Scan registration with multi-scale k-means normal distributions transform," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[65] B. Jian and B. C. Vemuri, "A robust algorithm for point set registration using mixture of Gaussians," in *Proceedings of the IEEE International Conference on Computer Vision*, 2005.

[66] A. Burguera, Y. González, G. Oliver, A. Burguera, Y. González, ·. G. Oliver, and G. Oliver, "On the use of likelihood fields to perform sonar scan matching localization," *Autonomous Robots*, vol. 26, no. 4, pp. 203–222, 2009.

[67] B. Jian and B. C. Vemuri, "Robust point set registration using Gaussian mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

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

[69] M. D. Aykin and S. Negahdaripour, "On feature extraction and region matching for forward scan sonar imaging," in *OCEANS 2012 MTS/IEEE: Harnessing the Power of the Ocean*, 2012.

[70] A. Burguera, Y. González, and G. Oliver, "The UspIC: Performing scan matching localization using an Imaging Sonar," *Sensors (Switzerland)*, 2012.

[71] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. J. Leonard, "Advanced perception, navigation and planning for autonomous in-water ship hull inspection," *International Journal of Robotics Research*, vol. 31, pp. 1445–1464, oct 2012.

[72] M. Barjenbruch, D. Kellner, J. Klappstein, J. Dickmann, and K. Dietmayer, "Joint spatial- and Doppler-based ego-motion estimation for automotive radars," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2015.

[73] M. Rapp, M. Barjenbruch, K. Dietmayer, M. Hahn, and J. Dickmann, "A fast probabilistic ego-motion estimation framework for radar," in *2015 European Conference on Mobile Robots, ECMR 2015 - Proceedings*, 2015.

[74] M. Rapp, M. Barjenbruch, M. Hahn, J. Dickmann, and K. Dietmayer, "Probabilistic ego-motion estimation using multiple automotive radar sensors," *Robotics and Autonomous Systems*, 2016.

[75] N. Hurtós, X. Cufí, Y. Petillot, and J. Salvi, "Fourier-based registrations for two-dimensional forward-looking sonar image mosaicing," in *IEEE International Conference on Intelligent Robots and Systems*, pp. 5298–5305, 2012.

[76] N. Hurtós, S. Nagappa, N. Palomeras, and J. Salvi, "Real-time mosaicing with two-dimensional forward-looking sonar," in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 601–606, Institute of Electrical and Electronics Engineers Inc., sep 2014.

[77] N. Hurtós, X. Cufí, and J. Salvi, "Rotation estimation for two-dimensional forward-looking sonar mosaicing," in *Advances in Intelligent Systems and Computing*, vol. 252, pp. 69–84, Springer Verlag, 2014.

[78] N. Hurtós, D. Ribas, X. Cufí, Y. Petillot, and J. Salvi, "Fourier-based registration for robust forward-looking sonar mosaicing in low-visibility underwater environments," *Journal of Field Robotics*, vol. 32, pp. 123–151, jan 2015.

[79] D. Rudolph and T. A. Wilson, "Doppler Velocity Log theory and preliminary considerations for design and construction," *Proceedings of 2012 IEEE Southeastcon*, pp. 1–7, 2012.

[80] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous Ego-Motion Estimation using Doppler Radar," in *IEEE Annual Conference on Intelligent Transportation Systems*, 2013.

[81] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous Ego-Motion Estimation using Multiple Doppler Radars," in *IEEE International Conference on Robotics and Automation*, 2014.

[82] C. D. Monaco, S. N. Brennan, and K. A. Hacker, "Doppler Velocity Log Placement Effects on Autonomous Underwater Vehicle Navigation Accuracy," in *IEEE OES Autonomous Underwater Vehicle Symposium*, 2018.

[83] P. A. Miller, J. A. Farrell, Y. Zhao, and V. Djapic, "Autonomous underwater vehicle navigation," *IEEE Journal of Oceanic Engineering*, vol. 35, no. 3, pp. 663–678, 2010.

[84] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. Artech House, second ed., 2013.

[85] K. Tang, J. Wang, W. Li, and W. Wu, "A novel INS and doppler sensors calibration method for long range underwater vehicle navigation," *Sensors (Switzerland)*, vol. 13, no. 11, pp. 14583–14600, 2013.

[86] P. Liu, B. Wang, Z. Deng, and M. Fu, "INS/DVL/PS Tightly Coupled Underwater Navigation Method With Limited DVL Measurements," *IEEE Sensors Journal*, vol. 18, no. 7, pp. 2994–3002, 2018.

[87] A. Tal, I. Klein, and R. Katz, "Inertial navigation system/doppler velocity log (INS/DVL) fusion with partial dvl measurements," *Sensors (Switzerland)*, vol. 17, no. 2, pp. 1–20, 2017.

[88] S. Hong, M. H. Lee, H. H. Chun, S. H. Kwon, and J. L. Speyer, "Observability of error states in GPS/INS integration," *IEEE Transactions on Vehicular Technology*, vol. 54, no. 2, pp. 731–742, 2005.

[89] F. Guo, L. Xie, J. Chen, C. Song, S. Wang, and H. Yu, "Research of SINS / DVL / OD Integrated Navigation System based on Observability Analysis," in *Proceedings of the 35th Chinese Control Conference (CCC)*, 2016.

[90] A. Borko, I. Klein, and G. Even-Tzur, "Observability and Performance Analysis of Velocity Measurements with Lever Arm Aided INS," *Proceedings*, vol. 2, no. 3, p. 138, 2017.

[91] Honeywell Aerospace, "HG1930 Inertial Measurement Unit," tech. rep., Honeywell International Inc., Phoenix, AZ, 2016.

[92] Rowe Technologies, "SeaPilot DVL and OEM," tech. rep., Rowe Technologies Inc., 2018.

[93] C. D. Monaco and S. N. Brennan, "Ego-Motion Estimate Corruption Due to Violations of the Range Flow Constraint," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

[94] A. I. Comport, E. Malis, and P. Rives, "Accurate quadrifocal tracking for robust 3D visual odometry," *Proceedings - IEEE International Conference on Robotics and Automation*, no. April, pp. 40–45, 2007.

[95] K. Koeser, B. Bartczak, and R. Koch, "An Analysis-by-Synthesis Camera Tracking Approach Based on Free-Form Surfaces," in *Pattern Recognition: 29th DAGM Symposium, Heidelberg, Germany, September 12-14, 2007. Proceedings* (F. A. Hamprecht, C. Schnörr, and B. Jähne, eds.), pp. 122–131, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[96] A. Comport, E. Malis, and P. Rives, "Real-time Quadrifocal Visual Odometry," *The International Journal of Robotics Research*, vol. 29, no. 2-3, pp. 245–266, 2010.

[97] T. Tykkala, C. Audras, and A. I. Comport, "Direct Iterative Closest Point for Real-time Visual Odometry," *Int. Conf. on Computer Vision (ICCV)*, pp. 2050–2056, 2011.

[98] B. K. P. Horn, J. G. Harris, and G. Harris, "Rigid body motion from range image sequences," *CVGIP: Image Understanding*, vol. 53, no. 1, pp. 1–13, 1991.

[99] M. Yamamoto, P. Boulanger, J. A. Beraldin, and M. Rioux, "Direct Estimation of Range Flow on Deformable Shape from a Video Rate Range Camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 1, pp. 82–89, 1993.

[100] J. Gonzalez, "Recovering Motion Parameters from a 2D Range Image Sequence," *IEEE International Conference on Pattern Recognition*, no. 1, pp. 1–15, 1996.

[101] J. Gonzalez and R. Gutierrez, "Direct motion estimation from a range scan sequence," *Journal of Robotic Systems*, vol. 16, no. 2, pp. 73–80, 1999.

[102] M. Harville, A. Rahimi, T. Darrell, G. Gordon, and J. Woodfill, "3D Pose Tracking with Linear Depth and Brightness Constraints," in *International Conference on Computer Vision*, pp. 206– 213, 1999.

[103] M. Jaimez, J. G. Monroy, and J. Gonzalez-Jimenez, "Planar odometry from a radial laser scanner. A range flow-based approach," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2016-June, pp. 4479–4485, 2016.

[104] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 573–580, 2012.

[105] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High Accuracy Optical Flow Estimation Based on a Theory for Warping," in *European conference on computer vision*, vol. 3024, pp. 25–36, Springer Berlin Heidelberg, 2004.

[106] C. Eisler, "Near Mode: What it is (and isn't)," 2012.

[107] U. Franke, D. Pfeiffer, C. Rabe, C. Knoeppel, M. Enzweiler, F. Stein, and R. G. Herrtwich, "Making Bertha See," in *IEEE International Conference on Computer Vision (ICCV)*, 2013.

[108] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Springer International Publishing, 2 ed., 2016.

[109] H. Winner, S. Hakuli, F. Lotz, and C. Singer, *Handbook of Driver Assistance Systems*. Switzerland: Springer International Publishing, 2016.

[110] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry for ground vehicle applications," *Journal of Field Robotics*, 2006.

[111] J. Van Brummelen, M. O'Brien, D. Gruyer, and H. Najjaran, "Autonomous vehicle perception: The technology of today and tomorrow," *Transportation Research Part C: Emerging Technologies*, 2018.

[112] T. D. Gillespie, *Fundamentals of Vehicle Dynamics*. SAE International, 1992.

[113] A. S. Trigell, M. Rothhämel, J. Pauwelussen, and K. Kural, "Advanced vehicle dynamics of heavy trucks with the perspective of road safety," *Vehicle System Dynamics*, vol. 55, pp. 1572–1617, oct 2017.

[114] M. A. Richards, J. A. Scheer, and W. A. Holm, *Principles of Modern Radar: Radar Applications*. Scitech Publishing, 2014.

[115] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive Radars: A review of signal processing techniques," *IEEE Signal Processing Magazine*, 2017.

[116] L. Stanislas and T. Peynot, "Characterisation of the delphi electronically scanning radar for robotics applications," in *Australasian Conference on Robotics and Automation, ACRA*, 2015.

[117] AutonomouStuff, "Delphi Electronically Scanning Radar," tech. rep., AutonomouStuff, 2018.

[118] D. Kellner, M. Barjenbruch, K. Dietmayer, J. Klappstein, and J. Dickmann, "Instantaneous Lateral Velocity Estimation of a Vehicle using Doppler Radar," in *16th International Conference on Information Fusion*, 2013.

[119] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A General Framework for Graph Optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[120] I. Hemisphere GNSS, "A325 GNSS Smart Antenna," tech. rep., Hemisphere GNSS, Inc., 2015.

[121] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: An Open-Source Robot Operating System," in *ICRA workshop on open source software*, 2009.

[122] J. Zhang and S. Singh, "Visual-lidar Odometry and Mapping: Low-rift, Robust, and Fast Localization," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 393–398, 2011.

[123] A. Barth and U. Franke, "Estimating the driving state of oncoming vehicles from a moving platform using stereo vision," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 4, pp. 560–571, 2009.

[124] S. Hong, J. Kim, J. Pyo, and S. C. Yu, "A robust loop-closure method for visual SLAM in unstructured seafloor environments," *Autonomous Robots*, vol. 40, no. 6, pp. 1095–1109, 2016.

[125] M. Mostafa, S. Zahran, A. Moussa, N. El-Sheimy, and A. Sesay, "Radar and visual odometry integrated system aided navigation for UAVS in GNSS denied environment," *Sensors (Switzerland)*, vol. 18, no. 9, 2018.

[126] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[127] D. Nistér, "An Efficient Solution to the Five-Point Relative Pose Problem," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004.

[128] H. Stewénius, C. Engels, and D. Nistér, "Recent developments on direct relative orientation," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 60, no. 4, pp. 284–294, 2006.

[129] P. Corke, J. Lobo, and J. Dias, "An introduction to inertial and visual sensing," *International Journal of Robotics Research*, vol. 26, pp. 519–535, jun 2007.

[130] FLIR Integrated Imaging Solutions, "FLIR Blackfly USB3 Vision," tech. rep., FLIR Integrated Imaging Solutions, 2017.

[131] Analog Devices, "Ten Degrees of Freedom Inertial Sensor: ADIS16407," tech. rep., Analog Devices, Inc., 2011.

[132] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.

[133] T. E. Blanford, D. C. Brown, and R. J. Meyer, "Design considerations for a compact correlation velocity log," *Proceedings of Meetings on Acoustics*, vol. 33, pp. 1762–1762, apr 2018.

[134] L. Silveira, F. Guth, P. Drews, P. Ballester, M. Machado, F. Codevilla, N. Duarte-Filho, and S. Botelho, "An open-source bio-inspired solution to underwater SLAM," in *International Federation of Automatic Control (IFAC)*, 2015.

[135] D. C. Brown, S. F. Johnson, and D. R. Olson, "A point-based scattering model for the incoherent component of the scattered field," *The Journal of the Acoustical Society of America*, vol. 141, pp. EL210–EL215, mar 2017.

[136] S. F. Johnson and D. C. Brown, "SAS Simulations with Procedural Texture and the Point-based Sonar Scattering Model," in *OCEANS 2018 MTS/IEEE Charleston, OCEAN 2018*, 2018.

[137] APL-UW, "High-Frequency Ocean Environmental Acoustics Models Handbook," tech. rep., APL-UW, 1994.

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

[139] G. Csurka, C. Zeller, and O. D. Faugeras, "Characterizing the Uncertainty of the Fundamental Matrix," in *Computer Vision and Image Understanding*, 1996.

[140] Z. Zhang, "Determining the Epipolar Geometry and its Uncertainty: A Review," 1998.

Vita

Christopher D. Monaco

Christopher (Chris) D. Monaco received his B.S. and M.S. in Mechanical Engineering at The Pennsylvania State University in 2014 and 2016, respectively. His master's thesis focused on detecting the instability of oncoming vehicles using optical flow and map-based context. Chris previously served as the Controls Team Leader and Advanced Driver Assistance Systems (ADAS) Graduate Student Adviser for the Penn State Advanced Vehicle Team. More recently, Chris served as an Autonomous Driving Testing intern and an Autonomous Driving Localization intern at Mercedes-Benz Research and Development North America. During his graduate studies, he was a Graduate Research Assistant at the Penn State Applied Research Laboratory and the Intelligent Vehicles and Systems Group. Chris's research interests primarily revolve around using perception to aid the localization of autonomous vehicles.