VISION-BASED MAP-TO-IMAGE CORRESPONDENCE FOR
ATTITUDE ESTIMATION IN AUGMENTED REALITY

APPLICATIONS

A Dissertation in
Mechanical Engineering
by
Richard Patrick Proffitt

© 2016 Richard Patrick Proffitt

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2016
The dissertation of Richard Patrick Proffitt was reviewed and approved* by the following:

Sean Brennan  
Associate Professor of Mechanical and Nuclear Engineering  
Dissertation Advisor, Chair of Committee

Matthew Parkinson  
Associate Professor of Mechanical and Nuclear Engineering

Karl Reichard  
Research Associate, Applied Research Laboratory

Richard Tutwiler  
Senior Scientist, Professor of Acoustics

Karen Thole  
Department Head, Department of Mechanical and Nuclear Engineering

*Signatures are on file in the Graduate School.
Abstract

This dissertation proposes a methodology for attitude correction by matching two different representations of the same environment. This is accomplished by matching features extracted from an image of the driving environment (measurement) to a pre-defined expectation of how the environment should look (map). A mobile mapping system (MMS) is developed as a platform for collecting the data necessary to conduct these tests. Using sample images, features are investigated for properties that generalize easily between the virtual and physical worlds. This problem is formulated specifically for cases in which mountains are visible, allowing for the use of the horizon contour as a low-dimensional waveform in which to search for features. Once processes for achieving alignment using feature-based methods are developed, they are applied to data-sets consisting of images taken while the MSS moves through an environment. System performance is enhanced by the use of Kalman filtering to predict the dynamic motion of the system and mathematically account for it in estimation of attitude. The system developed in this dissertation is proven to work for orientation correction on a vehicle traveling approximately 35 mph in a rural environment.
# Table of Contents

List of Figures vii
List of Tables xi

Chapter 1

Introduction 1

1.1 Introduction ................................. 1
1.2 The Pose Estimation Problem ...................... 3
1.3 State of the art in vision-based orientation estimation ....... 8
    1.3.1 General implementations .................. 8
1.4 Current Opportunities for Improvement ................. 21
1.5 Statement of research .......................... 22
1.6 Overview of the Proposed System .................. 23
    1.6.1 Assumptions ............................. 25
1.7 Organization ................................. 26

Chapter 2

Description of the Test Platform 27

2.1 Introduction ................................ 27
2.2 The IVSG Mobile Mapping System .................. 28
    2.2.1 The Software ............................. 28
    2.2.2 Pose Sensors ............................. 29
    2.2.3 Imaging Sensors ........................... 30
    2.2.4 Laser Sensors ............................. 31
2.3 Validation ................................ 32
2.4 The Map .................................. 34
Chapter 3
Horizon Extraction 38
3.1 Introduction ................................................. 38
3.2 The Canny Edge ............................................ 38
   3.2.1 Naive Canny edge technique ......................... 40
   3.2.2 Down-sampling the edges with a Gaussian blur .... 41
   3.2.3 Image partitioning ..................................... 43
3.3 Results ................................................... 46
3.4 Conclusions .............................................. 49

Chapter 4
Decomposition of the horizon contour into feature sets 51
4.1 Introduction .............................................. 51
4.2 Feature extraction ........................................ 52
   4.2.1 Introduction/background .............................. 52
   4.2.2 Method selection and implementation ............... 56
4.3 Correspondence ........................................... 58
   4.3.1 Introduction .......................................... 58
   4.3.2 Implementation ....................................... 59
      4.3.2.1 Recursive Bayes Filter ......................... 59
      4.3.2.2 k-d tree Nearest Neighbors Search .......... 61
4.4 Conclusions .............................................. 62

Chapter 5
Relative Orientation Between Feature Sets 64
5.1 Introduction .............................................. 64
5.2 The affine transformation matrix ........................ 64
5.3 Linear least squares regression .......................... 67
5.4 Determination of relative orientation .................. 68
5.5 Conclusions .............................................. 70

Chapter 6
The Kalman filter 71
6.1 Introduction .............................................. 71
6.2 Kalman filter equations .................................. 72
6.3 The dynamic model ...................................... 73
6.4 The Extended Kalman Filter ............................. 74
      6.4.1 System design ..................................... 75
6.5 Conclusions .............................................. 77
Chapter 7
Results
7.1 The test track facility .............................................. 78
7.2 Stationary tests ....................................................... 78
  7.2.1 Sequence length requirement ................................. 81
7.3 Moving tests ......................................................... 82
  7.3.1 The data-sets ..................................................... 83
  7.3.2 Map-to-map correspondence ................................... 84
  7.3.3 Simulated IMU .................................................... 84
  7.3.4 Results ............................................................ 84
  7.3.5 Error analysis .................................................... 86
7.4 Conclusions ........................................................... 90

Chapter 8
Conclusions
8.1 Summary .............................................................. 93
8.2 Future work .......................................................... 95
8.3 Conclusions .......................................................... 96

Bibliography ............................................................. 98
List of Figures

1.1 Perhaps the most prolific application of augmented reality is in military aviation, where it has been used for targeting and navigation since WWI. Seen here is the state of the art heads-up-display (HUD) on an FA-18 fighter jet [1]. ........................................ 2

1.2 Concept art from Jaguar displaying their plan for vehicle based augmented reality with a product called the Virtual Windscreen, projected for release in 2024 [2]. ................................. 3

1.3 In this figure, the map of the lane markers is represented in red. In the near field the markers match extremely well, but as they extend away from the vehicle orientation errors become apparent as the map diverges from the road. ................................. 5

1.4 Comparison of two images taken from the same location with a large difference in yaw. While the traffic cone in the near-field moves almost entirely across the image, the mountains in the background are almost unchanged, demonstrating the hardiness of horizons as a function of local movement. ................................. 8

1.5 The results of Stein Medioni’s work [3]. This visually displays the ground truth compared to the localized position. No scale or units were available in the paper. ................................. 13

1.6 In this image, occlusions of the horizon line are circled in red. These occlusions can be caused by any unmapped features such as buildings, vehicles, signs, fences, trees, and even pedestrians. ................................. 14

1.7 In Figures 2 and 3 from [4] the process and results for finding the pose of a single object in a stack of objects is demonstrated. ....... 16

1.8 Figure 3 from Lobo and Dias [5] presents the concept of vanishing points and how their intersection forms the horizon line. ................................. 17

1.9 Figure 10 from Roberts [6] demonstrates the angular error of the system during an experiment with (black) and without (gray) horizon and landmark matching. ................................. 17
1.10 Figure 7 from [7] demonstrates the RSAGS algorithm at work. Four random points are chosen in each contour and the disparities determined. These are then used to determine a transformation between the two curves and the process is continued iteratively until a certain level of agreement is achieved. 

1.11 In Figure 17 from [7], results from a test at 30 mph demonstrate average tracking of -104° in roll, -0.00766° in pitch, and -0.0821° in yaw. These results were obtained in post-processing.

1.12 This flowchart represents the subsystems necessary to complete this project. The system will consist of a GPS/IMU sensor suite, a forward-facing camera, and a map. These will interact as shown to generate an estimation of vehicle pose.

2.1 The data collection vehicle, seen here, is equipped with a Honeywell INS, Novatel GPS, a forward facing camera, two downward facing cameras in the rear, and a downward facing LiDAR scanner. This has been used to collect roadway maps in and around State College.

2.2 The Occam Omni S stereo camera, showing a stitched panoramic image on the top and the disparity map on the bottom.

2.3 The Occam Stereo camera consists of ten cameras in two rows that combined cover a 360° horizontal field of view.

2.4 This figure represents the bandwidth taken up by each sensor. In total, the system uses between 5 and 7 GB/s.

2.5 Using differential corrections, the mean lane error is observed to be approximately 0.05 +/- 0.4 meters.

2.6 Without differential corrections, the standard deviation and mean offset rise significantly.

3.1 Results from Canny filtering as presented in [8].

3.2 The process for the Canny-edge waterfall technique.

3.3 This is the output image from raw Canny filtering. The extracted horizon line is displayed in blue.

3.4 The Canny waterfall method is an intuitive and effective approach to horizon extraction, but it is easily foiled by overhead occlusions such as this. This mis-identification could also be caused by stark clouds, etc. Another method must be identified for the system to be robust to environments like this.
3.5 This figure demonstrates the process for the Gaussian blur approach to horizon extraction. This method is effective in consistently identifying the horizon region, but provides a low resolution horizon line.  

3.6 This is the horizon line extracted using the Canny filter method after a Gaussian blur has been applied. Note the amount of noise in the resultant line. In blue the results from the original Canny waterfall technique are demonstrated to be insufficient, whereas the results from the Gaussian blur are the true horizon contour.  

3.7 An example of a difficult horizon extraction problem as the mountains fade into the mists in the distance. Here the difference in the color-space of the sky and ground is very small.  

3.8 This method utilizes the robustness of the Gaussian blur method while maintaining fidelity to the true horizon contour.  

3.9 Using the results from the Gaussian blur approach, here the sky/not-sky partitioning technique is demonstrated in comparison with the Gaussian blur. Notice how much cleaner the results are from the partitioning method.  

3.10 This image demonstrates the failure of the simple Canny-edge waterfall technique. In the presence of overhead obstructions like power lines, the waterfall method stops whereas the Gaussian blur is able to accurately identify the horizon line with noise and the sky/not-sky partitioning can do so with high accuracy.  

3.11 Because of the slight overcast on the day this image was taken, both the waterfall and blur methods in red and green are unable to distinguish an edge at the horizon towards the center of the image. In this case, the sky/not-sky partitioning in blue was critical in correctly identifying the regions, but even here the horizon line is not perfect throughout.  

3.12 This image most importantly demonstrates the failure of the image partitioning technique to distinguish the sky from the bright lane markers.  

4.1 This image demonstrates how features are defined in the contours. The two feature matrices \( F_g \) and \( F_b \) are defined by the vectors that connect local extrema, quantified by magnitude and orientation.  

5.1 Coordinate frame for the vehicle as defined by standards in vehicle coordinates.
5.2 This image demonstrates LLSR working in the presence of translation and rotation offset with Gaussian random noise.

5.3 A random contour was rotated through 360° and the angle of rotation extracted at each rotation. The results in terms of percent error as presented here demonstrate that the approximation of linearity works quite well, even in the presence of Gaussian random noise added to the contour.

5.4 This figure demonstrates the percent errors in partially overlapping contours.

7.1 The Larson Transportation Institute test track facility where testing was conducted. The straight portion of the track on the left side (facing north) provided the most clear view of the horizon and was the location of test runs.

7.2 A sample of the time-series images collected from a stationary position over 40 hours.

7.3 Results for relative roll in a 40 hour time series of images from a stationary platform observing unchanging terrain. As is evident, at night time readings are untrustworthy, however during the day results are reliable.

7.4 This figure highlights the three major environmental conditions observed during stationary testing: daylight, night-time, and fog.

7.5 Determination of the optimal sequence length \( k \). This was determined by averaging the percentage of correct matches for a map to 48 separate test images for a range of sequence lengths \( k \).

7.6 Samples of the images and extracted horizons for the moving test sets.

7.7 Noisy simulation of IMU measurements from INS ground truth.

7.8 The complete process as applied with example figures taken at each step.

7.9 Estimates of orientation from the three techniques described in this document: first order fitting, linear least squares regression (LLSR), and Kalman filtering.

7.10 Direct error in degrees for each of the methods.

7.11 Histograms of the errors presented in figure 7.10.

7.12 Errors are largely attributed to mis-identification of features.

8.1 An example of an environment in which multiple ridgelines can be used simultaneously to mitigate extraction errors and occlusions.
List of Tables

7.1  Error statistics for estimates of roll  . . . . . . . . . . . . . . . . . .  88
7.2  Error statistics for estimates of pitch  . . . . . . . . . . . . . . . . .  90
7.3  Error statistics for estimates of yaw  . . . . . . . . . . . . . . . . . .  90
Now this is not the end. It is not even the beginning of the end. 
But it is, perhaps, the end of the beginning.

Winston Churchill
1.1 Introduction

Human interaction with the world is enabled by perception, and modern technology has enabled new forms of perception by merging digital information with real-world views. This process of merging real and digital information is called augmented reality (AR). A commonly accepted definition of augmented reality was proposed by Harold Azuma in 2001, stating “an AR system supplements the real world with virtual (computer-generated) objects that appear to coexist in the same space as the real world” [9]. Augmented reality has the benefit of reducing operator distraction by minimizing cognitive distance and divided attention, two problems that arise from a human attempting to access information from multiple physically or cognitively separated sources simultaneously [10]. Without the ability to fuse these environments, even the most imaginative and groundbreaking virtual capabilities would be of little use to a human operator.

AR started with pilots as early as the First World War [11], and is now moving into commercial applications such as head-mounted or helmet-mounted systems [12]. Until recently, development has been primarily driven by military needs for situational awareness, with technologies focused on enabling pilots (Figure 1.1) and ground forces to operate more quickly, safely, and effectively with systems like heads-up-display targeting and identify friend or foe (IFF) capabilities. Recently, technological maturity has lead to these same AR capabilities being available commercially for civilian applications. Perhaps the most recent and popular contri-
Perhaps the most prolific application of augmented reality is in military aviation, where it has been used for targeting and navigation since WWI. Seen here is the state of the art heads-up-display (HUD) on an FA-18 fighter jet [1].

In the commercial field, the most notable example is 2013’s Google Glass [13], which displays information from a smartphone-like on-board computer on the lenses of a pair of glasses. Development of such technologies has continued with the forthcoming Oculus Rift [14], a headset that immerses users in a virtual environment and allows dynamic perception via inertial head tracking. The success of these systems indicates a real desire for AR systems in the commercial world.

Driving is a particularly high-impact application that requires proper perception, and thus would benefit from the development of novel perception technologies. Unlike man-portable systems like the above, a vehicle platform is distinguished in that it has a large infrastructure to support implemented technologies. From an equipment perspective this provides access to an existing power system and the freedom to use bulkier, heavier equipment without impeding the system’s motion. Mathematically, the system has that advantage that it is constrained to dynamic motion that is well studied and modelled. These factors make vehicles a viable and attractive platform for augmented reality development.
In-vehicle augmented-reality displays offer the potential for great safety and performance gains in enhancing a driver’s perception of the environment. The World Health Organization (WHO) has published that in two fifths of vehicle-on-pedestrian accidents, poor visibility on the part of the driver was a significant contributing factor [15]. The needs for this technology has recently been recognized by automotive manufacturers and development on such systems has recently begun [16]. Jaguar proposes integration of a virtual windshield starting in 2024 to eliminate blind spots [2]. General Motors has also proposed an enhanced vision system that will seek to enhance visibility in poor weather conditions [17]. Ideas to improve vehicle safety are focused on improving driver attentiveness and visibility, and AR is an obvious way to accomplish this.

1.2 The Pose Estimation Problem

Bridging the virtual and physical worlds requires that pose be known as accurately as possible to define how virtual objects correspond to physical objects. This correspondence process is called registration. Given that two data-sets are the same or nearly so, registration defines the transformation that must be applied to one set so that all features in it match perfectly with the same features in the other set. This is critical for augmented reality applications so that objects in the virtual world will be displayed in the correct location in the physical one. Likewise, finding one
data-set within another is called localization. Without accurate localization and registration, augmented reality systems cannot possibly succeed in their intended applications.

Pose consists of a three dimensional position vector and three orientation angles: roll, pitch, and yaw. In outdoor applications, the former is typically determined with global positioning systems (GPS), which is a direct measurement of position based on the time of flight of a signal from a satellite to the sensor. Orientation is measured indirectly by an inertial measurement unit (IMU) which measures three-axis accelerations and integrates to determine orientation. Inertial navigation systems (INS) combine accelerometers with gyroscopes that measure orientation directly. An INS is more accurate than an IMU, but also more expensive and is still subject to drift. Based on a mathematical understanding of vehicle motion, orientation and position can be linked through dynamic models. Sensor measurements and mathematical calculations are commonly combined using a Kalman filter [18]. A Kalman filter is a classic method for combining multiple streams representing the same information via a weighted average using dynamic trust metrics based on covariance. In the state of the art, GPS and INS are combined using a Kalman filter with a dynamic model of body motion to determine the six degree-of-freedom pose.

The correct determination of orientation is crucial for safety, automation, and augmented reality applications. Small perturbations in orientation locally result in ever increasing position error as range increases. For example, half a degree of error in yaw equates to half a meter of lateral error at a range of only 50 meters. At only a quarter of a mile this lateral error is four meters, or an entire lane width. This is especially problematic in augmented reality applications, where orientation errors can very quickly cause large mis-registration at relatively short ranges.

While sub-meter accuracy can be expected from even basic GPS systems, correct measurements of orientation are harder to achieve. Since orientation is not measured directly, any noise in these sensor streams is amplified during the integration process. Inertial navigation systems correct for this with sensor redundancy and mathematical modeling; however, even the most expensive solutions are unable to entirely eliminate the inherent drift. Thus, solving the orientation problem with sensors alone is a difficult and expensive task.
The accuracy of state-of-the-art pose estimation techniques is in general too loose for tight correspondence between the physical and mapped worlds, as can be seen in Figure 1.3. This figure represents the results of a system equipped with a defense-grade INS and GPS. The near-field correlation suggests good position localization; however, the far-field disparity is representative of poor orientation estimation that has accumulated and is manifested as lateral position error. Humans are able to see changes on the order of approximately 0.005° [19], [20]. For augmented reality on vehicles to be viable, orientation accuracy must approach these levels to mitigate or eliminate the visible disparity seen in figure 1.3.

A major contribution to the literature in this field is a survey by Dr. Gregory Welch of the University of North Carolina at Chapel Hill [21]. In this survey, Dr. Welch covers fifty years of augmented reality history and how the Kalman filter has been used during this time to determine the relationship between the virtual and physical worlds. While the origins of virtual reality (VR) are obscure, Welch traces them all the way back to early flight simulators going on digitally in the 1960s, and in analog systems all the way back to rudimentary pilot trainers developed in the
1930s. Early work, starting in earnest in 1966, was heavily motivated by the Air Force due to the applications in cockpits for pilot situational awareness and interaction with controls. Welch chose to begin with Ivan Sutherland’s 1963 Sketchpad which tracked a pen light for computer drawing. This is often cited as the start of VR. Sutherland said that the “ultimate display” for a computer was “a room within which the computer can control the existence of matter. [A] chair displayed in such a room would be good enough to sit in. Handcuffs displayed in such a room would be confining, and a bullet displayed in such a room would be fatal” [22]. Sutherland generated perhaps the first head-mounted virtual reality display, tethered to a large telescoping and rotating frame that tracked the user’s position by way of ultrasonic transducers [23]. This was one of the first implementations of virtual reality.

Summarizing types of errors, Welch says there are two kinds: dynamic and static. Static error, such as DC bias and sensor noise, is less nefarious as much work has been done on eliminating these. Dynamic errors are much more difficult to root out. The most common cause of dynamic error is latency. Welch continues to emphasize the importance of proper registration and low latency in augmented reality applications, going even so far as to place its priority above all others, even photo-realism. He quotes Sutherland again as quantifying the metrics for performances to be a pixel resolution of 1/100 of an inch and 1/10,000 of a degree of rotation. In a 1995 dissertation, Rich Holloway examined sources of error in augmented reality and found that mis-registration is primarily caused by errors in head tracking, and errors in head tracking are primarily caused by latency [24]. Assuming a head slew rate of 100 degrees per second and system latency of 100-200 ms, the system latency would cause 10°-20° of error. In Holloway’s paper, he determined that 1ms of latency corresponds to 1mm of mis-registration at arms length [24]. This problem has been perennial in VR applications. An early magnetic system, first developed for the military but made commercially available in 1979, was found to have up to 1/2 second latency [25] due to the high rate of movement of a human head, especially in flight. The Air Force found angular accelerations of the human head up to 6,000 degrees per second squared [26] in yaw (side-to-side). There clearly exists a need for an algorithmic correction to this sensor-based problem.
In early work (1980’s and 1990’s) the primary motion tracking system was one developed for the aforementioned government work, the Polyhemus 6 DOF motion tracker. During this time the major roadblock to development was latency in the Polyhemus system. The first de-classified use of a Kalman Filter in VR/AR applications was by Rebo in his 1984 dissertation on head tracking, in which he used the Kalman filter to predict motion and thus attempt to counteract latency. This worked only minimally. Over then next 10 years, his work evolved and became merely a way of gaining a better pose estimate of the user’s head. Soon the Polyhemus system fell out of favor due to it’s cost, size, weight, and latency problems. A major breakthrough was Azuma’s 1992 paper that used LEDs embedded in the ceiling as an additional way of measuring orientation [27]. This led to the now popular implementation of the Kalman filter as a method for fusing computer vision and inertial measurements. Later, Azuma’s 1994 paper [28] was an enhanced implementation of the above in which they used Kalman filters both to fuse optical and inertial measurements as well as to predict head motion to reduce latency. In this paper they found that using two Kalman filters for position and velocity, instead of one and differentiating, they enhanced their accuracy greatly. Since then, the primary thrust of Kalman filters in AR/VR applications has been to fuse different measurement sources for increased accuracy and predicted motion for decreased latency. Welch concludes that the “The “Holy Grail” for researchers working on tracking for VR/AR still seems to be robust and accurate tracking outdoors, for augmented reality everywhere” [21].

Many of these early implementations identified the value of visual measurements of orientation, but many struggled with identifying ‘markers’ to track visually. Without some repeatably identifiable object to track, orientation cannot be determined visually. In these implementations controlled environments were used with artificial fiducials. In a naturally-occurring environment, this problem becomes more difficult and naturally occurring features must be identified which are consistent enough that they can be used as fiducials.
Figure 1.4. Comparison of two images taken from the same location with a large difference in yaw. While the traffic cone in the near-field moves almost entirely across the image, the mountains in the background are almost unchanged, demonstrating the hardiness of horizons as a function of local movement.

1.3 State of the art in vision-based orientation estimation

1.3.1 General implementations

In the late 1990’s researchers in this field began to utilize the boundary between the sky and the ground (the horizon) as a feature for tracking. The horizon is a useful feature primarily because of its large size and far distance from the observer. It’s size makes it clearly visible over large areas and slowly changing. The distance of horizons from the observer also make them visually move more slowly relative to local movement (figure 1.4), and they are thus repeatable in the presence of local disturbances.

The use of horizon lines to correct for orientation has been studied for many years. The main body of literature in this area is focused primarily on applications where a clear view of the horizon is guaranteed such as in aerial and extra-terrestrial applications. This is because these situations are free of occlusions allowing for an unobstructed view of the horizon. From the sky (as in [29] and [30]) or on another planet (as in [31]) there is little to no likelihood that an unmapped feature will occlude the horizon. Contrarily, typical ground vehicles are subject to occlusions constantly, especially in urban environments. These papers are considered foundational in that they introduce the problem without occlusions or dynamics.

Though vision-based pose estimation has been studied for decades, the field
of research has recently been additionally motivated by the presence of similar systems in nature. Stone et al [32] present remarkably relevant work in their study of ant vision. Here, Stone et al concluded that desert ants see in extremely low resolution and monitor the horizon in ultra-violet. One hypothesis regarding how they navigate is that they use a ‘visual compass’ approach, as documented by Labrosse et al [33]. This theory states that ants store a memory of horizons they have seen and compare every new horizon contour to the memorized ones to determine heading. Stone et al recreated their hypothesis of ant-navigation by walking around Edinborough with a panoramic ultra-violet camera. They used a traversal from January as a map and a repeated traversal from April as a query set, and were able to identify repeatability between the two sets. Their study determined that tracking of UV horizon lines was sufficient to navigate an urban environment and confirmed their hypothesis about the possibility of ants using this same guidance technique. The bio-mimicry aspect of this method is a powerful motivator, reflecting positively on the overall strategy of the approach. This study further reinforces that even low resolution images can be used successfully for basic navigation needs.

Some of the most influential papers in this field address orientation localization in unmanned aerial vehicles (UAVs). In 2005 Woo et al notably introduced the concept of a ‘horixel’ as a pixel with a high probability of being a part of the horizon [29]. This was determined based on the strength of the edge it forms, if any, and how much it segments the image into unique intensity regions. This process was applied to infrared images instead of visible wavelength cameras to enhance robustness to changes in lighting conditions and noise in the measurements. Like [34], the horizon line was described as a series of intersecting curves and corners were detected as the location of direction changes. Curvature scale space (CSS) was used for peak identification on the extracted horizon lines. CSS improves the peak extraction methods robustness to scale, noise, and viewing angle as compared to the Woo’s previous attempts with the Second Degree of Guassian (SDG) method, which seeks to detect mountain peaks by describing all mountains as Gaussian distributions. The CSS was repeatable in 90% of horizon lines when rotated by 90° whereas the SDG methods were only repeatable in 67%. The primary contribution of this paper is its study of features that represent the horizon and their behavior
in the presence of noise and disparate viewing angles.

In [30] published in 2005, Bao proposed to extract the horizon line from the forward-facing camera on an micro aerial vehicle (MAV) even in the presence of adverse weather conditions such as fog. The horizon is extracted similarly to the method proposed in [29]. Edge detection was performed on the image, and edge strength was determined based on whether the edges correspond to the boundaries between intensity regions. This generated a gray scale image of edges. A linear filter was then passed over the edge image at various angles of orientation and all possible translations, and the orientation and translation with the strongest resultant edge was defined as the first order fit to the horizon. This method is called the orientation projection method. The orientation of this first order line was used as the aircraft’s roll angle by assuming that the camera was far enough away from the horizon that it could be generally assumed to be flat and perpendicular to the gravity vector. This method was fast and robust to noisy measurements and blurry images. These methods were tested on aerial video footage taken using the investigators’ MAV. No results are provided except that “For these data, the horizon is correctly identified in nearly 100% of cases” [30]. This work is important because it proposes a much lower dimensional problem than complete curve matching of complex horizon lines. It is distinct from the proposed work in that a ground vehicle will rarely if ever be so far from the horizon line that the horizon can be assumed to be flat and normal to the gravity vector. If applied in tandem with map-based methods this work could significantly lower the computational expense of the line matching problem.

In 2011, Baboud et al aimed to match static images of mountains with a DEM representation of the environment [35]. The authors assumed the availability of camera parameters and the position from which the picture was taken. They used edge extraction on the image, and then partitioned images into silhouette and non-silhouette lines (the horizon line being a silhouette line) based on how the lines cross each other. Once silhouette lines were identified, they were defined as a series of vectors and matched with the rendered horizon from a DEM. This was accomplished using the fast Fourier transform (FFT) formulated as a maximum likelihood estimation (MLE) problem. This was then used for augmented reality such as road overlay and annotation. The process was tested on 28 images ran-
domly chosen from the Internet and localization was achieved with an 86% success rate at a computational speed of 2 minutes per photograph. The accuracy of the orientation recovered is described as “generally quite accurate, i.e. below 0.2°s” [35]. This work is significant because it so closely matches the goals of the proposed work. It still differs from this work because it classifies edges into silhouette and non-silhouette based on the assumption of no occlusions. In a vehicle-based application, this is impractical and consideration must be given to utilize more robust feature sets. The matching algorithm presented is quite robust, though it is too slow to be practical in real-time or quick-time applications.

Some papers differ from the proposed work in that they utilize horizon matching methods for localization of position only ([3], [34], [31]). These papers can still provide valuable insights to the orientation localization problem. In general a horizon line is measured from the environment and compared to a lookup table of expected horizon lines from candidate positions. The two lines are compared and the best fit is identified as the position of the robot. In contrast, orientation is generally achieved using only two lines, one measured and one mapped, and the angle is determined from the transformation that makes them correspond (as detailed above in [29], [30], and [35]). Therefore the estimation of orientation requires an additional correspondence and matching step as compared to localization of position alone. Nonetheless, both position and orientation localization research are valuable in identifying horizon extraction and characterization methods as well as feature set extraction.

In a 1993 paper [3], Talluri and Aggarwal present one of the earliest published works showing horizon-based position localization on a robotic platform. The robot was equipped with a pan-tilt-zoom (PTZ) camera, a compass, and an altimeter. A panoramic horizon line was collected by taking pictures pointing north, south, east, and west. The horizon line contour (HLC) was extracted from these images using a gradient operator. The map of the robot’s environment was gridded and the north, south, east, and west HLCs were stored in a table. Once collected from the camera, the center point of each measured HLC was extracted and compared to those in the lookup table. This gave the authors a computationally efficient method to reduce the number of possible locations in the grid. Once a subset of potential grid locations was identified, the HLCs from the camera were compared
to those from every possible grid coordinate using linear least squares, and the best match was chosen as the location of the robot. This method was tested on one data set from the United States Geological Survey (USGS) from areas in Colorado and Texas. Localization was achieved to within 50 m. A second case was tested in which Gaussian random noise was introduced to both HLCs, and the authors described the matching performance as ‘still quite close’. While this work does not specifically address the problem of localizing orientation, it is important in its use of the horizon line and linear least squares regression to match lines. This paper demonstrates the feasibility of least squares for this purpose.

In [34], a 1995 paper by Stein and Medioni from the University of Southern California for the Advanced Research Projects Agency of the Department of Defense, proposes to solve the ‘kidnapped robot’ localization problem. This is a common classification of the localization problem in which the robot must localize itself in a map with no a priori information regarding pose. The panoramic horizon line was calculated mathematically from a topological map for locations along a grid and these horizons were stored in a table. The horizon line was then extracted from a panoramic image by partitioning the image into sky and ground pixels based on color space. The horizon lines were defined using four metrics describing the direction and length of the vectors connecting the points. The feature sets were then compared to determine a match. The cardinality of the mapped horizons was varied to account for resolution differences between the map and the camera image. Matches are determined by minimizing the weighted error between the four metrics. It was tested in simulation and in a canyon in Utah with no occlusions. This process took one minute to compute and no numerical results were discussed. Results were presented visually in Figure 10 of their paper (Figure 1.5). This paper is important to the proposed work because it describes in detail the process of defining a curved line as a series of features, called here ‘super segments’, and using feature matching to determine correlation. It differs significantly from the proposed work in that it seeks to solve the entire localization problem with no a priori information, whereas this proposed work will assume accurate position localization and rough inertial information.

Another early application of this work is seen in [31], a 1996 paper by Cozman from Carnegie Mellon University. Here Cozman proposed solving for the position of
The results of Stein Medioni’s work [3]. This visually displays the ground truth compared to the localized position. No scale or units were available in the paper.

Figure 1.5. The results of Stein Medioni’s work [3]. This visually displays the ground truth compared to the localized position. No scale or units were available in the paper.

a lunar rover using a forward-facing camera, an inertial sensor, and a topographic map. In the forward-facing camera, the authors took advantage of the starkness of space to determine the horizon by partitioning the image using color. An A* search algorithm was used to partition the horizon into separate mountains. Each mountain’s peak location was used as a feature and the angle between the center of the camera and each visible peak was used a feature set. Global orientation and position are then determined by searching the topographic map for locations where these features are realizable. Matches were determined by minimizing the difference between the tabulated and measured values for peak orientation. This work is important because it proposes a novel feature set to use and proves the viability of using features rather than contours for horizon localization. Even though it does not solve for orientation, the description of the horizon line is a novel one and the use of features significantly reduces computational load.

Occlusions are features in an image that block the features of interest, in this case the natural horizon line, as seen in Figure 1.6. For this study, these would be unmapped features such as buildings, other cars, poles, signs, etc. They are
problematic as they give rise to an inaccurate representation of the horizon and will corrupt the match between the map and the image. This section discusses those papers that address the problem of occlusions and strategies for mitigating their effect.

In [4], Wolfson introduced a novel algorithm to register 2-d lines even when one has distracting features encoded within it. Like [29] and [34], the primary discussion targets matching lines that partially match but diverge through significant portions. This method was structured as follows: noisy data was smoothed with filtering, the data was represented as a ‘characteristic series’ of real numbers called ‘shape signatures’, correspondence was determined by matching sub-series of the characteristic series, and finally relative orientation and translation was determined using linear least squares. The shape strings have the advantage of being rotationally and translationally robust, allowing for a high level of repeatability. This was tested by photographing a set of objects (two pliers and a pair of scissors, in one experiment) both separately and in a pile. The outline of these objects was extracted using computer vision, and Wolfson sought to find the pose of the objects in the pile by matching the outline of the pile with the outline of the individual objects (Figure 1.7). Figure 1.7 demonstrates that this process worked quite well. This work is significant because it demonstrates techniques for identifying
matching sub-series of data in a cluttered environment. This is very similar to the problem of identifying the natural horizon line in a cluttered horizon.

[5] by Lobo and Dias seeks to overcome the problem of occlusions by utilizing them in the horizon extraction process. Here image processing was used to find parallel lines which were then used to identify the location of vanishing points in an image. Vanishing points are the intersection of parallel lines in an image (Figure 1.8). By definition, vanishing points lie on the horizon. Thus, the horizon line is determined as the line that connects two or more vanishing points. The resulting horizon line, a first order line, can then be used as an external reference to correct the orientation from an IMU. Results focused on using these techniques for calibration of the camera’s focal point, in which it was able to satisfactorily match state of the art techniques with tighter variance. This work could be significant as a means to determine the horizon line in cluttered urban environments. The work proposed in this dissertation would alter this technique with the use of maps. Comparing a first order fit of the mapped horizon line to the horizon line determined by the vanishing points in a cluttered environment could provide a good orientation correction when no view of the horizon is available.

Perhaps the most relevant work to this dissertation has been presented by Roberts et al in papers detailing the design and development of a soldier-worn augmented reality system for DARPA’s ULTRA-Vis program (Urban Leader Tactical, Response, Awareness, and Visualization) ([12], [6]). This was developed through a collaboration with Applied Research Associates, BAE Systems, and UNC Chapel Hill. This program aimed to give soldiers the capabilities necessary to receive AR information such as target highlighting while maintaining ‘finger-on-the-trigger’ capability. Their paper concerns itself with the full localization problem using low cost GPS/IMU fused with three types of visual orientation corrections in an extended Kalman filter (EKF). These three visual orientation corrections come from: landmark matching (LM), in which a user selects a known landmark in the visual image which is then used to determine relative pose; horizon matching (HM) in which the perceived horizon line is compared to the map’s predicted one; and sun matching (SM) in which the sun’s position and the time of day is used for orientation correction. In the horizon matching problem the perceived horizon was extracted by applying a Gaussian blur to the image and using a Sobel filter to find
Figure 1.7. In Figures 2 and 3 from [4] the process and results for finding the pose of a single object in a stack of objects is demonstrated.
Figure 1.8. Figure 3 from Lobo and Dias [5] presents the concept of vanishing points and how their intersection forms the horizon line.

Figure 1.9. Figure 10 from Roberts [6] demonstrates the angular error of the system during an experiment with (black) and without (gray) horizon and landmark matching. The horizon. The map’s horizon line was generated for a small search space around the sensor orientation and the best fit selected based on overlap with the perceived horizon line. Applying this method to real data from the Smokey and Rocky mountains an orientation correction of 4.3 +/- 2.6 mrad (or 246°s +/- 15°s) was obtained at 20 Hz (Figure 1.9). While these methods are demonstrated to work in the presence of some occlusions, the paper does not assert that it will work reliably in the presence of occlusions. The paper also emphasizes that the system does not work for fast slew rates, thus it would not be immediately applicable to a vehicle based implementation.

In their final paper on the project [36], Roberts et al present the final system
capable of 25 mrad (1.5°) accuracy without visual measurements and an accuracy of 10 mrad (.5°) with vision-based estimated of orientation. The authors attempted to implement constant visual updates; however, this was determined to be impractical because of computational burden and the inconsistent availability of visible horizons.

Similar work has been pursued at the Pennsylvania State University by Gupta and Brennan [7]. This work augments inertial measurements with corrections derived from matching of horizon lines extracted from a forward-facing camera. Here a clear view of the horizon was assumed throughout. Horizon lines were extracted using edge detection on the image and the horizon line was assumed to be the topmost line opposite the direction of gravity (the ‘waterfall’ technique). The mapped horizon line was determined by plotting USGS DEM data from the current pose estimate using the camera projection model. Gupta and Brennan in [7] use the random sample grid search (RSAGS) algorithm to simultaneously achieve correspondence and registration. RSAGS is an iterative algorithm that will match two data sets to one another without a previous assumption of correspondence. Four points are chosen randomly from one data set and compared to the correspondingly indexed points in the other set (Figure 1.10). Using these four disparities a rotation and translation is determined and the process repeated until the lines converge. This method is only applicable with the assumption that there is little difference between the two lines.

Results showed accuracy of 0.5°, 0.25°, and 0.8° in roll, pitch, and yaw respectively (the results of one experiment are shown in Figure 1.11). Open areas for future work include real-time implementation, a horizon line extraction method that is more robust to occlusions, and the addition of near-field features for robustness to stark or occluded horizon lines. The work proposed in this dissertation seeks to build from [7] and improve it’s robustness to occlusions and implement the work in real-time.

In 2012 Zhu et al published a conference paper that proposed solving the orientation problem using skyline matching between a physical image and a GIS map [37]. A panoramic horizon line was generated from the current pose. The horizon was extracted from the physical image using the classic ‘waterfall’ technique. The SAD algorithm was used to search for the location of best correspondence between
Figure 1.10. Figure 7 from [7] demonstrates the RSAGS algorithm at work. Four random points are chosen in each contour and the disparities determined. These are then used to determine a transformation between the two curves and the process is continued iteratively until a certain level of agreement is achieved.

The extracted horizon line and the panoramic horizon line. Using knowledge about the panoramic horizon line’s azimuth, they extrapolated an orientation differential and used it to correct orientation. Note that this method will only recover yaw if the camera is oriented with zero pitch and roll relative to gravity. This work was tested on static images, first by comparing artificial GIS generated partial horizons to the full panoramic horizon line also generated with GIS data, then with static images taken in the city center of Rennes, France. Both results proved ‘promising’ but also demonstrated failure in the presence of occlusions. The authors extended this work in 2013 [38], which adds skyline rectification and vanishing point detection to extend the approach to solve for full orientation. By extrapolating the
Figure 1.11. In Figure 17 from [7], results from a test at 30 mph demonstrate average tracking of -.104°s in roll, -.00766°s in pitch, and -.0821°s in yaw. These results were obtained in post-processing.

vertical vanishing point, the roll and pitch were solved for using projective geometry. Once roll and pitch were determined, the line segment was rectified to zero roll and zero pitch so that the methods proposed in [37] apply. Testing was primarily conducted on a series of 1419 synthetic images simulating a drive through GIS data. Computation took an average of 671 ms per image. Preliminary results for real data are mentioned but not discussed in depth, with the authors noting
that occlusions and overhead obstructions inhibit performance but results are still ‘quite good’. The authors ignore errors in GPS and assume a priori localization to be adequate to eliminate translational errors. Neither [37] or [38] address the problems associated with vehicle speed, which will be a critical challenge and is one of the primary capability enhancement proposed by this work.

Nghia in [39] approaches the problem of position localization on a highway using horizon comparison. This work utilized the mean-sum-of-absolute differences (MSAD) algorithm for correspondence and registration, which is a variation of the SAD algorithm used in [37].

1.4 Current Opportunities for Improvement

In recent years, the falling costs of sensors and computational power enables the use of advanced sensing systems and maps to aid perception. Sensors commonly found on modern vehicles include positioning sensors such as global positioning systems (GPS) and inertial measurement units (IMU), and environmental perception sensors such as cameras, RADAR (Radio Detection and Ranging), and LiDAR (Light Detection and Ranging). Sensing alone has its limits and may not work reliably in the presence of adverse conditions such as poor weather, occlusions, or sensor faults. In these situations performance can be enhanced through the use of maps that predefine the locations of permanent features, reducing the load on sensors. Maps have shortcomings as well: they are limited by their inability to account for transient features such as other vehicles, pedestrians, or new construction. The current gap in technology is the methodology for combining data feeds (cameras) with databases (maps).

There are opportunities to use AR to enable drivers to see what humans cannot see even in ideal conditions. With this technology, it is possible to see the mapped environment through mountains, vegetation, and buildings to enhance perception. Road maps can be highlighted to visually provide drivers with directions and information about the driving environment. Non-visual information can also be presented, such as the friction conditions on the road displayed as a colored heat map. The benefits of augmented reality integration with a vehicle platform is nearly limitless, providing many opportunities for ingenuity and novel interaction
with the driver.

1.5 Statement of research

This work proposes a new method for orientation correction by comparing an image from an observer’s physical location (collected by a forward-facing camera) to a rendering from the observer’s estimated pose in a virtual environment (a map). This results in a mathematical transformation from the virtual to the physical world. That transformation will be obtained by extracting and comparing the horizon line from each image. In this work, the horizon line is defined as the boundary between the sky and mappable features in the environment. Correspondence will then be determined by matching the 2-d horizon lines and determining relative orientation between the map and the image. This will be used to augment inertial measurements with an external reference to correct for drift and measurement inaccuracies. The goal will be to correct orientation sufficiently to provide satisfactory registration between the mapped and physical worlds to allow for an accurate augmented reality overlay even in the far-field. Additional details of the specific algorithm-level goals are provided in later chapters.

The goal of this work will be to present a visual information overlay to a driver with a focus on minimizing disparity, ideally approaching a resolution on the order of human visual acuity. A specific challenge of this work (versus traditional augmented reality applications) is that the system must work even when subjected to the large dynamic motion common to a vehicle. This will lead to challenges with time lag in the overlay, in which speed and disparity are directly linked. The final goal of this research is the implementation of an orientation correction system both statically and dynamically.

This method is unique in that it incorporates vision-based estimations of orientation as an external reference to mitigate drift, while also using a sophisticated understanding of vehicle dynamics in a Kalman filter for good tracking at high speed. Promising results with vision-based horizon line orientation matching have been achieved in state of the art literature [7], [12], [37]. Implementations of AR in the literature focus on occlusion-free scenarios such as aerial or extraterrestrial environments, or only perform statically or at very low speeds on the ground. This
Figure 1.12. This flowchart represents the subsystems necessary to complete this project. The system will consist of a GPS/IMU sensor suite, a forward-facing camera, and a map. These will interact as shown to generate an estimation of vehicle pose.

Work will seek to improve upon these results by generalizing test cases to less idealized scenarios and will seek to provide a real-time capability even under the fast dynamics of vehicular motion.

1.6 Overview of the Proposed System

The challenge of the proposed work is determining correspondence between the physical world and a virtual representation of the same. This study seeks to solve this problem by extracting what is assumed to be a universal, easily detectable feature: the horizon line, and determining relative orientation by comparing this measurement to a map of what the horizon is expected to look like given an estimate of pose. The output of this system is a corrected estimated of 3 DoF orientation as augmented by visual measurements. A flowchart representing this process is presented in Figure 1.12.

This system first assumes the existence of a map. This map can come from several different sources or even a combination of sources. The only requirement is that it represents the visible horizon from a given position. This map will be considered ground truth throughout the process.

The process begins with data acquisition. Data inputs consist at a minimum of position (latitude, longitude, and elevation) and an image. The image must be guaranteed to have a suitable overlap with the map (suitability defined as sufficient to provide enough features for a unique match). If the image is panoramic, then the resulting horizon contour will incorporate the visible horizon for all possible
yaw angles and thus no additional information is necessary. If not, an estimate of yaw must also be incorporated in the measurement. For the purpose of this work, the measurement will consist of a monocular camera image and a full 6 DoF pose that is assumed to be corrupted.

Once the measurement is collected, the horizon contour is extracted from the images. This is a series of x and y pixel coordinates corresponding to the boundary between the sky and the horizon. The exact definition of this boundary is dependent upon the map. If the map incorporates man-made features and vegetation, then the horizon extraction performed here can also include those features. Conversely, limitations on horizon extraction methods will dictate what must be included in the map. While some man-made features and occlusions can be ignored in the horizon extraction process, things such as vegetation cannot be easily ignored while remaining in the visible spectrum. Thus development of the horizon extraction process will help dictate the contents of the map.

In the next step, this horizon contour is quantified in terms of repeatable features. These features must be reasonably robust in the presence of noise and rotation variations. Subsequent sections will place some constraints on these requirements.

Once the horizon is represented in terms of features, correspondence can be done. Correspondence identifies features that are alike in both the query and map set so they can be compared ‘apples to apples’.

With correspondence, relative orientation is determined in terms of roll, pitch, and yaw. Pitch and yaw are quantified in terms of translational difference in the two horizons, while roll is determined by the amount the query image must be ‘twisted’ around the camera axis to achieve a match. With the orientation of the map image known, the orientation of the query can be determined with the newly found relative angles.

Once the orientation of the query image is determined, the results are processed with a Kalman filter that combines two measurements - one from the IMU and the other as determined visually using the process outlined above - with a dynamic model of motion. The dynamic model of motion with mathematically predict the movement of the vehicle and provide a third estimate of orientation. The Kalman filter will output the final result: an updated best fused estimate of vehicle
orientation that combines both visual and inertial measurements.

1.6.1 Assumptions

This dissertation work requires many assumptions, some of which are limiting but many of which are not. The primary assumptions are listed below:

1. In all test cases the horizon will be assumed to exist and be visible. Present work utilizes only cameras operating in the visible spectrum and no other hyper-spectral sensors will be used, therefore no form of occlusion penetration can be studied. Any work involving such sensors will be above and beyond the scope of this dissertation and will fall into the category of future work.

2. This work considers only rural landscapes where far-field horizon features are attainable. While this system would ideally work in an urban environment, preliminary work will focus on a rural testing environment where occlusions of the horizon line are minimized. Rural in this context implies that there consistently exists a large observable horizon line formed by the natural earth and the sky with only a few occlusions (unmapped features). Occlusions that will be tolerated are those such as other vehicles, signs, trees/natural objects, and power lines. Those that will not be considered are large buildings or groups of buildings or other large objects in the near-field. The exact nature of the features that will be used to characterize the horizon will be discussed in depth in later sections; however, the underlying assumption is that there exists some boundary between the sky and the ground that can be visually determined and this boundary consists largely of unchanging objects that can be repeatably and reliably found. The use of near-field features will be discussed but not implemented. The use of alternative features for when a horizon contour does not exist within the field of view of the camera will also be discussed but not implemented.

3. Position localization will be considered to be sufficiently accurate that errors in position are negligible.

4. Scale variance between the map and query image will be ignored. This follows from the previous assumption because scale variance would be caused by
two things: error in position localization (assumed above to be negligible), and differences in the camera’s focal length. It will be assumed that the focal length of cameras is known, and thus a static transformation from one camera image to another can be mathematically determined, rectifying any differences in zoom.

1.7 Organization

This document will be organized as follows: chapter 2 consists of an overview of the hardware capabilities required for and used in the execution of this project. Chapter 3 discusses the image processing and horizon extraction problem. Chapter 4 reviews the state of the art in feature extraction and correspondence and details the implementation and selection of strategies for that. Chapters 5 and 6 finalizes the work by explaining how an updated estimate of orientation is ultimately determined. Chapter 7 discusses test scenarios and results, and chapter 8 concludes the dissertation with a discussion of performance results and future work.
Chapter 2

Description of the Test Platform

2.1 Introduction

The research goal of this dissertation necessitates mobile mapping capability to observe the environment of interest, in this case roadways and their surroundings. Systems that provide this capability are called mobile mapping systems (MSS). In research communities, a large number of MSSs exist for a variety of applications including airborne mapping, ground robotics, and vehicles. MSSs often are capable of collecting all the information necessary for robotic vision, therefore development of MSSs are coupled to the development of autonomous vehicles. For efficiency and ease, most state-of-the-art autonomous vehicles are also MSSs. Major contributors in the field of autonomous vehicles, and thus roadway MSSs, include Carnegie Mellon University, Stanford, University of Michigan, and Google Mapping.

A major source of development for roadway based MSS was the DARPA Urban Grand Challenge in 2007 [40]. The challenge was to design and build an autonomous vehicle that was capable of navigating numerous challenging obstacles such as traffic, merging, and parking. A total of 89 teams originally applied, of whom 35 were selected to test at the qualifying event. Of those, 11 were graduated to the semi-finals, of whom only 6 successfully completed the final course. Of the teams that participating, those that have remained active in autonomous vehicle research are Carnegie Mellon and Stanford [41] [42] [43].
Figure 2.1. The data collection vehicle, seen here, is equipped with a Honeywell INS, Novatel GPS, a forward facing camera, two downward facing cameras in the rear, and a downward facing LiDAR scanner. This has been used to collect roadway maps in and around State College.

2.2 The IVSG Mobile Mapping System

To accomplish the goals of the research described above, Penn State IVSG has developed a mobile mapping system (MSS) that enables roadway mapping via a panoramic camera, three monocular cameras, and a defense-grade inertial navigation system (INS) (figure 2.1). Additionally, two light detection and ranging (LiDAR) sensors are included in the system.

The remainder of this chapter will discuss the details of the MSS capabilities, detail testing and validation results, and discuss the formation of a map database for use in the rest of this work.

2.2.1 The Software

The software architecture for the IVSG MMS is based around Ubuntu and the Robotic Operating System (ROS). ROS is a system of libraries and software tools that allow for the control and monitoring of multiple sensors on a common framework. For each sensor in the system there is a Linux/ROS driver which reads raw data from the input port and advertises a ROS topic. This topic holds all of the
data in a convenient structure and provides additional information, such as the arrival time of the data and various sensor parameters. Individual processes are called nodes, which are written in either Python or C++. Nodes can subscribe to topics, perform operations, and publish new or updated topics. ROS has additional utility for manipulating, monitoring, visualizing, and recording data. ROS topics are recorded using bagfiles, a custom file system that captures all of the information from the advertised topics. Bagfiles can be parsed into text files for post-processing. Real-time processes were implemented in ROS using nodes coded in Python and C++. All post-processing has been done in MATLAB.

2.2.2 Pose Sensors

Pose collection in the mapping system is achieved using a Novatel Synchronized Position Attitude Navigation (SPAN) DL-4 Plus INS with a Honeywell HG1700 inertial measurement unit (IMU). This is a tactical grade sensor system that can achieve position accuracy of 2 mm (with DGPS) and orientation bias of $1^{\circ}/\text{hr}$ and an angle of random walk $\sqrt{125^{\circ}/\sqrt{\text{hr}}}$. The Honeywell IMU collects inertial measurements using a ring-laser-gyroscope. The Honeywell is also equipped with resonating beam accelerometers that measure directional acceleration. Using double integration, these accelerations give relative position, which can be used to augment GPS position measurements obtained from the Novatel. If GPS measurements are unavailable - which is common in many driving scenarios - then the accelerometers can be integrated to solely produce a position estimate. This method of position estimation is called ‘dead-reckoning’ because it measures relative movement from a starting position instead of absolute movement. Because of this, it is considered to be inaccurate over long distances, but is very useful over short ranges. The Novatel system uses a factory-designed Kalman filter to integrate accelerometer measurements, spin-rate measurements, and GPS to give a more accurate measurement of position.

At the LTI test track facility, the GPS measurements are further augmented by a differential GPS (DGPS) base station. The base station increases the accuracy of GPS by correcting for unmodeled disturbances in the satellite signal, primarily those accumulated as the signals pass through the ionosphere. The station is
statically mounted at a previously surveyed location and monitors the difference between the pseudo-range reported by the sensor and the actual pseudo-range that yields the correct position. Given that the disturbances responsible for the disparity are quasi-static, the base station broadcasts a correction factor that is used to update all GPS signals in the area. These corrections are generally considered to be accurate to 1 cm RMS error, with an additional 1 cm per 10 km of distance when roving GPS measurements are corrected far from the base station. In this manner GPS signals can be increased in accuracy by an order of magnitude from meters to centimeters in real time. The authors have implemented this in a centralized location at the LTI test track facility and the correction is broadcast via radio. Due to terrain blocking signal reception even short distances from the test track, corrections can only be received at the test track facility itself. Use of differential corrections in tests will be clearly noted throughout this document.

2.2.3 Imaging Sensors

Three monocular and one stereo camera are included in the sensor suite. The three monocular cameras are Point Grey Research (PGR) Flea3 GigE cameras that communicate via Ethernet. The PGR cameras are in RGB color format, have a resolution of 768 x 1024, and are collected at 25 Hz. Two of the cameras are co-located with the Sick LiDAR pointing at the ground with slightly overlapped fields of view. Together, the two downward facing cameras have a lateral field of view of approximately 13.5 feet, which is enough to see both lane lines in a standard US 12 foot highway lane. The third PGR is positioned on the windshield just below the rearview mirror looking forward out the windshield. This camera collects imagery that sees approximately what the driver sees.

The stereo camera is an Occam Omni S (figure 2.3). The Omni S consists of two rows of 5 cameras positioned in a 360° circle. In total, the system has a horizontal field of view of 360° and a vertical field of view of 72°. The Omni S outputs the raw image from each individual camera, a stitched image from each row, and a disparity map generated from the stereo vision. Stitched images are collected at a rate of 30 Hz. The Omni S is mounted 96 inches off of the ground on the front of the vehicle.
To decrease the bandwidth demands on the master computer, each camera is paired with a MinnowBoard Max embedded computer. The MinnowBoards take the images in via Ethernet, compress them, and transmit them to the master computer via a USB 3.0 to Ethernet converter. The MinnowBoards achieve X compression ratio in the PGRs and Y compression ratio in the Occam.

2.2.4 Laser Sensors

Light detection and ranging (LiDAR) sensors collect three dimensional range information about an environment by emitting a laser and detecting its intensity upon reflection. Range information is extrapolated by measuring the time of flight of the laser from emittance to reflection back to the source. The IVSG MMS consists of two LiDAR sensors. The first is a Sick LMS 500 single scan LiDAR consisting of one laser scanning in one plane which generates a 2-d scan line, with a range accuracy of 24 mm. This LiDAR is positioned downward facing just off the rear bumper of the vehicle so it collects scan lines of the road surface. These scan lines are ‘stacked’ as the vehicle moves, generating a 3-d point cloud of the roadway. This sensor has a field of view of 190°, an angular resolution of .16667°, with a range of 65 m at a frequency of 25 Hz. It is mounted approximately 2 m off the ground giving a spatial resolution of approximately 1 cm. At this height, scan lines represent approximately 4 m long slices of the road profile, collected perpendicular to the direction of travel. Communication is done via Ethernet.

A Velodyne HDL32E is used for global point cloud collection. It consists of 32...
Figure 2.3. The Occam Stereo camera consists of ten cameras in two rows that combined cover a 360° horizontal field of view.

individual lasers that each collects at 10 Hz in a 360° field of view. Each of the Velodyne’s lasers has approximately 1.3° angular resolution and a range of 70 m. The sensor is mounted just above the Sick and is angled relative to the ground so that all 32 lasers are guaranteed to reflect off the ground at some point, given normal terrain.

2.3 Validation

Localization accuracy was determined by measuring the lateral position of a lane marker relative to the vehicle with a LiDAR. This was done to ensure that posi-
Figure 2.4. This figure represents the bandwidth taken up by each sensor. In total, the system uses between 5 and 7 GB/s.

Measurements were made relative to a stationary, global feature to increase accuracy. With DGPS active, the mapping vehicle has been demonstrated to be capable of position localization to within a few centimeters on a static platform. Driving approximately 50 mph decreases accuracy to 50cm (Figure 2.5). This disparity between the static and dynamic platform can be narrowed by improving the timing synchronization of sensors on the mapping vehicle. Without DGPS, this error goes up to approximately 2m when traveling 50 mph (Figure 2.6). Hardware modification are currently being pursued that can reduce error in the localization of position.
Figure 2.5. Using differential corrections, the mean lane error is observed to be approximately .05 +/- .4 meters.

2.4 The Map

This work necessitates the existence of a priori measurements of the environment. This set of data will be called the ‘map’. The map must consist of 1) accurate measurements of pose including three DOF orientation and three DOF position and 2) the horizon contour visible at that pose. A key benefit of this work is that there are many data sources that fulfill these requirements, and many methods exist for processing the data.

There are a number of state-of-the-art topographic maps that cover all or most of the United States. Commercial solutions such as Google maps are openly available to the public and are widely used for roadway navigation. Ubukawa characterizes the accuracy of these maps in his 2013 paper [44], in which he determines a root-mean-square error (RMSE) of 8.2m for Google maps and 7.9m for Bing maps. Google earth, collected with satellite imagery, is resolved to a grid approximately 15m square and has geo-location disparity of 50m. These localization accuracies are adequate for navigation; however, for roadway autonomy accuracy must be
Figure 2.6. Without differential corrections, the standard deviation and mean offset rise significantly.

several orders of magnitude lower (1 - 5cm) for a vehicle to successfully maintain a lane. Maps available from the government perform similarly. In 2000, NASA collected a digital elevation model (DEM) of the entire world using satellites in a program called the Shuttle Radar Topography Mission (SRTM). On September 23, 2014 the White House announced that it would begin releasing this data to the public. As of the writing of this document, all the data for North America is available to the public online [45]. In a study by Rodriguez, the 90% geo-location error for the SRTM data set was shown to be 12.6m. SRTM data is available in two resolutions, one sampled with a 90m grid and another at 30m.

These solutions are accurate enough for route planning and general localization, but vehicle automation work requires much finer position certainty. The benefit of the proposed work is that far-field features move very slowly relative to local translation, and thus far field localization accuracy does not need to be as tight as it would have to be if near-field features were used. Another problem with these lower resolution, global mapping solutions is that the data collected by them
has been smoothed in post-processing with a wide Gaussian filter. This ignores permanent man-made features such as buildings that could be useful to achieve an accurate match and replaces them with what appear to be hills or ‘lumps’ in the data.

Once the map database is established, the horizons must be extracted and stored as features. A variety of methods for this can be found in the literature. The first and most common method is to calculated mathematically what can be seen from a given pose using the pinhole projection method [12] and [7]. This method requires that data be processed and pinhole projection be computed for every query pose. To avoid this stringent feature extraction on the fly, an alternative method is to generate a horizon a priori and store it for every position in a grid, as used by [3], [34], [31], and [6]. This method reduces the computational load on the system but also increases storage demands and is crude if the grid is too large. A final method is to perform feature extraction on a visual representation of the data from the point of interest, as used in [37], [35], and [32]. This can be done either by ‘rendering’ the environment from the query pose or by using collected images as the map and directly performing horizon extraction.

These methods are expensive computationally. The pinhole projection method requires frequent mathematical computation of all points to determine visibility and perceived elevation based on visibility. The table look-up method requires the storage and rapid access of a large number of horizon contours, making it potentially restrictive in terms of data storage. Generating the horizon line from a rendered scene requires rendering of the virtual world, which also has a large computational overhead.

One of the primary challenges of this work is the presence of occlusions and unmapped features. As discussed above, commonly available maps include very good geographic data; however, cannot accurately represent the details or man-made features in an actual environment. Since unmapped features will be seen in the data as occlusions that will degrade the performance of the system, careful consideration was given to selecting a map database that would reflect as much of the actual, repeatable environment as was possible. The data also had to be stored in a quickly and easily accessible structure.

For these reasons, the authors have created a custom map database collected
by the system described above. This database consists of the full 6 DoF pose and camera images, either panoramic or monocular images pointing forward along the vehicle’s direction of travel. Maps are stored as 1-d contours representing the horizon. Horizon contours are stored in a grid of latitude and longitude coordinates. In the case of monocular camera data, the grid also contains yaw data. This prevents a query from being unable to obtain a match because the field of view of the query image and the map image do not align sufficiently.
Chapter 3

Horizon Extraction

3.1 Introduction

A crucial advantage of the methodology proposed by this work is that the three dimensional (3-D) problem of pose estimation is solved in one dimension (1-D), instead of performing operations on 3-d images (two dimensions and color-space). This is made possible by identifying strong, repeatable 1-d features in the data and establishing methods to repeatably and reliably extract these features. The introduction of this document (chapter 1) defined these features to be the far-field horizons. These were selected because of their large size as a feature and relatively high visibility, general robustness to fast temporal changes, and slow movement and variability relative to local movement. Challenges associated with the horizon line feature are their susceptibility to being blocked by local occlusions and poor visibility during inclement weather and at night. This chapter discusses these issues in greater detail, introduces methods utilized in the state of the art for horizon extraction, and details the selection, implementation, and results of horizon extraction methods chosen for use in this work.

3.2 The Canny Edge

Though a variety of methods exist for determining the horizon contour in an image, all are based on edge detection methods. This makes sense because when the
horizon is visible it will form a stark edge between the sky and the ground. Though many methods of edge detection exist, perhaps the most popular is the Canny filter [8]. The Canny filter is an intensity based gradient detector that determines edges in an image with robustness to noise and the orientation of the edges. First the image is smoothed with a Gaussian filter to eliminate the weakest edges. Next the derivative is taken vertically, $G_x$, and horizontally, $G_y$. The edge is defined as the magnitude of $G_x$ and $G_y$ and the orientation of the edge is determined. If $G_x$ and $G_y$ are small, then the magnitude of $G$ will be small and it will be classified as a weak edge.

These edges are ‘binned’ into four categories: horizontal, vertical, and upward or downwards diagonal. This definition is used to perform ‘non-maximum suppression’. This is a sharpening step that narrows the width of the edge and includes only those pixels that form the central portion of the edge. Next, a process called ‘double thresholding’ is applied. This process checks the strength of edges against a high and low threshold. All edges above the high threshold are considered strong edges, all those below the low threshold are dismissed. Those edges that fall between the thresholds are ‘weak’ edges that require additional vetting. The final step of the process determines if the weak edges should be suppressed or remain in the image. Weak edges that are in contact with strong edges are kept, those that are not in contact with a strong edge are suppressed. This is determined using 8-connectivity. Connectivity defines how pixels connect to one another. A pixel is surrounded by eight other pixels; if the pixel in question is a weak edge and one of the 8 surrounding pixels is a strong edge, the weak edge is kept. Otherwise it is dismissed. Results from Canny’s paper are presented in figure 3.1.

The Canny filter is one of many works in the field of edge detection and has served as a cornerstone for many edge detection applications. Once the edges in the image are defined, the single edge that represents the horizon must be identified and isolated. The following section discusses how this has been accomplished in the literature and what methods might be used by the proposed work to accomplish this.
3.2.1 Naive Canny edge technique

The most obvious strategy for horizon detection is to simply use edge detection and logic based on the test cases defined above. On a clear day, it would be reasonable to assume that the sky is generally featureless and the edge detection algorithm can be tuned to ignore features like clouds. When edge detection is performed on the input image, a number of lines will appear but they should all be at or below the horizon line. Assuming gravity is pointing down along the vertical axis of the image, the topmost edge pixel can be identified as the horizon (illustrated in figures 3.3 and 3.4). This is called the ‘waterfall’ technique because it is accomplished by iterating down image columns from the top of the image (‘falling’) until it finds the first edge, which is then identified as the horizon. It is used in [35], [3], [7], and [37]. The primary assumption of this method is that nothing in the sky could be identified as an edge. This assumption fails in the presence of unmapped aerial features such as power-lines and trees. This method is therefore only applicable in
This first method, using a Canny filter and waterfall edge selection, was implemented as described in Figure 3.2. Using MATLAB, a static input image is converted to a grey scale image. A Canny filter is applied to the grey scale image. This is done with a built-in function in MATLAB’s image processing toolbox. The top-most edge is selected from the Canny edge image by iterating down column-wise and selecting the first edge pixel found. This produces the result seen in Figure 3.3.

This method has been found to reliably extract the topmost line in an image when there exists stark contrast between the sky and ground. Figure 3.3 demonstrates a case in which this method has been demonstrated to work the best.

This method fails when the horizon does not form a strong enough edge in the intensity domain to be detected. This will primarily occur at night and in overcast or stormy days. This method will also yield erroneous results whenever the topmost edge is not the horizon. This situation is common in an urban environment where power-lines, trees, buildings, and stoplights all create edges above the horizon. A case in which the waterfall method fails is shown in Figure 3.4.

### 3.2.2 Down-sampling the edges with a Gaussian blur

Filtering of the input image before doing edge detection can weaken edges formed by confounding features. A power line, for example, is thin compared to the environment around it and the edges it forms can be weakened with a Gaussian filter that would have the effect of blurring the image. Strong edges, like that which separates the sky from the ground, are generally unaffected. This method was used in [6] and [12], and is far less sensitive than the waterfall method mentioned above.
Figure 3.3. This is the output image from raw Canny filtering. The extracted horizon line is displayed in blue.

Figure 3.4. The Canny waterfall method is an intuitive and effective approach to horizon extraction, but it is easily foiled by overhead occlusions such as this. This mis-identification could also be caused by stark clouds, etc. Another method must be identified for the system to be robust to environments like this.
Figure 3.5. This figure demonstrates the process for the Gaussian blur approach to horizon extraction. This method is effective in consistently identifying the horizon region, but provides a low resolution horizon line.

This method adds no additional computational load over the waterfall technique if the blur is applied manually by re-focusing the lens. Depending on the degree of blur in the image, almost all edges can be suppressed up to the horizon line, significantly down-sampling the number of candidate edges. The disadvantage of this method is that in the blurring process much detail is lost. Tuning the blurring is also highly application specific and it is possible that this method would still identify the wrong horizon in certain circumstances.

This method is identical to the waterfall technique but with the addition of a Gaussian blur before the edge detection. The blur step weakens all edges, leaving only the strongest contrast regions in the image. This eliminates overhead obstructions such as power-lines and fences, leaving the strongest edges such as the horizon largely unaffected. A major benefit of this method is that it can be implemented if needed by simply sampling with an unfocused camera, adding no computational load to the process. A flowchart for the system is shown in Figure 3.5.

Figure 3.6 demonstrates the methods robustness to obstructions that the waterfall method cannot handle; however, as shown in Figure 3.6 the resultant contour is noisy and of low resolution. This is a natural artifact of the blurring process. It is incapable of identifying the actual horizon line to pixel-level accuracy in almost all cases, but is extremely reliable at identifying the horizon region.

### 3.2.3 Image partitioning

Perhaps the most sophisticated of the proposed techniques, the boundary between the sky and the ground can also be determined by using the color-space of the
Figure 3.6. This is the horizon line extracted using the Canny filter method after a Gaussian blur has been applied. Note the amount of noise in the resultant line. In blue the results from the original Canny waterfall technique are demonstrated to be insufficient, whereas the results from the Gaussian blur are the true horizon contour.

image. Once a rough horizon has been determined using the Gaussian blur method detailed above, it is used to partition the image into general areas, assuming - as in the naive approach - that the area above the horizon represents the sky. An average pixel value is determined for the sky region and all image pixels are sorted into 'sky' and 'not-sky' based on it’s proximity to the average sky pixel in terms of standard deviation. Morphology is used to fill ‘holes’ in either region, filtering out mis-classified regions and separating the image into two distinct portions. Canny edge detection is done on the resultant image and the one edge determined is the horizon. Partitioning of the image with respect to color space was used for horizon extraction in these papers: [30], [29], [34], [31]. This solution should contain all of the robustness of the Gaussian blur technique and yet return a much higher fidelity horizon line.

This assumes a sufficiently large difference in the color-space of each that this boundary can be determined with high fidelity. In cases such as Figure 3.7 this
Figure 3.7. An example of a difficult horizon extraction problem as the mountains fade into the mists in the distance. Here the difference in the color-space of the sky and ground is very small.

would not be true as the fog and sky blend together so perfectly that even where the horizon is visible it is not sufficiently different from the ground to provide a stark bifurcation of the image. This is also by far the most computationally expensive of the proposed methods as it involves all the computations as in the previous method as well as three dimensional statistical analysis of every pixel. It also requires that there either exist two cameras or that a Gaussian blur operator is used on a properly focused forward-facing camera.

The image partition technique seeks to solve the horizon extraction problem by separating the image into two regions: sky and ‘not-sky’. This method combines the Gaussian blur and waterfall techniques. The Gaussian blur approach is used to first identify a horizon region. Using the rough horizon contour output from the Gaussian blur, the original, un-blurred image is segmented into sky and not-sky approximations, and the average RGB value of all pixels above the horizon is determined and used as the average RGB pixel for the sky. All pixels are then sorted based on their proximity to the ‘average sky’ pixel. Pixels within six sigma
Figure 3.8. This method utilizes the robustness of the Gaussian blur method while maintaining fidelity to the true horizon contour.

of the average are marked as sky and all others are ‘not-sky’. A binary image is then generated with these two regions. Morphology is used to identify holes in either region and fill them in. Finally the waterfall method of edge detection is used in reverse to detect the edge closest to the ground. This is done to avoid overhead obstructions, assuming the ground is more uniform than the sky. These steps are shown as a flowchart in Figure 3.8.

The accuracy of the horizon contour identified by the partitioning method as compared to the Gaussian blur can be seen in Figure 3.9.

Where this method fails is when the two regions are not uniform. As seen in Figure 3.12, one primary failure of this technique is in the presence of lane lines. Being highly reflective, these are identified as sky and often bifurcates the image such that they are not identified as holes during the morphology stage. In many cases this leads to an inaccurate horizon line.

3.3 Results

More than fifty static images taken in and around State College, PA were used for preliminary testing of these methods. These images were selected for their variety and representation of expected problems such as occlusions, overcast weather, and vast orientation changes. These did not make up a sequence of images but instead were a series of static images collected separately at unique locations and times.
Figure 3.9. Using the results from the Gaussian blur approach, here the sky/not-sky partitioning technique is demonstrated in comparison with the Gaussian blur. Notice how much cleaner the results are from the partitioning method.

For these images, the Canny waterfall method achieved a suitable horizon line in 28.3%, the Gaussian blur technique worked in 83%, and the partitioning method worked in 52.8%. This represented to the authors that in the presence of the confounding conditions that are expected, the Gaussian blur method is the most robust. It still presents the problem of identifying a low resolution horizon line that usually requires filtering of some kind before it can be used.

Figure 3.10 demonstrates the inability of the waterfall method to identify the horizon contour in the presence of overhead obstructions. The Gaussian blur method was able to successfully identify the correct region in most cases, but only the image partitioning method successfully identified the correct horizon line with high accuracy. Also notice the problem introduced by the grove of trees on the left hand side of the image. This area will change seasonally and therefore would be a poor feature to use for map-matching. The Gaussian blur and partitioning methods represent this area differently, demonstrating the problem areas like this will present in this work.
Figure 3.10. This image demonstrates the failure of the simple Canny-edge waterfall technique. In the presence of overhead obstructions like power lines, the waterfall method stops whereas the Gaussian blur is able to accurately identify the horizon line with noise and the sky/not-sky partitioning can do so with high accuracy.

Figure 3.11 represents a scenario in which the boundary between the horizon and the ground is too faint to be distinguished from the gray, slightly overcast sky. In this case the waterfall and Gaussian blur techniques both fail to accurately identify the entire horizon contour. The partitioning method works better but it is still not entirely accurate. Furthermore the building in the foreground is identified as the horizon in each case. This motivates an iterative process of horizon detection that involves a combination of methods and repeated querying of an area localized as the horizon region. Furthermore, filtering of the horizon contour could allow the elimination of unmapped features such as buildings that appear as flat lines with sharp corners.

In Figure 3.12, the vehicle itself causes problems as the A and B posts are partially in the image. This causes both the Canny waterfall technique and the Gaussian blur to misidentify it as a portion of the horizon. The Gaussian blur represents the best match of the horizon with the exception of confusion at the B
Figure 3.11. Because of the slight overcast on the day this image was taken, both the waterfall and blur methods in red and green are unable to distinguish an edge at the horizon towards the center of the image. In this case, the sky/not-sky partitioning in blue was critical in correctly identifying the regions, but even here the horizon line is not perfect throughout.

post, where the Canny filter exhibits erratic behavior where the brush and trees are. Furthermore, a major failure of the image partitioning method is seen wherein the lane line is identified as being part of the sky region, leading to a gross mis-identification of the horizon contour by this method.

3.4 Conclusions

This section discusses the primary challenges that may be involved in horizon extraction and develops methods for mitigating these issues. The most laborious of these problems has been the existence of occluding features, both overhead (power-lines, trees) and near-field (vehicles, poles, traffic lights). Recognizing that this dissertation does not intend to develop or study the image processing necessary to obtain horizons, the authors chose to mitigate these issues by choice of
Figure 3.12. This image most importantly demonstrates the failure of the image partitioning technique to distinguish the sky from the bright lane markers.

Data instead of with image processing techniques. One way to mitigate occlusions is defining a custom map database that includes all of the man-made features that will be observed (as described in section 2.4). Furthermore the environment can be controlled by only collecting query sets when un-mapped occlusions are minimized or not present (i.e. on controlled roads where other vehicles will not pass through the line of sight). Finally, issues associated with weather can be avoided by collecting data on days with minimal overcast and during daylight hours with adequate lighting so as not to introduce errors in horizon extraction. With these assumptions in place, the naive waterfall method has been demonstrated to perform adequately and will be used for all subsequent tests unless otherwise noted.
Chapter 4

Decomposition of the horizon contour into feature sets

4.1 Introduction

Once horizons are extracted from the images, correspondence must be achieved. Correspondence is necessary because direct overlap of horizons cannot be assumed due to differing fields of view of the camera and the very orientation differences we are seeking to determine. In order to ensure horizons are being compared ‘apples to apples’, each contour must be broken down into a series of features that define the unique characteristics of the line. These features must described the line with high enough dimensionality that a unique match can be determined from them, but they must also be of low enough dimensionality that a robust match exists and can be regularly identified. In this section, we will explore the options available for feature characterization of a horizon contour and matching of feature sets, the selection and development of the methods used in this study, and results from these methods in test cases.

For this study, the requirements on features are as follows:

1. Rotational invariance. This is the primary requirement because the purpose of this study is to examine horizon contours at different orientations and determine their rotational offset. This requirement is dependent on the amount of trust that can be placed in our sensors. Assuming that no IMU exists in
the system whatsoever and that inertial sensing relies entirely upon the devised system proposed here, rotation invariance is the only way to ensure that the system is applicable for large variations in roll and thus this requirement must be tight. If an IMU is introduced, with increasing IMU reliability the strictness of the rotational invariance correspondingly goes down.

2. Robustness to noise. Noise can be introduced in a variety of ways; perhaps most commonly via occlusions, lighting conditions, and pixelization. Noise can be mitigated by using high quality sensors; however, sources such as occlusions and lighting conditions that cause erroneous or corrupted measurements are difficult to eliminate.

3. Scale invariance. There are two things that would introduce scale variance: errors in position localization and differences in camera parameters between the map and measurement. This work assumes good localization of position and camera parameters are static and easily measured, therefore scale invariance will have a low priority.

4. Translational invariance. This requirement is the lowest priority because this work assumes good localization of position and utilizes far field features, both of which are chosen specifically to reduce the dependence of the algorithm on translational errors.

Feature characterization requires a balance of using enough features so that uniqueness is ensured (i.e. only one match can be found) but not using so many that the feature set are over-fit (i.e. the features are not so stringent that no matches can be found). There are a number of problems that could be encountered using the location of local extrema as features, which may motivate the use of more or different features.

4.2 Feature extraction

4.2.1 Introduction/background

Two extremely popular algorithms for feature characterization are scale-invariant feature transform (SIFT) [46] and speeded-up robust features (SURF) [47]. These
algorithms are commonly used on images for optical flow. SIFT is an algorithm that caters towards three dimensional data sets, specifically images (two dimensional image and RGB colorspace). It identifies ‘key points’ in an image by computing an approximation of the Laplacian of Gaussian (LoG) function. This is essentially the second derivative of a blurred version of the input image. The second derivative identifies edges, and the blur helps to eliminate noise. This process is repeated for ‘octaves’ and ‘scales’, where octaves represent changes in the size of the image, and scales represent the amount of blurring applied. Maxima and minima of the LoG are compared across octaves and scales and only the most consistent are kept as ‘key-points’. Once key-points are identified, they are assigned an orientation based on an aggregate direction of gradients. This gives the key-points rotational invariance. This method has extremely good accuracy and is commonly employed, but is computationally demanding and slow and has historically bad results under stark lighting changes.

SURF is a variant of SIFT that seeks to utilize additional mathematical approximations and short-cuts to achieve the same results as SIFT. The procedure remains identical, however; search for scale and noise invariant local minima and maxima in the intensity domain and make them rotationally invariant by normalizes with respect to a baseline orientation.

A major difference between SIFT, SURF, and the problem of this work is that SIFT and SURF are designed for three dimensional data sets, whereas the data examined herein is one dimensional. There are, however, analogies that can be made between the high and low dimensional problems. Where SIFT and SURF look for local minima and maxima in intensity space, an obvious choice of feature set for this work would consist of literal peaks and valleys in a mountain range. This method is common, presented most notably in [3], [31], and [37]. The position and magnitudes of extrema (peaks and valleys) relative to each other is used as the feature set. There are a number of variations on this method. In [31] the angle of peaks from the focal point of the camera is used as a feature, as well as the magnitude of the peak from normal. In other applications, the peaks are connected by vectors and the magnitude and direction of these vectors is the feature set. Noise can be particularly problematic when using this approach because these features rely so heavily upon the correct identification of peaks and troughs. Orientation
is also a confounding factor in the identification process as the definition of local extrema may change as the contour rotates.

A classic solution to the feature characterization problem is principal component analysis (PCA), first introduced by Karl Pearson in 1901 [48]. This method seeks to describe a set of possibly correlated data using ‘principal components’, which are axes along which the data has increasingly smaller variation. The first principal component is the axis along which the data has the largest variance, the second principal component has the second largest variance, and so on. This method could be applied to this problem by comparing the principal components of two sets of horizon line features for similarity.

Another prevalent method used in the literature describes lines as a series of curves using the curvature scale space (CSS) [49]. This method describes curvature as a ratio of the horizontal and vertical components of the vectors making up the curve. For this formulation, we describe a curve as a vector parametrization such as:

\[ r(u) = (x(u), y(u)) \]  

Here \( x \) and \( y \) are the vertices of the \( u^{th} \) vector in \( r \), where \( r \) represents the parameterized curve. The vector itself is defined by:

\[ \dot{r}(u) = (\dot{x}(u), \dot{y}(u)) \]  

\[ |\dot{r}| = \sqrt{\dot{x}^2 + \dot{y}^2} \]

The horizontal and vertical components of the vector \( r \) can be defined as follows:

\[ t(u) = \dot{r}/|\dot{r}| = (\frac{\dot{x}}{\sqrt{\dot{x}^2 + \dot{y}^2}}, \frac{\dot{y}}{\sqrt{\dot{x}^2 + \dot{y}^2}}) \]  

\[ n(u) = (\frac{-\dot{y}}{\sqrt{(\dot{x})^2 + (\dot{y})^2}}, \frac{\dot{x}}{\sqrt{(\dot{x})^2 + (\dot{y})^2}}) \]

Where \( n \) is the normal component and \( t \) is the tangential. The curvature of this line segment is then defined by:
\[
\dot{t}(s) = K(s)n(s) \quad (4.6)
\]
\[
\dot{n}(s) = -K(s)t(s) \quad (4.7)
\]
Where \(K(s)\) is the curvature of the line. In [49], the robustness of this method to scale, resolution, noise, and orientation differences is demonstrated. This makes it a very attractive feature set because it satisfies all of the major criterion required for the proposed work. This method is used in [29], [34], [50], and a similar approach is detailed in [4].

Another primary concern for this work is the orientation dependence of the extrema detection. Feature selection must be done so that the same features are extracted from a contour for any given rotation of that contour. The use of raw maxima and minima in a fixed frame gives incorrect results for defining features, as is demonstrated in figure 4.1.

In [51], Chew et al from Cornell university developed a metric for comparing closed polygons. This method is based off the ‘turning angle function’, which is defined as the angle of the vector tangent to any point in the contour relative to some arbitrarily selected origin point. This method is invariant to scale, rotation, trans-
lation, and noise. It achieves invariance between contours \( A \) and \( B \) to some relative rotation \( \theta \) and translation \( t \) using the following equation:

\[
d_p(A, B) = \left( \min_{\Theta \in \mathbb{R} \in [0,1]} \int_0^1 |\Theta_A(s + t) - \Theta_B(s) + \theta|^p ds \right)^{\frac{1}{p}}
\]

\[
= \left( \min_{\Theta \in \mathbb{R} \in [0,1]} D_p^{A,B}(t, \theta) \right)^{\frac{1}{p}} \quad (4.8)
\]

Where

\[
D_p^{A,B}(t, \theta) = \int_0^1 |\Theta_A(s + t) - \Theta_B(s) + \theta|^p ds \quad (4.9)
\]

Both CSS [49] and the turning angle function as presented above offer the required orientation and noise robustness, however they are also computationally burdensome processes. A simple solution to these problems is to compare approximations of the horizons instead of the horizons themselves. In [30] published in 2005, Bao proposed to extract the horizon line from the forward facing camera on an micro aerial vehicle (MAV) even in the presence of adverse weather conditions such as fog. Orientation differences were determined based on a first-order fit of the horizon. This method is called the orientation projection method. The orientation of this first-order line was used as the aircraft’s roll angle by assuming that the camera was far enough away from the horizon that it could be generally assumed to be flat and perpendicular to the gravity vector. This assumption falls apart in the presence of occlusions such as buildings which will slant a first-order fit of the horizon away from perpendicular to gravity. For a rough fit, however, this assumption should hold. Rotational invariance can be achieved by utilizing a first-order fit to rectify a horizon to a common axis before applying extrema detection.

4.2.2 Method selection and implementation

As demonstrated in Figure 4.1, extrema definition is meaningless without a coordinate frame. In order to establish a consistent, repeatable coordinate frame across contours, each horizon contour is fit with a first-order line. This first-order line is resolved to a common x-axis by determining the angle between the first-order line and the x-axis then rotating the contour by the negative of that angle.
This approach was partially inspired by Lobo and Dias in [5], who used the same first-order technique to identify vanishing points and thus the horizon.

Fits were achieved using linear least squares polynomial curve fitting. This fit is done in $y$ for some $x$, which is to say that the fit follows the form $y = mx + b$. This approach works well until the contour is rotated past $45^\circ$, at which point the axis effectively ‘swap’ and the relationship becomes $x = my + b$. Since this rotation cannot be known a priori, the first-order fit is done both for the set $(x, y)$ as well as $(y, x)$. The sum of squared residuals, $\sum (y_{\text{first-order}} - y)^2$, were then compared and the the fit corresponding to the lowest residuals was used.

Once a first-order fit was identified, the angle between that line and the $x$-axis was determined using the dot product. The entire, original contour was then rotated by the resulting angle.

With a common coordinate frame assigned to each contour, the next step requires that the contour be represented in terms of features. The features ultimately chosen to define the extrema are edges. Edges are vectors that connect two points in the contour and are defined by their vector length $V$ and orientation relative to a normal plane, $\theta$. Mathematically, for a set of $k$ local extrema defined by $x$ and $y$, the features that defined the $n^{th}$ extrema are:

\[ F_{n1} = \sqrt{(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2} \]  
\[ F_{n2} = \tan \left( \frac{x_n - x_{n-1}}{y_n - y_{n-1}} \right) \]  

where $F1$ is feature 1 and $F2$ is feature 2. This procedure is demonstrated in Figure 4.2. In many ways, this feature extraction method is a reflection of SIFT in one dimension.

Contours are next down sampled to retain only their extrema or inflection points, (i.e.) the vertices at which the contour changes direction. These inflection points are determined by monitoring changes in the orientation of the edges.

Once the contours are represented by only their extrema points, they are now ready for correspondence.
Figure 4.2. This image demonstrates how features are defined in the contours. The two feature matrices $F_g$ and $F_b$ are defined by the vectors that connect local extrema, quantified by magnitude and orientation.

4.3 Correspondence

4.3.1 Introduction

Once the horizon lines are represented in terms of feature sets, correspondence can be achieved by determining what entries align in two separate feature sets. This is often as simple as finding the minimum of the Euclidean distance between feature vectors. Yang et al propose a feature matching process in [52] that enforces the rigidity of the contour. Using angles and distances as features, this structure-based approach adds integrity to the match by preventing physically unrealizable matches being made based on spurious data points.

In [37] the sum of absolute difference (SAD) algorithm is used to determine correspondence. The SAD algorithm takes two data sets and compares them by calculating the difference between corresponding points. These differences are added to generate the SAD value. To determine correspondence, one data set is moved along the other and the position that achieves the lowest SAD value is considered to be the position of correspondence.
4.3.2 Implementation

4.3.2.1 Recursive Bayes Filter

Assuming that two contours are at least partially alike, one termed the map \( M \) and the other the query set \( Q \), there exists a measurement model such that the query set is some function of the map:

\[
Q_d = M_d \times s_e + GRN_d
\]

\[
Q\theta = M\theta + \theta_e + GRN\theta
\]

(4.12)

(4.13)

Therefore there are three states that define the relationship between a measurement and a map: the edge in the map \( e \) that corresponds to the measurement, which defines a scaling factor \( s_e \) and an offset orientation \( \theta_e \). These measurements are assumed to be corrupted by some Gaussian random noise (GRN) on scale \((GRN_d)\) and orientation \((GRN_\theta)\). Thus we will call this triplet, \( e, \theta, \) and \( s \), our state vector \( x \).

These states are inherently connected because they all represent the same edge \( e \). The measurement model implies that for some measurement \( Q \), there exists a scaling factor \( s_e \) and an offset orientation \( \theta_e \) such that the measurement will match some edge \( e \). As such, for every \( e \) in a map \( M \) there is some combination of states that will make the map match the measurement. The goal, therefore, is to find the state triplet that makes all the edges in \( Q \) match edges in \( M \).

Using Bayesian statistics, we can formulate the question that ‘given a set of measurements \( Q \) and a map \( M \), what is the likelihood of a state \( x \) that connects the two such that \( Q = M \times x + GRN \)’? Mathematically, this is expressed \( p(x|Q) \). Bayes theorem states:

\[
p(A|B) = \frac{p(B|A)p(A)}{p(B)}
\]

(4.14)

Knowing that for \( A \) of size \( N \), \( p(B) = \sum_i P(B|A_i)P(A_i) \), in order to determine the probability of a state given a map \( p(x|Q) \), we must know the probability of a measurement given a state vector, \( p(Q|x) \), and the probability of a state vector, \( p(x) \).
Initially, we will assume that the prior probability for all states are equally distributed over a range. The orientation offset $\theta$ can be anything from $-180^\circ$ to $+180^\circ$. Therefore the prior probability for all $\theta$ is $\frac{1}{360}$. The edge $e$ can be any edge in the map; thus for a map of size $N$ there are equal priors for all map edges equal to $\frac{1}{N}$. Scale is more elusive because the range of possible scales is theoretically infinite. In practice, however, if we assume the position localization is good and the cameras are calibrated and the parameters are known, scale can also be known with relative certainty. Thus we will constrain scale to a relatively small range from 0 to 2 with equal priors across the range.

Assuming equal priors across all states, the state probability $p(x)$ can be approximated to be separable, so $p(x) = p(s)p(\theta)p(e)$.

Furthermore scale and orientation will always be independent of each other and dependent on edge, so the probability $p(Q|x)$ can be separated:

$$p(Q|x) = p(Q_d|x)p(Q_\theta|x)$$  \hspace{1cm} (4.15)

In other words, offset orientation and scaling are separable. They are, however, related by the edge they represent. Thus we can say:

$$p(Q_d, Q_\theta | s, \theta, e) = p(Q_d|s, e)p(Q_\theta|\theta, e)$$ \hspace{1cm} (4.16)

$$= p(Q_d|s, d_e)p(Q_\theta|\theta, \theta_e)$$ \hspace{1cm} (4.17)

This is to say, that the probability of a set of measurements given a state is equal to the probability of a measurement of vector distance given a scale and the vector distance at that edge $d_e$ multiplied by the probability of a measurement of orientation offset given an orientation offset and the orientation at that edge $\theta_e$.

Thus using Bayes theorem, we can state:

$$p(x|Q) = \frac{p(Q|x)p(x)}{\sum_{scale} \sum_{\theta} \sum_{edge} p(Q|s, \theta, e)p(s, \theta, e)}$$ \hspace{1cm} (4.18)

This is iteratively computed for all combinations of scale at every measurement in a set $Q$ for every entry in a map $M$. For each measurement $Q_i$ the most likely state is computed as indicated above in equation 4.18. This equation, however, only tells half of the story. We cannot simply perform this computation for every mea-
surement, as it will always give us the best state for that particular measurement irrespective of the measurement as a set. Horizon lines are rigid; the orientation offset $\theta$ and scale factor $s$ must be equal for all measurements $Q_i$ in $Q$. This enforces that contours cannot be ‘bent’ to achieve a fit. That is to say that if we know that the orientation offset is $\theta$ with some certainty $P$, then we know that the orientation offset at the next measurement in the set should also be $\theta$ with equal probability $P$. Also, if we know edge alignment with some certainty $C$, we know that the next measurement in the query set should correspond to the next edge in the map. Thus we utilize knowledge of the contour as a set by establishing new priors for the next step of iteration as follows:

$$p(x_{n+1}|M_n) = \sum_{\theta} \sum_{s} \sum_{\text{edge}} p(s_{n+1}, \theta_{n+1}, \text{edge}_{n+1}|s_n, \theta_n, \text{edge}_n)p(s_n, \theta_n, \text{edge}_n|M_n)$$

(4.19)

Certainty grows with every iteration because it is using all of the information from previous measurements. Therefore the certainty associated with the last state will be the highest. From this we can determine the scaling factor, edge correspondence, and orientation with statistical certainty in our answer. This gives us the advantage of being able to establish trust that can be used later when combining estimations of orientation.

This method is mathematically rigorous and implementation is complex. Initial tests proved that the method was slow in implementation to the extreme. This method will be reserved for future work, however in initial testing a simpler solution has been utilized for feature characterization.

### 4.3.2.2 $k$-d tree Nearest Neighbors Search

Using the rotational and translational invariance of the features as defined above, correspondence can also be determined independent of the state by searching for similar sequences of features in the query and map sets. This nearest neighbor problem can be solved with the use of multidimensional binary trees, or $k$-d trees. $k$-d trees were first introduced by Bentley in his 1975 paper [53] and proposes a structure for searching for nearest neighbor of a query set in a map of $k$-dimension.
For a training set of size $m$ and a query set of size $n$, a brute force method would have to perform $m \times n \times k$ computations to determine the nearest neighbor. $k$-d trees simplify this by partitioning data. The $k$-d tree structure cuts down significantly on this computational burden through the use of ‘pointers’. Pointers help to constrain the search by ‘clumping’ map entries into categories; i.e. the first dimension of this entry is greater than one. This allows the search-space of the map to be significantly reduced in all dimensions.

The correspondence problem can be solved using $k$-d trees by defining a sequence of $k$ consecutive features as a $k$ dimensional tree. A map $M$ of size $l$ will be represented by $\frac{l}{k}$ sequences. These sequences will be $\theta_n, \theta_{n+1}, \ldots, \theta_{n+k}$ for $n = 1$ to $n = l - k$. Each sequence of $k$ features is stored as a single node in $k$ dimensions. The query set is then also decomposed into a series of line sequences, each of length $k$. For the purposes of this work, the features stored in the $k$-d tree are composed only of the relative orientation of features to one another, as defined in figure 4.2.

The length $k$ of these sequences is determined by how long of a continuous line segment can be reasonably expected to be observed in both the map and query images. Experimentally, that was determined by observing the number of correct matches as a function of size $k$. This size $k$ will change as a function of position. In more rural environments $k$ will most likely be large relative to the overall length of the horizon contour. Contrarily, in cluttered, urban environments where occlusions are expected, the size of $k$ will decrease.

Each sequence in the query set will be matched to a sequence in the map, so for each query there will be $l - k$ matches. Each match has a corresponding Euclidean distance from the query point to the matched map point. Since rigidity of the horizon requires that only one match be used to determine correspondence, the best match was selected as the match with the corresponding smallest Euclidean distance.

### 4.4 Conclusions

This chapter defines features to define a horizon as the local extrema, or maxima and minima of the contour. Rotational invariance is achieved by performing a rough first order fit of the contour and resolving that to the $x$-axis. Once the
horizon is characterized in terms of rotationally invariant maxima and minima, correspondence is achieved using a simple \( k \)-d tree matching feature sequences of length \( k \). This sequence matching approach raises the dimensionality of the feature matching problem, allowing the authors to tune the sensitivity of the match to provide a unique solution without over-matching.

With the horizon defined as a feature set and correspondence achieved, the next step is for the transformation between horizons to be determined.
Relative Orientation Between Feature Sets

5.1 Introduction

With horizons represented as a series of features where the set correspondence is known, and feature set correspondence achieved, horizon contours can now be compared to mapped horizon features in order to mathematically determine orientation offsets. This process is commonly referred to as ‘registration’. These orientations in reference to a coordinate frame as defined by figure 5.1 are: roll - rotation about the vehicles $x$-axis, pitch - rotation about the vehicles $y$-axis, and yaw - rotation about the vehicles $z$-axis. This chapter discusses the mathematical formulations for determining orientation offsets between two 1-d contours.

5.2 The affine transformation matrix

This work assumes that, once correspondence is achieved, for some map array $M$ and some query array $Q$, there exists a matrix $A$ that connects the two, such that $Q = A \times M$. This matrix $A$ is called the affine transformation matrix. For 2-D arrays, the affine transformation matrix can be of size $2 \times 2$ or $3 \times 3$. In the former case the affine transformation can contain information about scale and rotation. In the latter case, arrays of length $n$ must be represented in homogeneous coordinates.
of size $3 \times n$, or of the format $x, y, 1$, and the resulting affine transform contains information about rotation, translation, scale, and shearing.

In 2-D, the affine transformation takes the form:

$$
A = \begin{bmatrix}
    s_x \times (\cos(\theta) - h_y \sin(\theta)) & s_y \times (h_x \cos(\theta) - \sin(\theta)) \\
    s_x \times (\sin(\theta) + h_y \cos(\theta)) & s_y \times (h_x \sin(\theta) + \cos(\theta))
\end{bmatrix}
$$

(5.1)

where $\theta$ represents rotation, $s_x$ and $s_y$ represent scaling factors, and $h_x$ and $h_y$ represent shear factors. With known camera parameters, scale factor will not be unknown, therefore can be set to 1. Further, for features in the far-field, shearing effects will be negligible, thus $h_x$ and $h_y$ can be set to zero. This reduces the 2-D affine transformation to a simple rotation matrix:

$$
A = \begin{bmatrix}
    \cos(\theta) & -\sin(\theta) \\
    \sin(\theta) & \cos(\theta)
\end{bmatrix}
$$

(5.2)

In 3-D the affine transformation includes two additional rotations and translation. Pitch and yaw will both be observed in translation. Thus the 3-D affine transformation takes the form:
\[
A = \begin{bmatrix}
\cos(\theta) & -\sin(\theta) & t_x \\
\sin(\theta) & \cos(\theta) & t_y \\
0 & 0 & 1 
\end{bmatrix}
\]  \hspace{1cm} (5.3)

where \( t_x \) is translation in the \( x \) direction and \( t_y \) is translation in \( y \).

Because the features of interest are in the far field and relative perspective changes can be assumed to be small, pitch and yaw can be determined directly from translation. This comes from the pinhole projection model of a camera. Simply stated, the pinhole camera model represents a camera as a pinhole \( p \) that is located one focal length \( f \) away from the image sensor. All light is focused through the pinhole and projected onto the image sensor. Given the vertical and horizontal fields of view of the cameras, \( \text{fov}_{\text{vertical}} \) and \( \text{fov}_{\text{horizontal}} \), and an image of size \( m \times n \) pixels, each pixel represents some arc-length called the visual angle, \( V \) \cite{54}. This visual angle can be calculated as:

\[
V_{\text{vertical}} = \frac{\text{fov}_{\text{vertical}}}{m} \hspace{1cm} (5.4)
\]

\[
V_{\text{horizontal}} = \frac{\text{fov}_{\text{horizontal}}}{n} \hspace{1cm} (5.5)
\]

Thus for some translation error \( x \) and \( y \), pitch \( \theta \) and yaw \( \psi \) can be determined:

\[
\theta = x \times V_{\text{vertical}} \hspace{1cm} (5.6)
\]

\[
\theta = y \times V_{\text{horizontal}} \hspace{1cm} (5.7)
\]

While it is tempting to use homogeneous coordinates to calculate all three desired transformation parameters at once, in practice this complicates the procedure mathematically. Instead of performing additional mathematical calculations to extract roll and translation from a 3-D affine transformation, the 2-D affine matrix \( A \) for arrays \( Q \) and \( M \) can be determined, applied, and then translation errors \( x \) and \( y \) determined as follows:

\[
x = A \times \bar{Q}_x - M_x, y = A \times \bar{Q}_y - M_y \hspace{1cm} (5.8)
\]
Now all that remains is to determine the affine transformation matrix.

5.3 Linear least squares regression

A challenge of determining the affine matrix is that the data sets representing the contours will not align exactly, as various disturbances will cause small differences even after correspondence is obtained. Thus an exact transformation will not exist, but instead one must be found that is the best fit despite the noise. A common method for doing this is linear least squares regression (LLSR). This algorithm was originally developed for the prediction of celestial trajectories in the early 1800’s. The problem essentially solves the following equation:

\[ \text{Eq. 5.9} \]

\[ Y = AX \]

Where \( Y \) and \( X \) are known series of points and \( A \) is the transformation matrix that connects them. In the formulation for the horizon matching problem, \( A \) is the affine transformation matrix containing information about translation, rotation, skew, and scale. \( Y \) and \( X \) are the 2-D sets of \( x \) and \( y \) pixel coordinates for the horizon lines.

Defining the error between lines as:

\[ \text{Eq. 5.10} \]

\[ E = \sum |YAX|^2 \]

We want to identify an \( A \) that minimizes \( E \), thus we want to solve for \( A \) when the derivative of \( E \) with respect to \( A \) is zero. This can also be done using matrix algebra to solve the original formulation for \( A \). The result becomes:

\[ \text{Eq. 5.11} \]

\[ A = YX^T(XX^T)^{-1} \]

This was proven by Carl Friedrich Gauss in 1809 to be the optimal solution to the curve fitting problem.
5.4 Determination of relative orientation

The linear least squares regression (LLSR) algorithm was tested first by comparing identical line segments at different orientation angles, thus eliminating the correspondence problem. The recovered affine transformation was applied to the distorted contour and success was determined by the resulting contour’s overlap with the original. This was proven to work reliably even with the addition of Gaussian random noise to one signal, as can be seen in Figure 5.2. Over 500 tests without GNR added, the average percent error was negligible (mean of $1.4 \times 10^{-12}$, standard deviation of $2.7 \times 10^{-11}$).

This method when applied to actual data can only be an approximation as it is a linear approach to solving for a nonlinear system of equations. To test how accurate this approximation is, a trial set of random data was rotated by a known amount, and the rotation was determined using the methods described above. The method was proven to be robust across 360°s of rotation, as seen in figure 5.3.

The algorithm has been demonstrated to fail when correspondence is lost. A
Figure 5.3. A random contour was rotated through 360°s and the angle of rotation extracted at each rotation. The results in terms of percent error as presented here demonstrate that the approximation of linearity works quite well, even in the presence of Gaussian random noise added to the contour.

Loss of correspondence would lead to dissimilar contours being compared, i.e. data would exist in one or both that does not exist in the other. This is demonstrated in figure 5.4. In this figure, correspondence was artificially lost by comparing the percent error between contours as the percentage of the contour that overlaps is decreased. At 0% error in overlap, the two contours being compared are identical. Increasing percent error in overlap represents how much of one contour consists of data not represented in the contour it is being matched to. Figure 5.4 shows that even small errors in correspondence can lead to gross misalignment of the contours. This demonstrates the importance of the correspondence step described in chapter 4.
Figure 5.4. This figure demonstrates the percent errors in partially overlapping contours.

5.5 Conclusions

This chapter discusses the issue of registration for determining the transformation between two 2-D contours. Three parameters are considered: relative orientation in one dimension, and translation in two dimensions. It has been demonstrated that relative orientation can be calculated from these three transformations.

Linear least squares regression (LLSR) has been identified to be the registration method that will be utilized. This method has been proven to work if accurate correspondence exists. If correspondence has failed and two contours being matched are even slightly dissimilar, these errors will lead to extreme errors in registration.
The Kalman filter

6.1 Introduction

The methods presented thus far will provide an independent external measurement of orientation. By itself this overcomes the drift associated with state of the art inertial measurements, however also introduces new sources of error as will be discussed in chapter 7. In keeping with the original motivation of this work, it would be more powerful to utilize this external measurement with state of the art sensors to overcome the shortcomings of inertial measurement technology. This chapter discusses how the methods proposed in this work can be combined with the output of an IMU using a Kalman filter to make inertial measurements more accurate.

A Kalman filter is a popular method for fusing multiple sources of related information. In the context of this dissertation, Kalman filters are commonly used in automotive applications to combine mathematical models of vehicle motion with sensor measurements [18]. A Kalman filter classically consists of two components: a model and a measurement. Measurements are obtained with sensors and represent a real world view of current states. The model is a mathematical formulation of dynamic motion that contains within it a predictive capability that projects current states forward in time based on the physical possibilities allowed by the system. Both the measurements and models will have errors that the Kalman filter seeks to mitigate. It does so by iteratively establishing trust metrics in each. Estimates of covariance are established a priori and utilized as an initial trust
metric. As the process continues, the Kalman filter tracks how close each input is to the ultimately determined state and updates the associated covariances accordingly. These covariance matrices are used in a weighted average that combines the measurements and the model to output the best estimate of the states.

### 6.2 Kalman filter equations

Mathematically, the Kalman filter equations begin with a prediction step, or the determination of the states $x$ at the current time $k$ given measurements of $x$ at time $k - 1$. The derivation of the Kalman filter, as presented in Rudi Kalman’s seminal paper introducing the concept [18], are as follows:

$$x_{k|k-1} = Ax_{k-1} + Bu_{k-1} \tag{6.1}$$

Here $A$ and $B$ represent a dynamic model of the system, $x$ is a vector of states, and $u$ are the inputs to the system. Generically, this is represented by a dynamic system:

$$x_{k+1} = Ax_k + Bu_k + Bw_k \tag{6.2}$$

where $u_k$ is a vector of system inputs and $w_k$ represents the process noise caused by un-modeled or incorrectly modeled dynamics. The measurement model is defined as:

$$y_{k+1} = Cx_{k+1} + Du_{k+1} + Dv_{k+1} \tag{6.3}$$

where $v_k$ is the sensor noise. The specific dynamic model for this system is discussed below.

The covariance of this prediction is also calculated based on the prior covariance and system properties. This is important to the ‘trust’ aspect of the Kalman filter and will be used to combine measurements and predictions in a weighted average.

$$P_{k|k-1} = AP_{k-1}A^T + Q \tag{6.4}$$

Here, $Q$ represents the process noise as follows:
\[ Q = B_w \Sigma_w B_w^T \] (6.5)

Once measurements for the current time \( k \) arrive, they are combined with the prediction using the Kalman gain \( K_k \) as follows:
\[ K_k = P_{k|k-1}^T (C P_{k|k-1} C^T + R)^{-1} \] (6.6)

where \( R \) represents the sensor noise:
\[ R = D_v \Sigma_v D_v^T \] (6.7)

The covariance for the current estimate is determined:
\[ P_{k|k} = (I - K_k C)(P_{k|k-1})^{-1}(I - K_k C)^T \] (6.8)

Finally, a unified state estimate is generated using all of the information we have available to us.
\[ x_{k|k} = x_{k|k-1} + K_k (y_k - C x_{k|k-1}) \] (6.9)

where \( y_k \) are the current measurements. Once this is complete and the state estimate \( x_{k|k} \) is determined the system is advanced in time and the procedure repeated.

### 6.3 The dynamic model

Perhaps the defining characteristic of the Kalman filter is its dynamic model of state evolution. This accounts for the way related states affect one another and allows system designers to enforce system constraints. For attitude estimation, a common set of equations to define motion are the Euler equations:

\[ \dot{\phi} = p + q \sin(\phi) \tan(\theta) + r \cos(\phi) \tan(\theta) \] (6.10)
\[ \dot{\theta} = q \cos(\phi) - r \sin(\phi) \] (6.11)
\[ \dot{\psi} = q \frac{\sin(\phi)}{\cos(\theta)} + r \frac{\cos(\phi)}{\cos(\theta)} \]  

(6.12)

where \( p \), \( q \), and \( r \) are the angular rates as measured by the IMU.

This representation is ideal because it places no constraints on the platform being used. Alternative models may be used that contain additional information. On a ground vehicle, the above model can be modified to utilize position pose as well to constrain the vehicle to movement only along its axis of travel. This represents the reasonable assumption that lateral motion of a vehicle is so small as to be negligible. Such a model would not, however, be applicable to a boat or airplane, or to a slipping ground vehicle such as one on ice or involved in a collision.

A major benefit of this system is that this dynamic model is interchangeable and does not affect the subsequent steps. The only requirement of this model is that it accurately represents the propagation of states roll \( \phi \), pitch \( \theta \), and yaw \( \psi \).

### 6.4 The Extended Kalman Filter

As originally proposed by Rudolph Kalman, the Kalman filter as described above is based around a linear system of equations. The Euler equations (equations 6.10, 6.11, 6.12) described above are nonlinear, which seemingly prevents their application in the Kalman filter. As soon as the linear Kalman filter was introduced, however, engineers at NASA began development on a similar capability for nonlinear systems, as presented by Smith, Schmidt, and McGee in [55]. The nonlinear Kalman filter, also called the Extended Kalman Filter (EKF), defines the state model as:

\[
A = \frac{\partial A}{\partial X} \tag{6.13}
\]

\[
B = \frac{\partial A}{\partial U} \tag{6.14}
\]

\[
C = \frac{\partial C}{\partial x} \tag{6.15}
\]
Thus the state matrices are their own Jacobians with respect to the states and the inputs.

Since the Jacobians are continuous with respect to time, they must be linearized about the last estimate of state. This can be accomplished with a simple Taylor series expansion. Because of this linearization, the EKF now estimates changes in state, \( \partial x \) instead of the state itself.

### 6.4.1 System design

For this system, the states \( X \) will be roll \( \phi \), pitch \( \theta \), and yaw \( \psi \). Inputs \( U \) will be angular rates about the X-axis \( p \), about the Y-axis \( q \), and about the Z-axis \( r \). There will be six measurements: gyroscope outputs of \( p \), \( q \), and \( r \) measuring accelerations in \( x \), \( y \), and \( z \), and image based measurements of \( \phi \), \( \theta \), and \( \psi \). Based on the Euler equations above, the system will be represented as follows:

\[
\begin{bmatrix}
\dot{\phi} \\
\dot{\theta} \\
\dot{\psi}
\end{bmatrix} = A \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} + B \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{6.17}
\]

\[
\begin{bmatrix}
p \\
q \\
r \\
\phi \\
\theta \\
\psi
\end{bmatrix} = C \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} + D \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{6.18}
\]

Where the discrete time system state matrices \( A \), \( B \), \( C \), and \( D \) are:

\[
A = \begin{bmatrix}
(q \cos(\phi) \tan(\theta) - r \sin(\phi) \tan(\theta)) \Delta t + 1 & (q((\tan(\theta)^2) + 1) \sin(\phi) + r((\tan(\theta)^2) + 1) \cos(\phi)) \Delta t & 0 \\
(-q \sin(\phi) - r \cos(\phi)) \Delta t & (q \sin(\phi) \tan(\theta) \Delta t + q \sin(\phi) \tan(\theta)) \Delta t + 1 & 0 \\
(q \sin(\phi) \cos(\theta) + r \cos(\phi) \sin(\theta)) \Delta t & (q \sin(\phi) \cos(\theta) \Delta t + r \sin(\phi) \cos(\theta) \tan(\theta)) \Delta t & 1
\end{bmatrix} \tag{6.19}
\]
\[
B = \begin{bmatrix}
(cos(\phi) \tan(\theta) + 1)\Delta t + 1 & \sin(\phi) \tan(\theta) \Delta t & (cos(\phi) \tan(\theta) + 1)\Delta t \\
-\sin(\phi)\Delta t & cos(\phi)\Delta t + 1 & -\sin(\phi)\Delta t \\
\frac{f r a c \cos(\phi) \cos(\theta) \Delta t}{\cos(\theta)} & 1 & \frac{f r a c \cos(\phi) \cos(\theta) \Delta t}{\cos(\theta)} + 1
\end{bmatrix}
\]

\[C = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\]  

\[D = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

The covariance associated with each measurement is initialized in the matrices \(Q\) and \(R\). \(Q\) acts on the dynamic model formed by \(A\) and \(B\), while \(R\) acts on the measurement equation, formed by \(C\) and \(D\). \(Q\) is a term called ‘process noise’, which represents contributions of factors that cannot be represented mathematically (commonly referred to as ‘un-modeled dynamics’). \(R\) represents measurement noise, which contains information about sensor noise. These matrices are both square. \(Q\) is the size of the state matrix, in this case 3 \(\times\) 3, and \(R\) is the size of the measurement matrix, or 6 \(\times\) 6. If noise is uncorrelated then both matrices will have entries only on the main diagonal. Any off-diagonal entries represent correlation of noise parameters. Correlated noise exists when disturbances of one state affect another. The entries of these matrices are the expected covariance \(\sigma\) in the corresponding states/measurement.

Thus the \(Q\) and \(R\) matrices are as follows:
Because the Kalman filter updates these matrices automatically, absolute precision in the initialization of the matrices is not necessary. They are initialized with guesses of initial covariance. The longer the Kalman filter iterates, the more accurate it will become. Process noise was estimated to be $\sigma_{\phi, \theta, \psi} = 1^\circ$ on all states, based on trust in the Euler equations. Measurement noise for the visual estimates of roll, pitch, and yaw were determined experimentally from the stationary tests described in section 7.2. Finally, the IMU measurement of angular rates was simulated as described in section 7.3.3. This noise parameter was designed to be $\sigma_{\text{measured} p, q, r} = 5 \times 10^{-4} \text{ rad/s}$. This value is the standard IMU accuracy in automotive applications [56].

### 6.5 Conclusions

This Kalman filtering step is the last part of the process presented in this paper. At this point relative orientation has been measured twice: once by a simple first order fit to achieve rotational invariance of the features and thus correspondence, and again more robustly by linear least squares regression (LLSR). This second, more accurate representation of orientation is then fed into the Kalman filter along with an IMU. Implementation and results of this final step of the process are presented in chapter 7.
Chapter 7

Results

7.1 The test track facility

Primary testing was conducted at the Pennsylvania State University’s Larson Transportation Institute (LTI) test track facility (figure 7.1). The test track is a one mile oval that loops a handling circle, bus testing lane, and garages. The test track is located in the rural outskirts of State College, PA surrounded by mountains. It is bordered by trees to the south and has a large group of trees in the center of the loop. The north/north-west quadrant has an open view of the mountains across fields and the nearby University Park airport. It was on this northern portion of the track where the majority of testing was conducted.

7.2 Stationary tests

One of the primary concerns of this work is the repeatability of horizon extraction and the subsequent reliability of correspondence. While horizon contours will not physically change over time, temporal changes such as lighting and weather conditions can have an effect on the extracted horizon. These factors are often ignored in the state of the art. Noise will be introduced either by the image itself via environmental conditions, dazzling, etc, and by the extraction method, such as is seen with the Gaussian blur technique. Once the methods were defined as described above, the effects of weather and lighting conditions on feature extraction were studied. This has been accomplished by monitoring a horizon from a stationary
observation point over a forty hour period and calculating the orientation for each image using the methods presented in this dissertation. Because the camera was stationary during the entire period, variations in orientation could only be caused by algorithmic errors. Drift in orientation is used as a metric for studying when the algorithms fail.

For these tests a monocular camera was left stationary from approximately 1700 on Day 1 of testing until 0900 on Day 3 (40 hours). Images were collected every five minutes for a total of 485 images (figure 7.2). Over the forty hours, three major environmental conditions were observed: normal daylight, night time, and fog. The gravel pile in the foreground provided a ‘false’ horizon to potentially confound the algorithm. Over this time, the average error in each estimate was as follows: roll $0.13^\circ$, pitch $0.71^\circ$, and yaw $3.28^\circ$. These errors are not, however, equally distributed over the duration of the test. Errors in orientation over time of day are presented in figure 7.3. Figure 7.4 highlights the time of day these images were taken, and
Figure 7.2. A sample of the time-series images collected from a stationary position over 40 hours.

Figure 7.3. Results for relative roll in a 40 hour time series of images from a stationary platform observing unchanging terrain. As is evident, at night time readings are untrustworthy, however during the day results are reliable.

shows an obvious but important conclusion: when the environment is such that the horizon is not visible, horizon extraction techniques will not work. Furthermore, the foggy conditions were able to fool the algorithm into thinking that the gravel pile in the near-field was a horizon. This erroneous match demonstrates a pitfall of the horizon extraction technique and how it can be fooled into selecting the wrong contour. This could occur in a number of situations wherein the top-most horizon is not the same temporally.
Figure 7.4. This figure highlights the three major environmental conditions observed during stationary testing: daylight, night-time, and fog.

### 7.2.1 Sequence length requirement

A major result of the stationary testing was the determination of the feature-sequence length necessary to achieve a good match. In section 4.3.2.2, it was described how correspondence would be achieved by matching sequences of $k$ features using a $k$ dimensional $k$-d tree. In these stationary tests correspondence was easy to determine because horizons alignment was one-to-one (or nearly so). As described in section 4.3.2.2, a map of size $l$ will consist of $\frac{l}{k}$ feature sequences, and thus the $k$-d tree matching will provide $\frac{l}{k}$ matches. The match with the smallest associated Euclidean distance is selected as being the correct one. Using the stationary images, the sequence length $k$ was varied and the percentage of matches that were correct was determined for each length $k$ (figure 7.5). The percentage of correct matches predictably rises as dimensionality increases until a point at which the feature set becomes over-defined and a unique match is more difficult to find. As the sequence length approaches the size of the feature set, the probability of a good match rises again. This is misleading because for these tests correspondence was one-to-one, which is essentially what a sequence length $k = l$ represents.
Figure 7.5. Determination of the optimal sequence length $k$. This was determined by averaging the percentage of correct matches for a map to 48 separate test images for a range of sequence lengths $k$.

correspondence is not exact, this relationship will deteriorate. Furthermore, a feature set so large is undesirable because the likelihood of over-fitting is large and the computational benefits of the $k$-d tree diminish as sequence length increases. Thus it was determined that sequences approximately 20% of the total length are optimal for this method.

7.3 Moving tests

Once the system has been confirmed to work statically, moving tests are conducted. This involves the introduction of the Kalman filter.
7.3.1 The data-sets

As described in chapter 3, this dissertation does not seek to introduce novel image processing techniques, and thus test cases in which the horizon is clearly visible were selected. At the LTI test track facility, the far-field horizon is most visible at the north end of the site. Test sets were collected from the MSS (chapter 2) traveling 35 - 40 mph over approximately \(\frac{1}{2}\) of a mile (for sample images, see figure 7.6). The test and map sets were both taken on the same day within minutes of each other. Passes were completed by a vehicle moving in the same direction; counter-clockwise around the track. All images were collected by the Occam camera (monocular images were collected by subscribing only to one of the 10 cameras).

The waterfall horizon extraction process was observed to be the most robust of the three proposed techniques. In the collected data there were no overhead obstructions, the weather was clear, and there existed no occlusions of the mapped horizon. For these reasons, it was entirely appropriate to apply the most basic of extraction processes.

Figure 7.6. Samples of the images and extracted horizons for the moving test sets.
7.3.2 Map-to-map correspondence

For these tests there were two maps of the horizon as a function of pose: the reference map and the query map. The reference map is the a priori knowledge of the environment, and the query map is the sensor output for the test data. Maps were created by extracting the horizons using the waterfall technique and storing them in $x$ and $y$ arrays. The reference arrays were stored alongside the 6-DoF ground truth pose as collected by the Novatel INS. The query sets were stored with only 3-DoF position and yaw. Yaw was included because for a monocular camera image the orientation of the camera must be known to ensure that the query camera and map camera were both pointing in the same direction. If panoramic images were to be used, yaw would no longer be necessary in the query set.

Correspondence between the query map and the reference map was determined using a 4 dimensional $k$-d tree search: latitude, longitude, altitude, and yaw. This determined the closest horizon contour in the reference map for a given query.

7.3.3 Simulated IMU

As described in section 6.4.1, the measurements for this system are roll $\phi$, pitch $\theta$, yaw $\psi$ as measured visually, and angular rates $p$, $q$, and $r$ as measured by a traditional IMU. However, the high fidelity INS available for this study does not directly output $p$, $q$, and $r$ - instead only output roll, pitch, and yaw. To avoid introducing noise with unknown characteristics, this INS output was used to calculated a theoretical value for $p$, $q$, and $r$ using the Euler equations (equations 6.10, 6.11, and 6.12). Noise could then be added in a controlled manner to the ‘measurements’ of $p$, $q$, and $r$ so its effect on the final system could be studied.

The magnitude of the noise on the IMU was set to $5 \times 10^{-4} \text{rad s}^{-1}$. This value came from current automotive standards, as referenced in Groves’ book on inertial sensing [56]. The resulting error in ‘simulated IMU’ measurements are presented in figure 7.7.

7.3.4 Results

For the moving tests, a query set of 1100 images was compared to a map database consisting of 4790 images. The query set represented a half mile traversal on the
north side of the LTI test track and the map set consisted of images taken from a mile long lap of the entire track. The Canny waterfall method was utilized to extract horizons from both the map and query sets. Maps were stored as the raw Canny output contour referenced with the with the full 6-DoF pose output from the Novatel system. The horizons were redefined in terms of feature sets consisting of peaks and valleys (local extrema), with rotational invariance achieved by resolving the contours to the $x$-axis using a first order fit that provided an initial estimate of relative roll. Sequences of these features were compared to determine correspondence. Once correspondence was achieved, linear least squares regression was applied to determine the full transformation matrix relating the two contours. From this roll, pitch, and yaw were determined. This process is illustrated in figure 7.8.

These visual estimates of relative orientation were then combined with a simulated automotive-grade IMU in a Kalman filter to provide the third and final estimate of relative orientation. Based on results from the static tests above, initial trust in the visual measurements of relative orientation were set to 5°s. Results
Figure 7.8. The complete process as applied with example figures taken at each step. From the moving tests are presented in figure 7.9. All orientation values are relative to the initial position of the vehicle.

7.3.5 Error analysis

Error results are presented in figure 7.10. Error is measured directly in terms of deviation from truth due to the percent error formula’s lack of fidelity around truth values of zero. Mathematically, $Error_x = x_{measured} - x_{truth}$. The results in figure
Figure 7.9. Estimates of orientation from the three techniques described in this document: first order fitting, linear least squares regression (LLSR), and Kalman filtering.

7.10 demonstrates increasing accuracy with the subsequent steps of the process: the first order fit, the linear least squares regression, and finally the Kalman filter.

Of the 110 images tested, approximately 10% were observed to have error of greater than $5^\circ$ in roll, pitch, or yaw (figure 7.11). The errors were normally distributed about a zero mean and show no indication of bias or bi-modal distribution.

A statistical representation of the results is presented in the following tables (tables 7.1, 7.2, and 7.3). While the average error wavers, the progression clearly shows a ‘tightening’ of the standard deviation in measurement. This demonstrates that alone without an inertial measurement unit the horizon extraction methods are capable of determining attitude within $5^\circ$'s certainty. With the addition of the Kalman filter utilizing an automotive grade IMU, accuracy tightens to $3^\circ$'s.
Upon investigation, the majority of these errors were attributed to the mis-identification of features. As can be seen in figure 7.12, the map and query horizon contours are visually similar but different points are identified as features. Feature mis-identification is largely due to pixel level noise that causes the appearance of erroneous peaks and valleys. In the specific examples shown in figure 7.12, there is a sequence in the middle that is represented by more features in the map than the query set. This causes sequence matching to be incorrect, which causes errors
Figure 7.11. Histograms of the errors presented in figure 7.10.
Table 7.2. Error statistics for estimates of pitch

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean error</th>
<th>Standard deviation of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Order</td>
<td>0.74°</td>
<td>2.60°</td>
</tr>
<tr>
<td>LLSR</td>
<td>1.54°</td>
<td>2.35°</td>
</tr>
<tr>
<td>Kalman Filter</td>
<td>0.70°</td>
<td>0.86°</td>
</tr>
</tbody>
</table>

Table 7.3. Error statistics for estimates of yaw

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean error</th>
<th>Standard deviation of error</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Order</td>
<td>1.12°</td>
<td>12.92°</td>
</tr>
<tr>
<td>LLSR</td>
<td>1.28°</td>
<td>2.12°</td>
</tr>
<tr>
<td>Kalman Filter</td>
<td>0.82°</td>
<td>1.03°</td>
</tr>
</tbody>
</table>

These results also demonstrate that when linear least squares fails, it does so dramatically and errors are significant. This knowledge can be accounted for by implementing outlier rejection and thus ‘smoothing’ the results from linear least squares. Since the majority of measurements work, these erroneous measurements can be ignored.

7.4 Conclusions

This chapter demonstrates the ability of the procedures outlined in the rest of this dissertation to achieve orientation accuracy on the order of an automotive IMU. This process consists of three stages of attitude estimation: first a simple first order fit rectified to the $x$-axis. Second, linear least squares regression is used to determine the 1-d rotation matrix that represents the roll angle between the two horizons. In this step, translation error is also determined and utilized to calculate pitch and yaw. Finally, measurement of attitude from an IMU and the visual measurements are processed by a Kalman filter to determine the best estimate of states roll $\phi$, pitch $\theta$, and yaw $\psi$.

While this work was not implemented in real-time, it does present significantly more computationally efficient mechanisms than previously utilized in the state of the art. One of the primary computational savings of this procedure is the $k$ dimensional correspondence described in section 4.3.2.2. Instead of matching the entire horizon line or even the entire feature set, a subset of the feature set is
Errors are largely attributed to mis-identification of features. The algorithm is also more efficient because it is able to obtain rotationally invariant features using a simple first order fit compared to the more burdensome SIFT and SURF techniques. This

Figure 7.12. Errors are largely attributed to mis-identification of features.
methodology lends itself well to real-time implementation, which follows naturally from this dissertation work.
Chapter 8

Conclusions

8.1 Summary

This dissertation has presented a methodology for the visual estimation of differential attitude between maps using horizon contours. First, sample images from a variety of test locations were used to determine the potential challenges that would be encountered while extracting the horizon line from an image. These challenges were identified to be overhead occlusions, weather, lighting conditions, and dynamic occlusions. Strategies for overcoming these obstacles have been identified, and design requirements have been developed based on the difficulties associated with these problems. As a result, this work focuses on rural environments in daylight conditions with clear skies.

Next, feature definition was reviewed and the local extrema (minima and maxima) were determined to be an ideal feature for the characterization of horizons. Rotational invariance was achieved by fitting a first order line to the horizon contour and resolving it to the x-axis. This method has limitations, such as ambiguity at rotations greater than 90°s; however, it was demonstrated to be consistent enough to provide rotational invariance sufficient for typical road vehicle localization scenarios.

Once features were defined, several methods of achieving correspondence were explored. An initial procedure called for probabilistic matching using a recursive Bayes filter. This method was proven to work, but was slow and sensitive to noise. It was determined that the k-d tree matching method was a quicker and more
robust alternative. This was implemented by breaking feature sets into sequences of length $k$ and finding matches using a $k$ dimensional tree. This length $k$ was determined experimentally by observing the number of correct matches as a function of changing sequence length.

The relative orientation between feature sets was then determined in a two-step process. First the rotation matrix that resolves two contours was determined using linear least squares regression (LLSR). This $2 \times 2$ rotation matrix provides the value of roll $\phi$. With $\phi$ applied, translation errors $t_x$ and $t_y$ are determined. Pitch $\theta$ and yaw $\psi$ can be calculated from translation using the visual angle, which is a function of camera parameters.

Finally, relative orientation as determined above is processed with noisy IMU measurements in a Kalman filter to determine the best estimate of attitude. The Kalman filter was designed for states roll, pitch, and yaw ($x = [\phi, \theta, \psi]$), and inputs of angular rate about each axis as measured by the IMU. These IMU measurements were simulated by back-calculating angular rate from ground truth attitude. This provided the flexibility to experiment with different noise parameters on the IMU and observe the resulting Kalman estimate.

To collect the data necessary to test these processes, a mobile mapping system (MMS) was designed, developed, and constructed. This MMS enabled the collection of data-sets consisting of visual measurements of the environment (images) as well as range information (LiDAR) fused with defense-grade measurements of pose. A map set and a query set consisting of forward-facing monocular camera were compared to each other for this study. Each subsequent step of this process demonstrated increasing accuracy. When combined with the Kalman filter, results were shown to match or exceed those of a standard automotive IMU.

This work answered the research question ‘what is the sparsest representation of an environment I can use to achieve a visual estimate of attitude?’ This sparsity is introduced first by representing a horizon as a set of features, then representing the features as a series of feature sequences, which dramatically reduces the number of computations being performed.
8.2 Future work

Throughout the body of this work a number of important and interesting concepts were acknowledged, but deferred to ‘future work’. This section lists and discusses the primary extensions of this work and resurrects those ideas worthy of additional research and development.

1. Real-time implementation. As discussed in chapter 7, this work lends itself well to real-time implementation due to the relative computational simplicity of the processes utilized.

2. The use of hyper-spectral images for horizon detection. This extension would help solve many of the problems of the horizon extraction process, primarily those of weather and lighting conditions. Depending on the spectrum chosen, this could also alleviate the challenge posed by many occlusions.

3. Use of SRTM, DEM, or other commercially available or open-source maps. The use of maps collected by the MMS is attractive because it offers a horizon contour that will include all major landmarks. This helps to minimize the problem of occlusions and alleviates computational burden by avoiding the rendering or pinhole projection process necessary for a DEM or SRTM map. The problem with this method is that the maps may be subject to other forms of disturbances that cannot be easily avoided. Maps of busy highways may constantly contain other vehicles that will introduce spurious feature and partially or totally obliterate the horizon. Maps collected in poor lighting conditions may be disrupted by glare or speckling. Furthermore there is no high-fidelity publicly available map that consists of images, either panoramic or monocular, taken along roadways. Methods involving both rendering of a virtual map and pinhole projection of elevation data should be explored to mitigate these issues.

4. Additional image processing techniques. One of the major challenges encountered in this work has been mis-registration of horizons due either to mis-identification of the horizon or occlusions that introduce spurious features into the contours. Future work could add robustness here by matching
Figure 8.1. An example of an environment in which multiple ridgelines can be used simultaneously to mitigate extraction errors and occlusions.

multiple contours per image instead of only one. Consider, for example, a mountain range in which multiple ridge-lines are present (figure 8.1). Instead of only utilizing the horizon formed by the sky and the ground, it is possible to all ridge-lines, including those in the foreground. It is believed that such an addition would increase system robustness by adding more potentially high quality features to the feature set to offset the effects of any disturbances.

5. Use of near-field features. Near-field features are advantageous because they are more easily recognized with a higher resolution and could be used when no view of the horizon is available. While near-field features violate many of the foundational assumptions of this work, the core implementation would remain the same with some alterations to the mathematics.

8.3 Conclusions

The use of visual estimates of orientation using the horizon is not a new concept. The methodology exists even in nature, as was demonstrated in Stone’s study of
ant-vision [32]. This work stands apart not by the novelty of the steps outlined, but of the minimalistic techniques of the procedure as a whole. The theme of this work is determining what the minimum representation of an environment is to achieve valid correspondence and thus an accurate visual estimate of attitude. This work presents a series of relatively simple processes from a variety of fields elegantly implemented in concert to achieve results on par with the state of the art.
Bibliography


[65] PATON, M., F. POMERLEAU, and T. D. BARFOOT “In the Dead of Winter: Challenging Vision-Based Path Following in Extreme Conditions.”


Vita

Richard Patrick Proffitt

Patrick Proffitt was born in Richmond, Virginia in 1992. He attended the Virginia Commonwealth University and graduated with a Bachelors of Science in Mechanical Engineering in 2011. Patrick has worked for the Department of the Army’s Night Vision and Electronic Sensors Directorate (NVESD) since 2009. Through his work there, he received the Science, Mathematics, And Research Technology (SMART) Scholarship in 2011 and used it to attend Penn State University. At Penn State he worked with Dr. Sean Brennan in the Intelligent Vehicles and Systems Group in the Department of Mechanical Engineering from 2011 - 2016. Patrick was in charge of designing and constructing Penn State’s mobile mapping van. With the group, Patrick earned his M.S. in 2015 and his Ph.D. in 2016.