Robust Localization in 3D Prior Maps for Autonomous Driving

by

Ryan W. Wolcott

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2016

Doctoral Committee:

Associate Professor Ryan M. Eustice, Chair
Assistant Professor Matthew Johnson-Roberson
Professor Benjamin Kuipers
Associate Professor Edwin Olson

2016

ACKNOWLEDGMENTS

This has been quite a journey. With ups and downs, countless deadlines and just as many last minutes, and I am so grateful for the support that I've had throughout it all. Without my advisors, family, and friends, this work would not have been possible. Thank you to all.

First, I'd like to thank my advisor Prof. Ryan Eustice. Your watch took me from an undergrad that was filled with curiosity and presented me with numerous opportunities to satisfy this curiosity. From UM::Autonomy to the Next Generation Vehicle (NGV) project, all tasks seemed insurmountable at the start, though your steady guidance brought much clarity and made things a reality. I'm certain you spent most deadlines wondering "are you gonna make it?", as I could never get myself out of the code to finish the paper. Thank you for your encouragement, patience, and wisdom—without it I certainly would not have made it. While I was grateful for the extra 3 hours afforded by the Pacific Time submission deadlines, I was even more grateful to you for staying up late on the other end of Gchat.

Next, I'd like to thank my other committee members, Prof. Ed Olson, Prof. Matt Johnson-Roberson, and Prof. Ben Kuipers. I am grateful for our discussion and your suggestions throughout this thesis. I'd especially like to thank Ed for his new perspectives on problems and willingness to get into the nitty-gritty discussing source code and implementations. I am thankful for your and Ryan's leadership, along with Jim McBride at Ford, throughout the NGV project.

A big thank you to all PeRL members that have come and gone over the years: Ayoung, Gaurav, Nick, Jeff, Paul, Steve, Schuyler, Jie, GT, Arash, Enric, Alex, Sparki, Vittorio, Josh, Vozar, Derrick, Gonzalo, Carl, and Sud (whew). I have learned so much from all of you. The laughter, idea bouncing, and random discussions have kept me going. I'll always remember "social hour" where I hung out in the lab and distracted you all because I got bored or frustrated with my work. Further thanks to Joho and Andrew from April—I look back fondly on the countless hours spent working in the garage and the late nights driving around parking lots.

Now on to my family. Thanks to Mom, Dad, and all my immediate family for always encouraging me throughout my studies. I certainly wouldn't have made it this far without you, so thank you. And yes, Mom, my paper is going well. To my loving wife Bethany, I am thankful for your support that has been simply indescribable. Through every deadline that

I was sure I was going to miss, your encouragement kept me going until the final minute and I will always be grateful for having you by my side. I know my "just five more minutes" are usually off by an hour, but your patience throughout has been a blessing. Finally, my sweet baby Sarah—your arrival halfway through this thesis brought me a remarkable amount of joy that encouraged and motivated me to complete this work. Thank you to both of you for making life always wonderful; I look forward to the many experiences ahead.

Finally and most importantly, I'd like to thank God for making this all possible. Every time I thought this thesis was impossible (which was often), I found peace in Proverbs 3:5-6 that granted me trust through the many trials of graduate school.

Funding

This work was supported in part by Ford Motor Company under the Ford-UM Alliance and in part by The SMART Scholarship for Service Program by the Department of Defense.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF ACRONYMS

2D two-dimensional

3D three-dimensional

AUV autonomous underwater vehicle

BA bundle adjustment

BEV Bird's Eye View

CPU central processing unit

CNN convolutional neural network

DOF degree of freedom

EIF extended information filter

EKF extended Kalman filter

EM expectation-maximization

GICP generalized iterative closest point

GPS global positioning system

GPU graphics processing unit

GLSL OpenGL Shading Language

ICP iterative closest point

IMU inertial measurement unit

INS inertial navigation system

LIDAR light detection and ranging

MAP maximum *a posteriori*

MCL Monte Carlo localization

MI mutual information

MLE maximum likelihood estimate

MLS multi-level surface

MMSE minimum mean squared error

MRF Markov random field

MAD median absolute deviation

NMI normalized mutual information

NDT normal distributions transform

NGV Next Generation Vehicle

NID normalized information distance

NIS normalized innovation squared

PCCS pose-constrained correspondence search

RMS root mean squared

SIFT scale-invariant feature transform

SLAM simultaneous localization and mapping

ABSTRACT

In order to navigate autonomously, many self-driving vehicles require precise localization within an *a priori* known map that is annotated with exact lane locations, traffic signs, and additional metadata that govern the rules of the road. This approach transforms the extremely difficult and unpredictable task of online perception into a more structured localization problem—where exact localization in these maps provides the autonomous agent a wealth of knowledge for safe navigation.

This thesis presents several novel localization algorithms that leverage a high-fidelity three-dimensional (3D) prior map that together provide a robust and reliable framework for vehicle localization. First, we present a generic probabilistic method for localizing an autonomous vehicle equipped with a 3D light detection and ranging (LIDAR) scanner. This proposed algorithm models the world as a mixture of several Gaussians, characterizing the z-height and reflectivity distribution of the environment—which we rasterize to facilitate fast and exact multiresolution inference. Second, we propose a visual localization strategy that replaces the expensive 3D LIDAR scanners with significantly cheaper, commodity cameras. In doing so, we exploit a graphics processing unit to generate synthetic views of our belief environment, resulting in a localization solution that achieves a similar order of magnitude error rate with a sensor that is several orders of magnitude cheaper. Finally, we propose a visual obstacle detection algorithm that leverages knowledge of our high-fidelity prior maps in its obstacle prediction model. This not only provides obstacle awareness at high rates for vehicle navigation, but also improves our visual localization quality as we are cognizant of static and non-static regions of the environment. All of these proposed algorithms are demonstrated to be real-time solutions for our self-driving car.

CHAPTER 1

Introduction

Car accidents claim the lives of roughly 1.24 million people per year (World Health Organization, 2013). In the United States alone, deaths top thirty thousand with over 6 million crashes per year (Maddox, 2012). Despite these devastating statistics, the public has grown to accept traffic accidents because vehicles provide a vital piece of technology for our culture and are relatively safe per capita, where average drivers can expect to be in a car accident only once every ten years. Fortunately, fatality rates have declined over the past several decades due to advancements in passive safety and active safety measures. However, 93% of these remaining crashes are a result of some human error—a byproduct of increased mobile phone usage and other internal vehicle distractions.

Over the past several years, many have looked toward robotics as a solution to reduce the number of traffic accidents and enable those that are unable to drive for various medical conditions—an attractive option as it removes the risk of distracted drivers from roadways. The growth in mobile robotics during this time has made this a reality, allowing self-driving vehicles to safely navigate roadways congested with other human-driven vehicles. Systems such as the Google driverless car have successfully driven hundreds of thousands of miles without user intervention (Thrun, 2010), and several car manufacturers have begun looking into commercialization of such technology.

A common approach to self-driving cars is to use detailed prior maps that are annotated with precise lane locations, traffic signs, and other metadata that govern the rules of the road. These maps are generated offline, which allows the use of complex algorithms that are not necessarily "real-time" to be used by the operating self-driving car. The use of prior maps allows researchers to turn some of the difficult perception tasks into a localization problem. Localization in the robotics community is a mature research area that yields a bounded problem given the well structured environment an automobile operates in. The needs of the car are a localization system that can robustly track and verify that the perceived world matches its prior belief, as downstream components of the autonomous car (e.g., planning and control) rely on precise localization for decision making. Therefore, localization robustness is

critical as it is a subsystem that cannot fail or the online autonomous platform would no longer be able to operate.

This thesis focuses on extending the state-of-the-art to increase robustness of localization for autonomous cars. We propose a fast and efficient three-dimensional (3D) light detection and ranging (LIDAR) localization algorithm that can jointly reason over prior structure and appearance. Additionally, we propose a visual localization algorithm that is able to achieve a similar order of magnitude error rate as its LIDAR counterpart using an optical monocular camera that is an order of magnitude cheaper. Finally, we leverage these detailed prior maps in our proposed visual obstacle detection system, which in turn, can be used to improve the performance of our visual localization system. These parts combined yield a robust, multi-modal localization system for self-driving cars.

1.1 Simultaneous Localization and Mapping

Robots operating in an *a priori* unknown environment use a class of algorithms referred to as simultaneous localization and mapping (SLAM) to answer the questions "where am I?" and "what does my environment look like?" When attempting to solve the full SLAM problem, we are interested in concurrently estimating the robot's full trajectory and the map it is operating in to answer these two questions. In this section, we consider solving the SLAM problem in a metric framework as it is more natural for our application, though there are many alternative solutions that consider SLAM in a topological (Angeli et al., 2009; Choset and Nagatani, 2001) or a hybrid metric-topological formulation (Beeson, Modayil, and Kuipers, 2009; Blanco, Fernández-Madrigal, and Gonzalez, 2008).

We consider the robot's trajectory as a set of poses, $\mathbf{X} = \{x_i\}_{i=0}^{n}$, with typically $x_i \in \mathbb{R}^6$, though these vectors can be augmented to capture other time-dependent state elements. To solve the SLAM problem, we seek to find the optimal alignment of our trajectory and map, $\mathbf{M}$, given noisy odometry measurements $\mathbf{U} = \{u_i\}_{i=1}^{n}$ and other sensory measurements (typically of the map) $\mathbf{Z} = \{z_k\}_{k=1}^{p}$. Note that in various applications, this map can be made up of point landmarks, occupancy grids, etc., or the map can be circumvented altogether using sensory measurements directly to establish constraints between two poses (this is known as pose-graph SLAM); this full SLAM problem is depicted in Fig. 1.1.

Formally, we seek to find the maximum *a posteriori* (MAP) estimate of our trajectory and map by evaluating

$$\mathbf{X}^*, \mathbf{M}^* = \arg\max_{\mathbf{X},\mathbf{M}} p\left(\mathbf{X}, \mathbf{M} | \mathbf{U}, \mathbf{Z}\right). \tag{1.1}$$

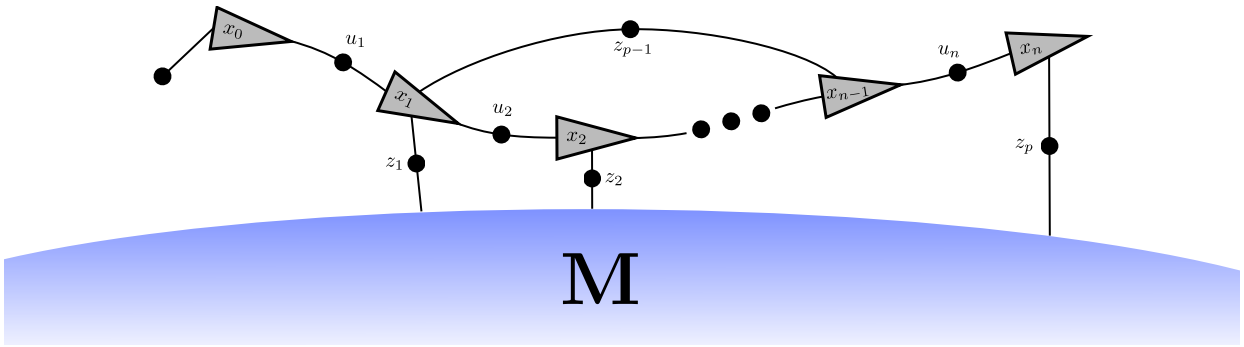


Figure 1.1: Sample factor graph for solving the full SLAM. In solving the SLAM problem, we seek the optimal arrangement of pose nodes, $\{x_i\}$, and map configuration, $\mathbf{M}$, given noisy sensor measurements in the form of vehicle odometry, $\{u_i\}$, and observations $\{z_k\}$. Note that the map representation here is a generic representation that can encompass point-based landmarks, occupancy grids, and other map representations that each have their own estimation requirements within the full SLAM problem.

By modeling our noisy odometry and sensory measurements as Gaussian random variables,

$$x_i = f_i(x_{i-1}, u_i) + w_i, \tag{1.2}$$

$$z_k = h_k(x_{i_k}, \mathbf{M}) + v_k, \tag{1.3}$$

where $f_i(\,\cdot\,)$ and $h_k(\,\cdot\,)$ are our process and observation models, respectively, and $w_i \sim \mathcal{N}(0, \Sigma_i)$ and $v_k \sim \mathcal{N}(0, \Sigma_k)$, we can represent the joint distribution as

$$p(\mathbf{X}, \mathbf{M}, \mathbf{U}, \mathbf{Z}) = p(x_0) \prod_{i=1}^{n} P(x_i | x_{i-1}, u_i) \prod_{k=1}^{p} P(z_k | x_{i_k}, \mathbf{M}) \tag{1.4}$$

$$\propto \prod_{i=1}^{n} e^{-\frac{1}{2}\|f_i(x_{i-1}, u_i) - x_i\|^2_{\Sigma_i}} \prod_{k=1}^{p} e^{-\frac{1}{2}\|h_k(x_{i_k}, \mathbf{M}) - z_k\|^2_{\Sigma_k}}. \tag{1.5}$$

Thus, to solve the SLAM problem of Eq. 1.1, we can find the MAP estimate by minimizing the negative log of the joint probability:

$$\mathbf{X}^*, \mathbf{M}^* = \arg\max_{\mathbf{X},\mathbf{M}} p(\mathbf{X}, \mathbf{M} | \mathbf{U}, \mathbf{Z}) = \arg\max_{\mathbf{X},\mathbf{M}} p(\mathbf{X}, \mathbf{M}, \mathbf{U}, \mathbf{Z}) \tag{1.6}$$

$$= \arg\min_{\mathbf{X},\mathbf{M}} -\log p(\mathbf{X}, \mathbf{M}, \mathbf{U}, \mathbf{Z}) \tag{1.7}$$

$$= \arg\min_{\mathbf{X},\mathbf{M}} \left\{ \sum_{i=1}^{n} \|f_i(x_{i-1}, u_i) - x_i\|^2_{\Sigma_i} + \sum_{k=1}^{p} \|h_k(x_{i_k}, \mathbf{M}) - z_k\|^2_{\Sigma_k} \right\}. \tag{1.8}$$

1.1.0.1 Filtering Solutions

Filtering frameworks approach the SLAM problem from a recursive Bayesian estimation standpoint. With measurements under Gaussian noise, the extended Kalman filter (EKF) is the optimal minimum mean squared error (MMSE) estimator in which system nonlinearities are handled by linearizing the process (Eq. 1.2) and observation models (Eq. 1.3). Typically, these applications (Davison et al., 2007; Leonard and Durrant-Whyte, 1991a; Smith, Self, and Cheeseman, 1990) continuously marginalize out prior robot poses, thus solving for only the most recent robot pose and the map of its environment:

$$x_n^*, \mathbf{M}^* = \arg\max_{x_n, \mathbf{M}} p\left(x_n, \mathbf{M} | \mathbf{U}, \mathbf{Z}\right). \tag{1.9}$$

As the map, $\mathbf{M}$, continues to grow over time, the problem quickly becomes intractable to solve real-time as the EKF requires expensive matrix inversions on each recursive update.

Many researchers, including Bar-Shalom, Rong Li, and Kirubarajan (2001), have instead looked at the extended information filter (EIF) to overcome these computational limitations of the EKF. The EIF is the dual of the EKF, where state belief is parameterized in terms of the information vector and information matrix (inverse covariance matrix). Though the covariance matrix of the EKF is typically a dense matrix, Thrun et al. (2004) noted that this reparameterization results in a nearly sparse information matrix, where elements encode spatial relationships between landmarks. The authors then enforce a sparsification procedure yielding an exactly sparse representation that facilitates constant time updates and linear memory usage in number of landmarks.

Eustice, Singh, and Leonard (2006) noted that the explicit marginalization of prior robot poses (Eq. 1.9) is the reason for information matrix fill-in and sparsification leads to overconfident state estimates (Eustice, Walter, and Leonard, 2005). Instead, they propose to use a delayed state EIF that solves the full SLAM problem without marginalizing historic robot poses. This framework maintains exact sparsity in the information matrix, enabling very large scale filtering results.

1.1.0.2 Optimization Solutions

Ultimately, filtering methods suffer because they commit to a single linearization for process (Eq. 1.2) and observation models (Eq. 1.3) as they are measured, which can pose a problem as linearization errors compound. Viewing Eq. 1.8 as a nonlinear least squares problem has led the SLAM community to leverage sparse linear algebra techniques to solve the full SLAM problem. Further, storing the full nonlinear measurements allows for continuous relinearization of the measurements observed. Dellaert and Kaess (2006); Kaess, Ranganathan,

and Dellaert (2008); Kaess et al. (2012), Olson, Leonard, and Teller (2006b), Thrun and Montemerlo (2006), and Kummerle et al. (2011) have all considered solving SLAM using various optimization approaches.

1.1.0.3 Monte Carlo Solutions

The SLAM problem can alternatively be solved using sampling based approaches in which a finite set of *particles* can be used to model the posterior distribution (as opposed to using parametric Gaussian densities). Particles are then weighted and resampled as new measurements are processed.

Solving landmark-based SLAM directly this way would be intractable as the required number of particles to capture the full state space would be incredibly large. This led Montemerlo et al. (2002, 2003) to consider the Rao-Blackwellization of the SLAM problem whereby the posterior density can be factored as: $p(\mathbf{X}, \mathbf{M}|\mathbf{U}, \mathbf{Z}) = p(\mathbf{M}|\mathbf{X}, \mathbf{Z})\, p(\mathbf{X}|\mathbf{U}, \mathbf{Z})$. Thus, particles can be used to capture the statistics (i.e., $p(\mathbf{X}|\mathbf{U}, \mathbf{Z})$) of the robot, while each map element can be modeled as an independent Gaussian distribution conditioned on the robot.

1.2 Localization within Metric Maps

In many domains, a map can be obtained ahead of time reducing to the single question of "where am I in this map?" These maps can either be provided from human cartographers or built using SLAM during previous traversals of the world. This leads us to solving a simpler estimation problem than the full SLAM problem because our map is known *a priori*, resulting in the following estimation goal:

$$\mathbf{X}^* = \arg\max_{\mathbf{X}} p(\mathbf{X}|\mathbf{M}, \mathbf{U}, \mathbf{Z}). \tag{1.10}$$

Frequently, smoothing the full robot trajectory in this form is not done, and can simply be reduced to filtering over the most recent robot pose,

$$x_n^* = \arg\max_{x_n} p(x_n|\mathbf{M}, \mathbf{U}, \mathbf{Z}). \tag{1.11}$$

As in the previous section, this distribution can evolve using recursive Bayesian estimation as new measurements are observed. More succinctly, Bayes rule allows us to repeatedly estimate:

$$p\left(x_n|\mathbf{M},\mathbf{U},\mathbf{Z}\right) = \eta \underbrace{p\left(z_k|x_n,\mathbf{M}\right)}_{\text{likelihood}} \underbrace{p\left(x_n|\mathbf{M},\mathbf{U},z_{1:k-1}\right)}_{\text{prior}} \tag{1.12}$$

Thus as new measurements are made distributed according to the likelihood distribution, our prior belief can be updated.

The problem of localization falls into two distinct categories depending on the amount of prior information available: *local* and *global* localization (Fox, Burgard, and Thrun, 1999). In *local* localization, one is concerned with refining a pose belief given some prior information (thus, *local* uncertainty) and a lack of measurements could cause the robot belief to diverge and lose track in the map. However, in *global* localization, there is no prior belief and the robot must equally weight its likelihood of being anywhere within the map (thus, *global* uncertainty). Even more difficult is the *kidnapped-robot* problem proposed by Engelson (1994), in which the robot has a confident prior on its location and is miraculously moved to another location in the environment without notification. Research looking at solving this problem must be cognizant of this possibility and realize when sensor data is suddenly incoherent with the world, indicating a necessity for re-global localization.

For applications that require global localization, many researchers will only apply global localization techniques until the posterior confidence exceeds a predetermined threshold, consequently the robot will transition to local localization strategies, as is done by Ozog and Eustice (2014) while visual localizing aside a ship hull. However, these two phases are not required to be two distinctly different estimation techniques.

1.2.1 Kalman Filter Localization

Kalman filtering has been successfully used to track the position of a robot through a prior map, but is typically not successful for global localization as it only captures a unimodal belief of the robot pose. Gutmann, Weigel, and Nebel (2001) noted that the Kalman filter was sufficient in their application for global localization, though only because of their relatively small environment; in larger environments, it would be impossible to consider the many permutations necessary for global localization. However, when considering local localization, the Kalman filter can be an ideal choice for fusing measurements and tracking pose (Chaves, Wolcott, and Eustice, 2015; Negenborn, 2003). Further, Kalman filtering has been a common approach for localizing in a map consisting of point beacons (Leonard and Durrant-Whyte, 1991b; Olson, Leonard, and Teller, 2006a).

1.2.2 Monte Carlo Localization

Monte Carlo localization (MCL) has become a popular approach to the global localization problem as it is trivial to implement in software, well suited to handle multi-modal beliefs, and yields solutions that quickly converge to the global pose; the techniques were introduced in a sequence of works by Dellaert et al. (1999) and Fox et al. (1999). MCL works by randomly sampling *particles* through the map according to a pose prior that can be fully uninformed. These particles are then weighted and resampled according to coherence of sensory measurements to the prior map. Then, each particle samples and propagates according to the platform's odometry model, and the observation updates iterate. The work was then altered by Thrun et al. (2001), where the proposal distribution was altered by instead sampling from the observation model, then weighted according to the odometry model.

With the emergence of low-cost depth sensors, such as the Microsoft Kinect, there is an abundance of new research being published looking at localizing such sensors in prior maps. Many of these methods rely on an iterative alignment to a 3D prior map in point cloud space, as proposed by Newcombe et al. (2011) and Cunha et al. (2011). More recently, Fallon, Johannsson, and Leonard (2012) use a Monte Carlo scheme for estimating a Microsoft Kinect's posterior pose, exploiting OpenGL for generating synthetic depth-views. Additionally, rather than taking the common approach to evaluate likelihoods by evaluating point-to-plane distances, as is common in iterative closest point (ICP) cost functions, they instead create a generative likelihood function—comparing against synthetic particle filter views.

1.2.3 Self-driving Car Localization

Many self-driving car localization approaches rely on the methodology of localization within a prior map, where doing so provides a wealth of knowledge regarding the operating environment of the vehicle, including lanes, traffic signals, and other rules of the road. Typically, these approaches are a *local* localization task, where the objective is to track the position of the vehicle through the prior map or previous robot experiences; with external references available such as global positioning system (GPS), *global* localization strategies are usually not necessary.

Algorithms proposed by Levinson et al., which utilize 3D LIDAR scanners and an integrated GPS/INS system, have been considered the benchmark method for autonomous car localization for many years. In their work, they extract the ground plane points from 3D LIDAR scanners and build an orthographic map of ground-plane intensities (see Fig. 1.2 for an example map). They then use online measurements to localize their vehicle using either a particle filter (Levinson, Montemerlo, and Thrun, 2007) or histogram filter (Levinson

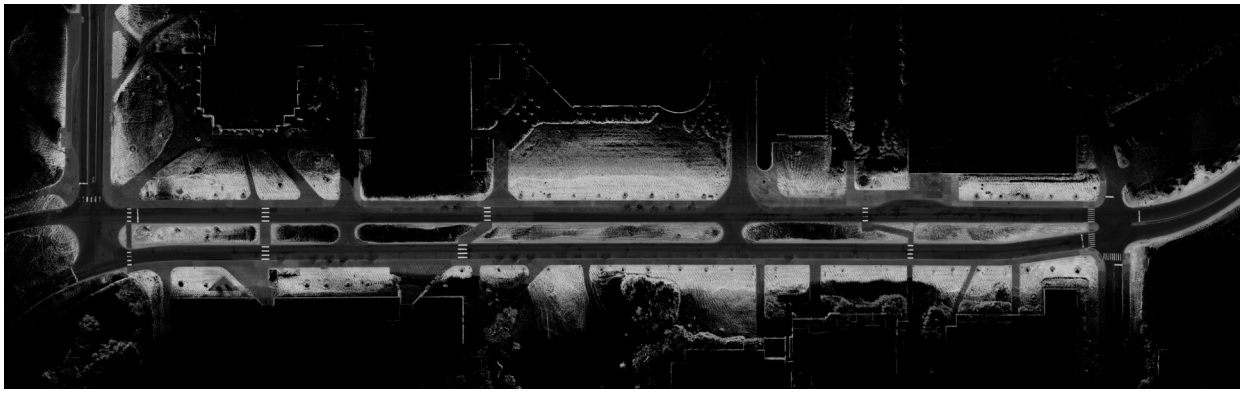


Figure 1.2: Sample prior map used for LIDAR reflectivity-based localization; commonly used by autonomous cars. This map represents a portion of Bonisteel Ave. on the University of Michigan's North Campus.

and Thrun, 2010). Further, for increased robustness, they consider the variance of this orthographic prior map so that their Bayesian inference is able to appropriately weight more confident measurements.

Napier and Newman (2012) extended a similar orthographic ground-image localization technique to use a stereo camera pair directed at the ground. Their work relaxes the requirement of highly expensive GPS/INS solutions, instead using state-of-the-art visual odometry to augment their localization. This method was able to achieve centimeter-level precision by maximizing mutual information (MI) between camera measurements and the expected map view.

Finally, Stewart and Newman (2012) present a solution called *LAPS*, which was inspired by the localization side of Newcombe, Lovegrove, and Davison (2011)'s DTAM (Dense Tracking and Mapping). Given their prior map, they project known scene points into a pair of estimated camera frames. They then compute the normalized information distance (NID) between the appearance of the projection in the two frames. This process is then repeated until overall NID is minimized.

1.3 Visual SLAM

1.3.1 Feature-based Methods

There are two distinct methods for feature-based visual SLAM: filtering frameworks and, more recently, keyframe-based systems. Both methodologies use unique, repeatable image features, such as scale-invariant feature transform (SIFT) features proposed by Lowe (2004), to establish constraints between the current pose and the underlying map. Filtering frameworks

continuously marginalize out prior poses, summarizing all accumulated information in a probability distribution over the current pose and belief map. On the contrary, keyframe-based approaches store keyframes in a pose-graph of the full SLAM problem. Localization is then typically done by using two-view bundle adjustment to minimize the reprojection error against keyframes in the graph.

The seminal filtering approach to monocular SLAM is that of Davison et al. (2007). In this work, features are modeled as landmarks in an EKF estimation problem. While the approach works well in office settings, it has the similar issue of other SLAM problems in that it does not scale well. As with the SLAM community, computer vision faced the similar question of "Why filter?" (Strasdat, Montiel, and Davison, 2012), instead proposing to solve the full SLAM problem using bundle adjustment (BA).

As an alternative to filtering, more modern efforts include a distribution of keyframes, typically connected by a pose-graph. Localization is done by performing a two-view bundle adjustment between the current image and a similar keyframe in the map—establishing a 5-degree of freedom (DOF) pose constraint that minimizes reprojection error. To keep localization tractable, these methods rely on a place recognition front end, such as FAB-MAP (Cummins and Newman, 2008), to efficiently propose link hypotheses between keyframes, as bundle adjustment can be an expensive task. A primary benefit of keyframe-based visual SLAM over filtering is its scalability to larger environments.

Keyframe-based pose-graph SLAM has been frequently used from localizing off-road vehicles as done by Konolige and Agrawal (2008) to autonomous underwater vehicles (AUVs) as done by Eustice (2005) and Kim and Eustice (2013). The PTAM framework, proposed by Klein and Murray (2007), uses the keyframe approach for managing an underlying map, which they then perform local and global bundle adjustment over to generate feature landmarks for localization. They then treat localization as a separate task, assuming a static map.

All of these visual SLAM strategies rely on co-observing robust image features; however, these features are often *not* time invariant and vary with time of day and weather conditions. To circumvent this issue, Churchill and Newman (2012) store sequences of different camera views called *experiences*, and Konolige and Bowman (2009) cluster views into candidate *exemplars*. These approaches of storing all possible views from a given location further complicates the localization problem because it provides even more keyframes to compare against. Carlevaris-Bianco and Eustice (2012) present a possible solution to this problem by using a Chow-Liu tree to model the temporal relationship between views, providing a way to predict similar views based on the current image sequence.

1.3.2 Direct Methods

Recent approaches consider direct methods where the underlying belief is that the world is made up of planar patches that exhibit brightness constancy across successive images. This approach allows for direct use of image intensity values for image alignment, as developed by Irani and Anandan (2000), eliminating the need to have robust image feature detectors/descriptors and the explicit need to perform data association.

Early direct visual SLAM systems tracked locally planar patches in a scene. For example Molton, Davison, and Reid (2004) and Silveira, Malis, and Rives (2008) demonstrated successful results, with the latter utlizing a novel method for handling illumination changes. Newcombe, Lovegrove, and Davison (2011) proposed DTAM, which estimates fully dense depth maps using a monocular camera stream by minimizing a global energy function with spatial regulurization.

Direct methods have since been applied to visual odometry (Forster, Pizzoli, and Scaramuzza, 2014), dense monocular reconstruction (Pizzoli, Forster, and Scaramuzza, 2014), and the full SLAM problem including novel loop closure methods (Engel, Sturm, and Cremers, 2013; Engel, Schöps, and Cremers, 2014).

1.4 Visual Obstacle Detection

Research to date considering visual obstacle detection using a monocular camera can be grouped into three primary categories. The first is that of framing the problem as a tracking task in which image pixels are tracked over time with optical flow. Using various approaches, this signal can be used to derive and segment out obstacles from the temporal image flow. The second category is quite similar to the first by tracking sparse feature sets over time to triangulate a local depth map to perform obstacle detection over. Finally, the task can also be approached as an image-space segmentation problem.

1.4.1 Obstacle Detection using Optical Flow

1.4.1.1 Optical Flow Methods

The goal of optical flow extraction is to estimate the projection of the 3D scene flow onto a monocular camera's imaging surface. This scene flow arises from relative motion between objects in the scene and the camera. Estimating this two-dimensional (2D) motion field in the image frame can be an especially difficult task because, from the view of a 2D image, one can only reason about the apparent motion of brightness patterns in an image. The apparent

motion of brightness patterns alone does not always reflect the true motion field; for example, a fixed, rotating object may appear to be translating from the camera frame (e.g., the barber pole illusion). Similarly, in a uniform, textureless scene, there simply isn't enough brightness variance to measure optical flow despite the fact that the scene is under motion.

Many optical flow estimation algorithms rely on the assumption of the brightness constancy constraint. This assumption is that the illumination of a 3D scene point remains constant over a short time interval. More formally,

$$I(x + \Delta x, y + \Delta y, t + \Delta t) \approx I(x, y, t), \tag{1.13}$$

where $I(x, y, t)$ is the intensity of the image at pixel location $\langle x, y \rangle$ and time t. Expressing this constraint as the time derivative of a scene point's intensity being *zero*, we can use the chain rule for differentiation to derive

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0. \tag{1.14}$$

Further, the terms in this derivation can be easily interpreted: $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ correspond to the image's spatial derivatives, which we'll denote I_x and I_y, and $\frac{\partial I}{\partial t}$ corresponds to the temporal image derivative, which we'll denote I_t. Also, $\frac{dx}{dt}$ and $\frac{dy}{dt}$ are the desired image plane flow measurements, u and v. Writing this out as

$$I_x u + I_y v + I_t = 0, \tag{1.15}$$

we see this is an underdetermined linear system, with two unknowns in one equation. Because of this, the brightness constancy assumption alone only allows us to calculate the "normal flow" that is normal to the image isophotes (i.e., in direction of the brightness gradient). To solve this, we must introduce different smoothness constraints in the flow field. As concisely pointed out by Barron, Fleet, and Beauchemin (1994), nearly all optical flow extraction methods come down to a 3-step process: (*i*) prefiltering to enhance signal-to-noise ratio in the image, (*ii*) extraction of measurements such as spatio-temporal derivatives or image features, and (*iii*) an optimization over these measurements, while assuming some smoothness in the flow field.

Historically, smoothness constraints come in two varieties: *global* and *local* approaches. One of the seminal pieces in optical flow research is the work of Horn and Schunck (1981), in which they enforced a *global* smoothing of the flow field. Using variational calculus, they

estimated flow with an iterative solution for minimizing a global error function,

$$E^2 = \int\int (I_x u + I_y v + I_t) + \alpha^2 \left(\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 + \left(\frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 \right) dxdy, \quad (1.16)$$

where the first term is the standard brightness constancy constraint and the second term is the smoothness regularization with weighting factor α^2. While their solution yields a desirable, dense flow field, errors are often present because the algorithm smooths sharp discontinuities in the flow field, which is common in real-world scenarios with varying depths of field.

As an alternative to global smoothing, Lucas and Kanade (1981) proposed a *local* smoothing approach that makes the assumption that the flow of a pixel is similar to its neighboring pixels. This assumption turns optical flow estimation into a weighted least squares problem at each pixel, pulling evidence from neighboring pixels. Further, to avoid ill-posed least squares problems, they only estimate optical flow at corners, where I_x and I_y gradients are strong—resulting in a sparse flow field.

Many of the latest state-of-the-art methods that are used employ a combination of *global* and *local* constraints. For example, work of Bruhn, Weickert, and Schnörr (2005) literally combines the two methods by replacing the individual brightness constancy term with a tensor that integrates brightness constancy constraints over a spatial window. Additionally, others expand on Horn and Schunck's method by adding an additional discontinuity-preserving smoothness constraint to the energy minimization (Brox and Malik, 2011; Brox et al., 2004; Brox, Bregler, and Malik, 2009), this term makes it possible to resolve large displacements in optical flow. Liu advanced this again by substituting the brightness constancy with a SIFT descriptor constancy term at each pixel (Liu, 2009).

A growing trend in the literature is to use a randomized approach called PatchMatch, originally proposed by Barnes et al. (2009, 2010) for image editing applications (e.g., image retargeting, completion, and reshuffling). The algorithm argues for finding, for each pixel, it's nearest neighbor in another image, where "neighbor" is defined in terms of the local patch around a pixel (hence, matching patches between images). However, finding the true nearest neighbor would be computationally intractable given the state space of an image. Instead, they demonstrate a randomized approach that iteratively *randomizes* beliefs and *propagates* good matches to neighboring pixels. Recent papers have looked at various ways of transitioning this ideology into an optical flow or stereo matching framework (Bao, Yang, and Jin, 2014; Chen et al., 2013). These methods have been shown to be more robust for large displacement optical flow.

1.4.1.2 Optical Flow Applications

Most early work using optical flow for obstacle detection had an intended use with a stationary camera for surveillance tasks. These works typically focus on segmenting the dynamic parts of a scene aside from an otherwise static image. Haag and Nagel (1999) look at image edge elements, where the optical flow is more accurate, to guide their model-based tracking. Work by Rosales and Sclaroff (1999) tracked objects in 3D using an EKF, while also using a background subtraction mechanism.

Earliest notable work with applications to obstacle detection from a moving vehicle is that of Krüger, Enkelmann, and Rössle (1995). In this work, optical flow vectors are first clustered as a noise reduction and speed enhancement step. These clustered flow vectors are then used in a probabilistic framework to classify into three succinct object categories: (*i*) ground plane, (*ii*) static obstacle, and (*iii*) dynamic obstacle. Quite similarly, Roberts and Dellaert (2013) proposed obstacle detection as a classification problem where optical flow vectors are used to classify superpixels into a set of optical flow templates.

Roberts and Dellaert (2014) then extended this work to infer these optical flow templates directly using the spatio-temporal gradients, leaving optical flow as a latent variable in the estimation. This Bayesian framework allows them to avoid the expensive computation of optical flow vectors.

McManus et al. (2013) applied optical flow for background detection on an autonomous vehicle, which is the dual of the obstacle detection problem. This system assumes an already known localization within a dense 3D prior map, which they then compare real-time optical flow against. They perform this comparison by predicting optical flow and running the same real-time optical flow algorithm on the synthetically generated views derived from the prior map.

1.4.2 Obstacle Detection using Depth Reconstruction

Methods looking to reconstruct the local depth of the scene are akin to approaches in the structure from motion and monocular SLAM community. Work by Yamaguchi, Kato, and Ninomiya (2006) is able to track sparse features and use consistent features for determining egomotion, yet label inconsistent features as dynamic obstacles. Wedel et al. (2006) derive scene depth by considering the scale factor change of image regions that are tracked between images, noting that this method is more robust to detecting distant obstacles near the focus of expansion. Finally, Becker et al. (2013) demonstrate dense, accurate depth maps obtained in a variational estimation over scene structure and egomotion, which also yields depth map uncertainties.

1.4.3 Obstacle Detection using Segmentation

Segmentation methods to obstacle detection look at single image decomposition into object categories using either extracted features or the full image. Perhaps the most successful general obstacle detection through classification is the work of Felzenszwalb et al. (2010); this work uses HOG-like image features for part-based object detection in a latent SVM. Held, Levinson, and Thrun (2012) extended this approach by specializing the detector to car detection—considering a simplified, structured environment that allows for constraints on context and scale of the detected objects.

Without relying on features for classification, Ulrich and Nourbakhsh (2000) directly classify each pixel as obstacle or not by comparing against a learned appearance model of ground images—adaptively learning this appearance model as the robot traverses the environment. Further, road scene segmentation has been a heavily researched area that typically requires a learned appearance model of various class labels, such as roadways, obstacles, and vegetation (Alvarez et al., 2012; Brostow et al., 2008; Irie and Tomono, 2013; Tighe and Lazebnik, 2010). Many of these works leverage a 2D Markov random field (MRF) over the image plane to robustly fuse different extracted features and cues, which can then be optimized over using graph cut (Boykov, Veksler, and Zabih, 2001)

More recently, the use of 1D MRFs have been proposed in which the variables form a chain across the image, left-to-right, and model a partitioning between the top and bottom halves of an image column. The notion is that obstacles in the image frame are resting on the roadway, thus, our imager observes two disjoint sets per column: the road and obstacles on the road. Viewing the problem in this way enforces a strict regularization on the problem of obstacle detection from cameras. This idea was originally conceived by Badino, Franke, and Pfeiffer (2009) in their "Stixel World" and was used for obstacle detection with stereo cameras. The task was also considered with monocular cameras in which the appearance of the partition is discerned from cues that are hand-tuned (Yao et al., 2015) or learned using a convolutional neural network (CNN) (Levi, Garnett, and Fetaya, 2015).

This approach has also been proposed in a multi-level framework so that obstacles can be split to a higher fidelity than strictly ground versus obstacle, instead looking at many layers of obstacles that can exist in an image column. The original stixel-world was augmented to include this by Pfeiffer and Franke (2011). Similar multi-layer dynamic programming solutions have been considered in generic image segmentation (Felzenszwalb and Veksler, 2010) as well as for multi-class image labeling (Liu et al., 2015).

1.5 Next Generation Vehicle (NGV) Project

Since 2012, the University of Michigan and Ford Motor Company have been developing a fully autonomous vehicle as part of the Next Generation Vehicle (NGV) project; sample platforms developed through this project are shown in Fig. 1.3. The end goal of the project is to develop a system that enables full platform autonomy. Work with this project allows us direct access to platforms and these vehicles are all equipped with the following sensors:

- Four Velodyne HDL-32E 3D LIDAR scanners
- Applanix POS-LV 420 inertial navigation system (INS) with external DMI
- Point Grey monocular camera
- Delphi radar

The arrangement of our four LIDAR scanners is especially unique in that we opted for a distributed network of smaller scanners, as opposed to the larger and more expensive Velodyne HDL-64E. While this requires us to carefully calibrate each sensor with respect to each other—including extrinsic and remittance calibration—it allows for more unique sensing patterns and is a step toward our eventual target of considering a network of even smaller and cheaper 3D LIDAR scanners than the Velodyne HDL-32Es that can simply augment the needs of the vision system. This configuration also allows our platform to be robust to sensor outages; while we cannot operate autonomously without full sensor input, an outage from one of our sensors allows us to more safely execute emergency procedures with the remaining available sensors.

1.6 Thesis Outline

The current state-of-the-art of localization within prior maps for autonomous cars has relied heavily on registration with a 3D LIDAR scanner against a ground reflectivity map. This thesis is interested in solving many issues that can improve on this approach with the goal of greater robustness, toward an always available localization solution. We further have a thrust in limiting the use of intermediate feature layers in our approach, where we have a preference of using data in its rawest form as often as possible for localization, thus reducing the failure points of our system. We address the following core problems:

1. Localization using ground-plane reflectivities alone can lead to issues and localization divergence under poor weather, degraded ground appearance, and in areas that do not provide enough visual variation for accurate localization.

(a) TORC ByWire XGV (b) Ford Fusion Hybrid Autonomous Research Vehicle

Figure 1.3: Two of the several autonomous vehicles available as part of the Next Generation Vehicle project. Our highly configurable TORC ByWire XGV that is outfitted with an adjustable roof rack helpful for prototyping various sensor configurations is shown in (a), while (b) shows one of our Ford Fusion Hybrids. These platforms are equipped with four Velodyne HDL-32E 3D LIDAR scanners, an Applanix POS-LV 420 INS, a monocular camera, and automotive radars.

2. Significant cost of LIDAR sensors limit the ability to add additional sensors as a measure of redundancy—while cameras provide similar data content for localization at a fraction of the cost. Meanwhile, the current approaches to visual localization often consider feature-based methods that rely on hand-tuned features that can be prone to failure.

Toward this objective, we have produced the following contributions[1]:

1. Collected an extensive dataset of over 500 km of road data spanning several months; these datasets traverse construction zones including areas that are fully repaved and a dataset collected under heavy snowfall. Further, we generated ground-truth pose estimates for all trajectories so that we can benchmark the quality of our proposed work.

2. Developed a Gaussian mixture map representation that allows us to condense the full 3D and reflectivity information of the world into a compact, parametric representation. This allows us to jointly reason over structure and appearance of our environment in an efficient multiresolution, branch-and-bound search. Our proposed method is robust to areas where appearance has been altered by heavy snowfall or road construction, and in areas that are visually feature poor.

3. Developed a visual localization framework that allows for localization within LIDAR-derived maps, which allows for accurate, redundant localization using inexpensive sensors.

[1]Portions of this work appear in Wolcott and Eustice (2014, 2015, 2016)

Our approach uses whole-image matching against a projected view of the LIDAR maps, evaluating candidate registrations using normalized mutual information (NMI).

4. Developed a visual obstacle detection pipeline that leverages these detailed prior maps for improved obstacle detection. Further, this understanding of obstacles in the visual field of view allows us to account for them during localization for added robustness.

1.6.1 Document Roadmap

The contributions are detailed in the following chapters:

Chapter 2 We show that the full 3D state of the world can be captured into a compact Gaussian mixture map, which we can register an online point cloud into using an efficient multiresolution, branch-and-bound search. This efficient approach allows registration rates that match the high framerate of our LIDAR sensors. We further show our method jointly reasons over structure and appearance of the environment, which allows our method to be robust to areas that are visually feature poor or have undergone severe degradation relative to the original mapping. Thorough evaluation over more than 500 km of on-road data is performed that culminates in what we believe to be the state-of-the art for robust, LIDAR-based localization.

Chapter 3 We propose a method for visually localizing into prior maps built using a survey vehicle equippped with LIDAR scanners. In this section, we efficiently generate synthetic views of this prior map and use NMI to directly evaluate raw pixel data for registration of our camera's location within this prior map. We demonstrate our method on experimental datasets that show that the use of a camera alone for localization can achieve similar orders of magnitude error rates as its LIDAR counterparts with a sensor that is several orders of magnitude cheaper.

Chapter 4 We introduce a visual obstacle detection framework that allows for improved robustness to visual occlusions in our proposed visual localization method. Further, we demonstrate that our method, which relies on a detailed prior map, can rival state-of-the-art methods for visual obstacle segmentation.

Chapter 5 We conclude by highlighting the key contributions of this thesis and present areas for future work.

Appendix A We detail the odometry model used throughout our thesis on our platforms, including our proposed learned uncertainty model.

Appendix B We summarize our offline SLAM framework that is used to construct maps for localization and provide a mechanism for generating ground-truth for our experimental analysis.

CHAPTER 2

Fast LIDAR Localization using Multiresolution Gaussian Mixture Maps

2.1 Introduction

Over the past several years, fully autonomous, self-driving cars have become feasible with progress in the simultaneous localization and mapping (SLAM) research community and the advent of consumer-grade three-dimensional (3D) light detection and ranging (LIDAR) scanners. Systems such as the Google driverless car use these LIDAR scanners combined with high accuracy GPS/INS systems to enable cars to drive hundreds of thousands of miles without user control (Thrun, 2010). As manufacturers continue to reduce the price and increase the aesthetic appeal of 3D LIDAR scanners, their use on production automated vehicles will become a reality.

In order to navigate autonomously, the prevalent approach to self-driving cars requires precise localization within an *a priori* known map. Rather than using the vehicle's sensors to explicitly extract lane markings, traffic signs, etc., metadata is embedded into a prior map, which reduces the complexity of perception to a localization problem. State-of-the-art methods (Levinson and Thrun, 2010; Levinson, Montemerlo, and Thrun, 2007) use reflectivity measurements from 3D LIDAR scanners to create an orthographic map of ground-plane reflectivities. Online localization is then performed with the current 3D LIDAR reflectivity scans and an inertial measurement unit (IMU).

Reflectivity-based methods alone can fail when the road appearance is degraded over time or occluded by harsh weather. In this work, we seek a fast, optimal scan matcher that allows us to quickly localize a vehicle within a prior map by exploiting the 3D structure of the scene in addition to ground-plane reflectivities.

We propose the use of a pair of Gaussian mixture maps—a two-dimensional (2D) grid structure where each grid cell contains a Gaussian mixture model. One such map characterizes the distribution over z-height (i.e., vertical structure) and another for capturing the

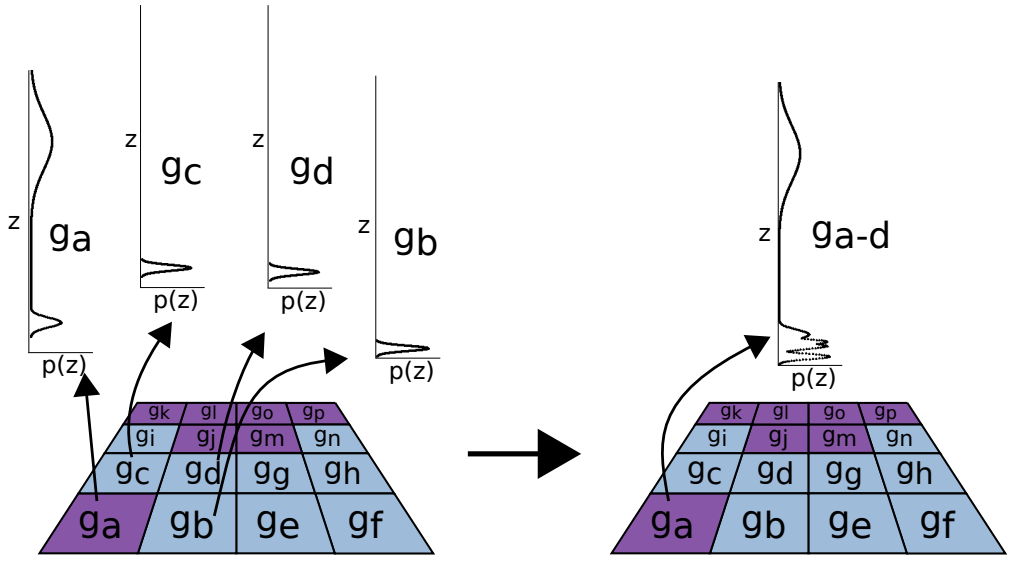


Figure 2.1: Overview of our proposed LIDAR localization scheme. We propose the use of Gaussian mixture maps—a 2D grid over xy where each cell in the grid holds a one-dimensional Gaussian mixture model that accurately models the distribution of points contained in this infinite-height cell. We consider two representations that independently model z-height and reflectivity of points, then perform registration in these maps by formulating a branch-and-bound search over multiresolution, rasterized versions of the Gaussian mixture maps where coarser resolutions provide an upper-bound over the finer resolutions. This methodology finds the guaranteed optimal registration over a user-specified search space. The figure on the left depicts a z-height Gaussian mixture map, where the grid is colored by the difference between the two Gaussian modes in the cell, *blue* indicates 2 overlapping mixture components around the ground-plane and *purple* indicates two distinct modes captured including ground-plane and superstructure; the figure on the right shows the multiresolution look-up tables that our method uses.

distribution over reflectivity (i.e., appearance). Gaussian mixture maps allow us to fully extract all point cloud data while mapping and compress the distributions into a compact, parametric representation.

When used for localization, we can again use all online point cloud data to register against these maps, thus improving robustness of our method by avoiding the need to extract higher level features to perform registration. While this registration may appear expensive, we present a novel upper-bound through rasterizations of the sum of Gaussian mixtures that enables us to formulate the scan matching problem as a branch-and-bound search. See Fig. 2.1 for a sample of these maps.

In this chapter, we present Gaussian mixture map localization as we initially published in (Wolcott and Eustice, 2015) as well as several extensions. This work represents the following contributions:

- Data reduction of large point clouds to a compact mixture of Gaussians, capturing both structure and appearance of the point cloud.
- An online rasterization of these parametric maps that enables a fast branch-and-bound registration formulation for real-time, guaranteed-optimal registration.
- A robust formulation that jointly considers structure and point appearance using a robust cost function for removing outliers.
- Implementation of our algorithms on a graphics processing unit (GPU) that yields 40× speedup over the central processing unit (CPU), which allows us to localize using all point cloud points without spatial downsampling.
- Extensive evaluation over several hundred kilometers of road data, in which we demonstrate successful localization through diverse environments including heavy snowfall, construction, and asphalt repaving—all demonstrating robustness to appearance changes.

2.2 Related Work

Automated vehicles require robust localization algorithms with low error and failure rates. One of the most pervasive strategies relies on observation of ground plane reflectivities, a signal that captures lane markings, pavement variation, tar strips, etc. Levinson, Montemerlo, and Thrun (2007) initially proposed using a 3D LIDAR scanner to observe the ground-plane reflectivities, with which they were able to build orthographic maps of ground reflectivities and perform localization using the current 3D LIDAR scans and an IMU. Baldwin and Newman (2012) employed a similar approach by using a 2D LIDAR scanner to build 3D swathes as the vehicle traversed the environment.

Despite attempts by Levinson and Thrun (2010) to model slight changes in appearance of these ground plane maps by considering the variance of the prior map (in addition to previous methods that only captured the mean), appearance based methods can fail when harsh weather is present in the environment—for example, rain puddles and snowdrifts can build up and occlude the view of the informative ground signal, see Fig. 2.2. Additionally, long two-lane roads with a double lane-marker between them can allow longitudinal uncertainty to grow unbounded due to lack of texture perpendicular to the road. Thus, to increase robustness to these types of scenarios, we are interested in augmenting these appearance methods by exploiting the 3D structure of the scene that is observed with a LIDAR scanner in a fast and efficient manner.

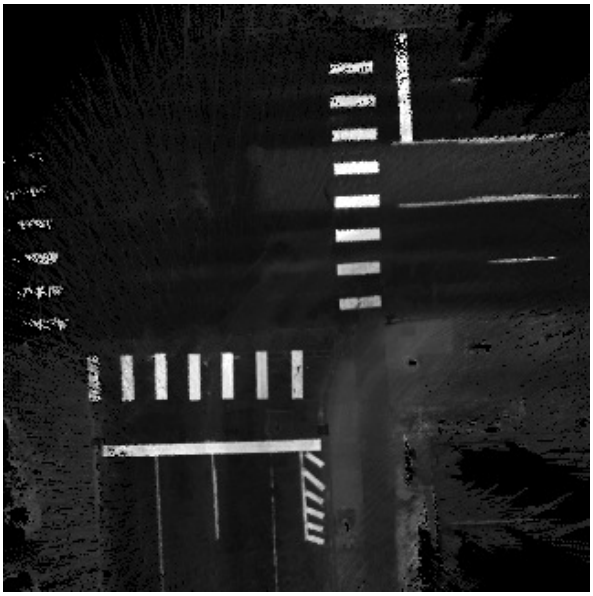
Specifically, we are interested in registering a locally observed point cloud to some prior 3D representation of our environment. Many similar robotic applications use iterative closest point (ICP) (Besl and McKay, 1992), generalized iterative closest point (GICP) (Segal, Haehnel, and Thrun, 2009), normal distributions transform (NDT) (Magnusson, 2009), or other similar variants to register an observed point cloud to another point cloud or distribution. Registration using these methods typically requires defining a cost function between two scans and evaluating gradients (either analytical or numerical) to iteratively minimize the registration cost. Due to the nature of gradient descent, these methods are highly dependent on initial position and are subject to local minimums.

To overcome local minima and initialize searches near the global optimum, several works have been proposed that extract distinctive features and perform an alignment over these first. For example, Rusu (2009) and Aghamohammadi et al. (2007) presented different features that can be extracted and matched from a raw point cloud. Pandey et al. (2011) bootstrap their registration search with visual feature correspondences (e.g., SIFT). However, these feature-based approaches rely on extracting robust features that are persistent from various viewpoints.

As an alternative to searching for a single best registration for each scan, Chong et al. (2013), Kümmerle et al. (2008), and Maier, Hornung, and Bennewitz (2012) all demonstrated localization implementations built upon a Monte Carlo framework. Their approach allows particles to be sampled throughout the environment and evaluated relative to a prior map. This filtering methodology should be more robust to local minima because the particles should ideally come to a consensus through additional measurements—though this is dependent on random sampling and can make no time-based optimality guarantees.

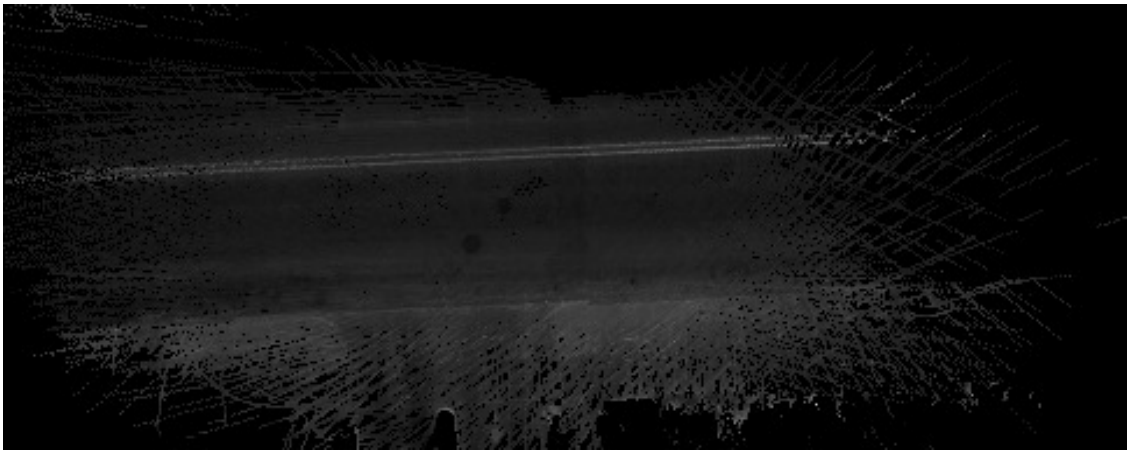
Finally, multiresolution variations on the above algorithms have been proposed that allow expanded search spaces to be explored in a coarse-to-fine manner in hopes of avoiding local minima. This has been applied to ICP (Granger and Pennec, 2002), NDT (Magnusson, 2009; Ripperda and Brenner, 2005; Ulaş and Temelta, 2013), and occupied voxel lists (Ryde and Hu, 2010). These searches use heuristics to greedily guide the coarse-to-fine steps that yield good results in practice, but still cannot guarantee global optimality.

We employ techniques presented by Olson (2009, 2015) to formulate the multiresolution search as a branch-and-bound problem that can guarantee *global* optimality over our search space. In this work, we extend Olson (2009) to handle full-3D point clouds by creating efficient Gaussian mixture maps for fast and accurate inference. Similar to Maddern, Pascoe, and Newman (2015), we formulate a joint cost function that allows LIDAR localization using z-height and reflectivity maps, though our approach captures the full distribution rather than just a mean and a fixed variance.



(a) Good Weather

(b) Light Snow on Roads



(c) Poor Texture in Road

Figure 2.2: Common snapshots of orthographic LIDAR reflectivity maps. Notice the severe degradation of quality in the snow covered roads and the hallucination of lane markings caused by tire tracks through snow. Also, poor texture is a common occurrence on two-lane roads, which often result in laterally constrained cost functions.

2.3 Gaussian Mixture Maps

The key challenge to fast localization is a prior representation of the world that facilitates efficient inference. We propose using Gaussian mixture maps that discretize the world into a 2D grid over the xy plane, where each cell in the grid contains a Gaussian mixture that characterizes the 3D points contained within this infinite column. To capture both structure and appearance, we construct a pair of independent Gaussian mixture maps that capture the z-height and reflectivity distribution of each cell, respectively.

The Gaussian mixture map over z-height offers a compact representation that is quite similar to a 2.5D map, with the flexibility of being able to simultaneously and automatically capture the multiple modes prevalent in the world—including tight distributions around the ground-plane and wide distributions over superstructure. This representation is quite similar to NDT maps—Gaussian mixture maps exist in the space between the 2D-NDT (Biber, 2003) and the 3D-NDT (Magnusson, 2009). Like these approaches, Gaussian mixture maps can be viewed as a Gaussian mixture over the environment, though our maps are a collection of discontinuous one-dimensional Gaussians rather than a continuous multivariate Gaussian. This means that, when registering a point, likelihood evaluation is a function of z conditioned on the corresponding xy cell the point falls in.

Moreover, the z-height Gaussian mixture map is also similar to multi-level surface (MLS) maps from Triebel, Pfaff, and Burgard (2006), which cluster the point cloud into *horizontal* and *vertical* structure components using distance-based heuristics. Rather than reducing our point cloud into similar discrete intervals to characterize the z-height distribution, we instead run expectation-maximization (EM) to fit a Gaussian mixture model for each grid cell to capture the true probabilistic distribution of our observed point cloud.

The reflectivity Gaussian mixture map is a generalized version of the probabilistic reflectivity maps presented by Levinson and Thrun (2010). Their approach fits a single Gaussian per cell, while Gaussian mixture maps can fit more than one mode to capture above-ground appearance features (e.g., signs, building facades, foliage) as well as accurately capture the true distributions at the edge of lane markers.

The reminder of this section details construction of these maps and how they are used in a joint framework for robustly registering a vehicle equipped with LIDAR sensors.

2.3.1 Map Construction

The first portion of our localization framework is the offline mapping stage, which generates the map to be used for online localization. Our goal here is to generate a map that is metrically accurate to the environment. To do this, we use the state-of-the-art in nonlinear

least-squares, pose-graph SLAM and measurements from a 3D LIDAR scanner to map the 3D structure in a global frame.

As detailed in Appendix B, our offline SLAM pipeline provides a ground-truth set of poses, $X = \{\mathbf{x}_i\}_{i=0}^{M}$ optimized into a locally consistent frame. Each ground-truth pose has a corresponding point cloud, $\mathcal{P}_i = \{\mathbf{p}_j\}_{j=1}^{n}$, where $\mathbf{p}_j = [x_j, y_j, z_j, r_j]^\top$ is the metric position of a point in space and its corresponding reflectivity as measured by our laser scanner. Each point is motion compensated according to our odometry to account for motion during point cloud acquisition.

We then transform each of these point clouds into the SLAM optimized frame, accumulating into a single, global point cloud, $\mathcal{P} = \{\mathbf{x}_i \oplus \mathcal{P}_i\}_{i=0}^{M}$, where $\oplus$ denotes the head-to-tail composition operation (Smith, Self, and Cheeseman, 1990) transforming each body-frame point cloud into the SLAM frame. Accumulating every point directly would be inefficient in memory and computation, so we instead use a sparse histogram implemented with a hash table. Thus, we incrementally build two separate sparse histograms $H_z(x, y, z)$ and $H_r(x, y, r)$ whose hash key is a bitwise concatenation of cell locations,

$$\texttt{key}_m(x, y, m) = \lfloor x/q_{xy} \rfloor \lfloor y/q_{xy} \rfloor \lfloor m/q_m \rfloor \tag{2.1}$$

where q_{xy} is the corresponding Gaussian mixture map grid resolution that will be analyzed in Section 2.6.2, $m \in \{z, r\}$, and q_m is the resolution set to the desired fidelity of z and r. The corresponding hash value is a histogram count that is gradually incremented as points are added to the sparse histogram. In order to capture the variance of our LIDAR scanner and reduce discretization errors, we blur each point by incrementing neighboring histogram cells according to a Gaussian kernel with standard deviation of 5 cm. This helps us later account for measurement uncertainty in our likelihood evaluation.

Next, we perform weighted EM for each "column" in H_z and H_r to construct Gaussian mixture maps for z-height, $\mathcal{G}_z \leftarrow H_z$, and reflectivity, $\mathcal{G}_r \leftarrow H_r$. These two reductions are independent and can again be generalized as $\mathcal{G}_m \leftarrow H_m$ for clarity.

For a specific cell $(\hat{x}, \hat{y})$, we extract the corresponding histogram "column" of data,

$$E_m(\hat{x}, \hat{y}) = \left\{ \mathbf{e}_i^{(\hat{x},\hat{y})} \right\}_{i=1}^{f} = H_m(x = \hat{x}, y = \hat{y}, :), \tag{2.2}$$

where $\mathbf{e}_i^{(\hat{x},\hat{y})} = [c_i, m_i]^\top$ is the i^{th} histogram entry with count c_i for the cell centered at $(\hat{x}, \hat{y}, m_i)$; f is the number of cells observed in this histogram column.

We are then interested in condensing this column of data into a Gaussian mixture, $\mathcal{G}_m(x = \hat{x}, y = \hat{y}) = \left\{ \mathbf{g}_j^{(\hat{x},\hat{y})} \right\}_{j=1}^{g}$, where $\mathbf{g}_j^{(\hat{x},\hat{y})} = [w_j, \mu_j, \sigma_j]^\top$ is the j^{th} component of g

Gaussians, parameterized by weight, mean, and standard deviation, respectively. This is achieved using the EM algorithm to iteratively estimate likelihood of Gaussian components given the data (**E**xpectation) and re-estimate new components that maximize this expected likelihood (**M**aximization). Histogram counts, c_i, are used to weight the expected likelihood derived in the expectation step.

Contrary to our previous work that iterated through multiple numbers of Gaussians, g, choosing the number of parameters that best fit the data while penalizing proportional to number of mixture components to avoid overfitting (Wolcott and Eustice, 2015), we found that this approach does not scale well with creating large maps and overcomplicates the online usage of the maps. We instead suggest a fixed number of Gaussians so that EM only needs to be run once per map cell. In the worst case, this results in redundant Gaussians summing to the same resulting likelihood distribution. We provide discussion and recommendations in Section 2.6.2.1 for how many Gaussians to choose and a visual depiction of what each map is capturing can be seen in Fig. 2.12, Fig. 2.13, and Fig. 2.14.

In this work, we present two methods for deriving the Gaussian mixture map over reflectivity. First, we consider the reflectivity of the entire point cloud resulting in $\mathcal{G}_r \leftarrow H_r$. Alternatively, we can use the 3D position of each point in space to extract the ground-plane only using a region growing method emanating from the known ground height around the vehicle; this results in a Gaussian mixture over ground surface reflectivities $\mathcal{G}_{r,\texttt{grd}} \leftarrow H_{r,\texttt{grd}}$. This is a general representation that is identical to the probabilistic maps presented by Levinson and Thrun (2010) when the number of Gaussian components is $g = 1$.

2.3.2 Registration Formulation

Given a point cloud, $\mathcal{P}$, we seek to find the optimal transformation that maximizes the likelihood of being drawn from the underlying Gaussian mixture maps, $\mathcal{G} = \{\mathcal{G}_z, \mathcal{G}_r\}$. This is directly formulated as the maximum likelihood estimate (MLE) to find the optimal alignment T^*,

$$T^* = \underset{T}{\operatorname{argmax}}\, \mathcal{L}(\mathcal{P}|T, \mathcal{G}), \tag{2.3}$$

where $T = [x, y, z, r, p, h]^\top$ is a 6-DOF transformation that transforms points of $\mathcal{P}$ into $\mathcal{G}$.

The point cloud is made up of a set of n points, $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^n$, where $\mathbf{p}_i = [x_i, y_i, z_i, r_i]^\top$ is the metric position and reflectivity of each point. We assume independence between points

to arrive at

$$T^* = \underset{T}{\operatorname{argmax}} \sum_i \mathcal{L}(\mathbf{p}_i | T, \mathcal{G}). \tag{2.4}$$

We further assume independence between z-height of a point and its reflectivity, and use the chain rule over x_i and y_i,

$$\mathcal{L}(\mathbf{p}_i | T, \mathcal{G}) = \mathcal{L}(z_i | T, \mathcal{G}_z, x_i, y_i) \mathcal{L}(r_i | T, \mathcal{G}_r, x_i, y_i) \mathcal{L}(x_i, y_i). \tag{2.5}$$

We further marginalize out x_i and y_i, realizing that this distribution is fully captured by the blurring in our maps—thus, the corresponding distribution $\mathcal{L}(x_i, y_i)$ is already accounted for in the likelihood measure. This results in the joint likelihood over structure and appearance as

$$\mathcal{L}(\mathbf{p}_i | T, \mathcal{G}) = \mathcal{L}(z_i | T, \mathcal{G}_z, x_i, y_i) \mathcal{L}(r_i | T, \mathcal{G}_r, x_i, y_i). \tag{2.6}$$

These likelihoods are computed by first transforming the point $[x_i, y_i, z_i]^\top$ by T, resulting in $[x_i', y_i', z_i']^\top = T \oplus [x_i, y_i, z_i]^\top$. This allows us to compute each likelihood by indexing into the corresponding Gaussian mixture maps and summing over the mixture components,

$$\begin{aligned} \mathcal{L}(z_i | T, \mathcal{G}_z, x_i, y_i) &= \mathcal{L}(z_i' | \mathcal{G}_z(x_i', y_i')) \\ &= \sum_j \frac{w_{ij}}{\sqrt{2\pi {\sigma_{ij}}^2}} \exp\left(-\frac{(z_i' - \mu_{ij})^2}{2{\sigma_{ij}}^2} \right), \end{aligned} \tag{2.7}$$

and

$$\begin{aligned} \mathcal{L}(r_i | T, \mathcal{G}_r, x_i, y_i) &= \mathcal{L}(r_i | \mathcal{G}_r(x_i', y_i')) \\ &= \sum_j \frac{w_{ij}}{\sqrt{2\pi {\sigma_{ij}}^2}} \exp\left(-\frac{(r_i - \mu_{ij})^2}{2{\sigma_{ij}}^2} \right), \end{aligned} \tag{2.8}$$

where w_{ij}, μ_{ij}, and σ_{ij} are the weight, mean, and standard deviation, respectively, of the j^{th} component of $\mathcal{G}_z(x_i', y_i')$ and $\mathcal{G}_r(x_i', y_i')$.

However, we notice that these points may be drawn from the underlying Gaussian mixture maps *or* are obstacles drawn from a separate distribution. To limit the effect of outliers in our registration formulation, we modify (2.7) and (2.8) by mixing them with a uniform distribution. Specifically, the robust likelihoods take the form

$$\mathcal{L}'(z_i | T, \mathcal{G}_z, x_i, y_i) = \alpha \mathcal{L}(z_i | T, \mathcal{G}_z, x_i, y_i) + (1 - \alpha) \mathcal{U}(z_i), \tag{2.9}$$

and

$$\mathcal{L}'(r_i|T, \mathcal{G}_r, x_i, y_i) = \beta\mathcal{L}(r_i|T, \mathcal{G}_r, x_i, y_i) + (1 - \beta)\mathcal{U}(r_i), \tag{2.10}$$

where α and β are mixing parameters that control the region of influence of the underlying Gaussian mixture—having the effect of truncating the Gaussian distribution outside this region of influence. Further, the range of the uniform distributions are inconsequential, though are set to the range of the data. The resulting maximization including robust cost functions then looks like

$$T^* = \underset{T}{\operatorname{argmax}} \sum_i \mathcal{L}'(z_i|T, \mathcal{G}_z, x_i, y_i)\mathcal{L}'(r_i|T, \mathcal{G}_r, x_i, y_i). \tag{2.11}$$

The addition of the robust cost function is quite important for points in the roadway where Gaussian mixtures of the ground plane z-height typically have an extremely small variance. Without the robust formulation, these points would dominate the cost function and force the registration to overfit to outliers (e.g., obstacles). Moreover, we compute the log-likelihood for numerical stability, which allows us to compute a running sum independently as a parallel reduction.

Further, if the reflectivity Gaussian mixture map is modeled using the ground plane only ($\mathcal{G}_{r,\texttt{grd}}$ is used), then we only evaluate the reflectivity likelihood using ground points from $\mathcal{P}$. Online we use the same region growing method to extract the local ground plane; points not belonging to the ground plane will have a fixed likelihood that has no impact on the cost function. Note, however, that the z likelihood is still computed for all points in $\mathcal{P}$.

Considering now the optimization to find T^*, we make the observation that a typical wheeled-robotic platform is well constrained in *roll*, *pitch*, and *height* because (i) most IMUs constrain *roll* and *pitch* to within a few degrees due to observation of the gravitational force (note that wheeled platforms only traverse minor roll/pitch) and (ii) any wheeled vehicle must be resting on the ground surface, which constrains *height* with a prior map. Thus, (2.3) can be maximized by exhaustively searching over a range of x, y, and *heading* transformations. As in Olson (2009), we can efficiently compute these by applying the *heading* rotation to all points *first*, then evaluate at xy translations.

With our solution within the vicinity of the optimum, we then perform a simple, constrained 6-DOF hill-climbing to lock into the global optimum over our search space, T^*. This allows for the small, but necessary refinements of *height*, *roll*, and *pitch*. Because our registration problem is parameterized by the search boundaries, we are able to use pose priors to improve run-time performance. A detailed overview of registration into our Gaussian mixture map can be found in Algorithm 1 and Algorithm 2.

Algorithm 1 Full Registration

Input: GMM $\mathcal{G} = \{\mathcal{G}_z, \mathcal{G}_r\}$, Point Cloud $\mathcal{P}$, guess $T_0 = (x_0, y_0, z_0, r_0, p_0, h_0)$, search space X, Y, H
Output: Optimal registration, $T^* = (x^*, y^*, z^*, r^*, p^*, h^*)$

$(\hat{x}, \hat{y}, z_0, r_0, p_0, \hat{h}) =$ `SEARCH`$(x_0, y_0, z_0, r_0, p_0, h_0)$
$(x^*, y^*, z^*, r^*, p^*, h^*) =$ `HILL-CLIMB` $(\hat{x}, \hat{y}, z_0, r_0, p_0, \hat{h})$

Algorithm 2 Exhaustive Search

Input: GMM $\mathcal{G} = \{\mathcal{G}_z, \mathcal{G}_r\}$, Point Cloud $\mathcal{P}$, guess $T_0 = (x_0, y_0, z_0, r_0, p_0, h_0)$, search space X, Y, H
Output: Best 2D registration $= (\hat{x}, \hat{y}, \hat{h})$

$best = -\infty$
for h_i in $h_0 + H$ **do**
    apply rotation h_i to $\mathcal{P}$
    for x_i, y_i in $\{x_0, y_0\} + XY$ **do**
        $likelihood = \mathcal{L}(\mathcal{P}|x_i, y_i, \mathcal{G})$ ▷ (2.11)
        if $likelihood > best$ **then**
            $best = likelihood$
            $(\hat{x}, \hat{y}, \hat{h}) = (x_i, y_i, h_i)$
        end if
    end for
end for

2.4 Multiresolution Branch-and-Bound

Typically, exhaustively searching for the maximum likelihood is not a realistic, tractable solution. In this section, we replace the expensive, exhaustive search with an efficient multiresolution branch-and-bound search.

2.4.1 Multiresolution Formulation

The idea behind our multiresolution search is to use a bounding function that can provide an upper-bound over a collection of cells in our reference map. This means that a majority of the search can be executed at a coarser resolution that upper-bounds the likelihood at finer scales. Using tight bounds can transform the exhaustive search presented in the previous section into a tractable search that makes *no* greedy assumptions. The branch-and-bound strategy achieves exactly the same result as the exhaustive search, only arrives at it in a more efficient manner.

For evaluating a single transformation (i.e., (T_x, T_y)), one must evaluate the log-likelihood of each point in a point cloud, then sum all of these for a total log-likelihood. Therefore in the exhaustive case, each point is evaluated against a single Gaussian mixture. In order to search a range of transformations, such as (T_x, T_y) to $(T_x + Nq_{xy}, T_y + Nq_{xy})$, each point is evaluated against a total of $(N+1)^2$ Gaussian mixtures. However, each cell in our map is

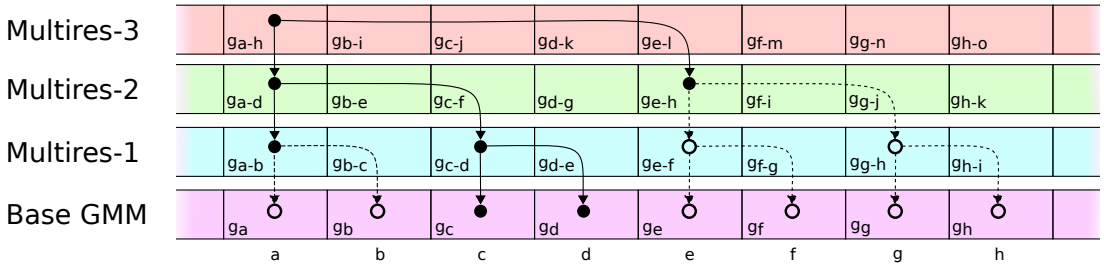


Figure 2.3: A one-dimensional example of our multiresolution search formulation, where we demonstrate how a *single* point cloud point would traverse through the multiresolution tree. Given some knowledge that the best transformation aligns the point somewhere within a-h, we begin the search at the coarsest resolution in cell a. Using branch-and-bound and computing upper-bounds over the Base GMM distribution in the multiresolution layers, we can efficiently search large spaces by avoiding low likelihood registrations (as depicted by dashed lines and open circles). In this figure, the notation $\mathrm{g_{a\text{-}h}}$ refers to the fact that inference in that cell is an upper-bound over the distributions $\mathrm{g_a} - \mathrm{g_h}$, where g_x is the Gaussian mixture in cell x of the Base GMM. Note that contrary to several other multiresolution approaches, coarser resolutions in our framework *do not* imply a coarser resolution map. We maintain uniform resolution by using many overlapping coarse blocks—a technique that facilitates tighter upper-bounds.

quite spatially similar, meaning that inference into (T_x, T_y) yields a similar log-likelihood as $(T_x + q_{xy}, T_y)$, so the exhaustive search can often spend unnecessary time in low-likelihood regions that can ideally be ruled out quicker.

We formulate a branch-and-bound search that exhaustively searches over the coarsest resolution providing upper-bounds over a range of transformations. These coarse search results are then added to a priority queue, ranked by upper-bound likelihoods. We then iterate through this priority queue, branch to evaluate the next finer resolution, and add back to the priority queue. The search is then complete once the finest resolution is returned from the priority queue.

We propose a slightly different multiresolution map structure than is traditionally considered. In many domains, multiresolution searches imply building coarser versions of your target data and making evaluations on that (e.g., the image pyramid). However, our approach creates many overlapping coarse blocks (as depicted in Fig. 2.3) to better compute tight upper-bounds. This optimization makes the trade off for better bounds as opposed to a smaller memory footprint.

Because our maps are the same resolution throughout each multiresolution layer, this results in us taking larger *strides* through the coarser resolutions, where $stride = 2^{layer} \cdot q_{xy}$. Branching factor and number of multiresolution maps is completely user-defined. In our experiments, we opted for a branching factor of 2; that is, (T_x, T_y) branches into (T_x, T_y),

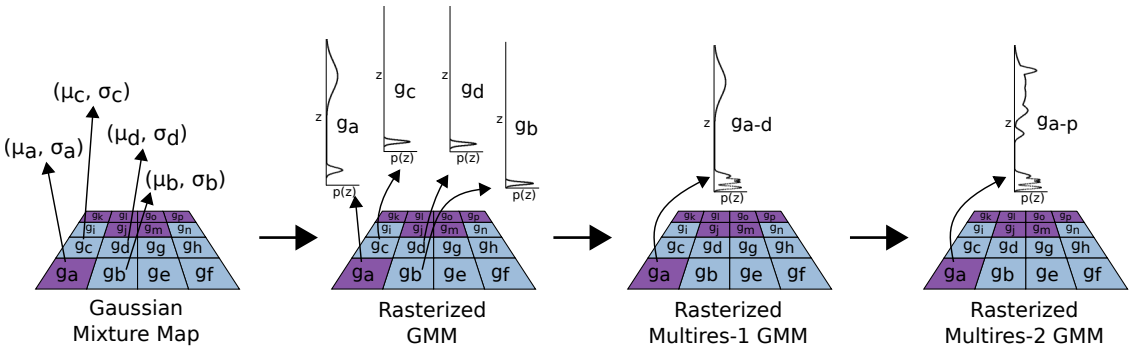


Figure 2.4: Demonstration of the rasterization performed on the original Gaussian mixture map to facilitate exact upper-bounds. We begin with a parametric 2D map that encodes a Gaussian mixture in each cell, where the grid is colored by the difference between the two Gaussian modes in the cell, *blue* indicates 2 modes around the ground-plane and *purple* indicates multiple modes captured including ground-plane and superstructure. We then rasterize each cell (note we display the likelihood, not log-likelihood for clarity); these rasterized representations can then be used to create rasterized upper-bounds for multiresolution search. The first step of this evaluates the upper-bound at each discretization by taking the `max` of the underlying cell rasterizations. Note that as you continue to move to coarser resolutions the distribution generalizes quite well—data for this figure was generated from looking at the edge of a tree, where the multiresolution map can capture the two common modes of tree limbs and ground-plane. In this figure, the notation $g_{a\text{-}d}$ means the rasterization is an upper-bound over the $g_a - g_d$ rasterizations.

$(T_x + stride, T_y)$, $(T_x, T_y + stride)$, and $(T_x + stride, T_y + stride)$ of the finer resolution map. Refer to Algorithm 3 and Fig. 2.3 for a more detailed overview.

2.4.2 Rasterized Gaussian Mixture Maps

Finding tight, *parametric* bounds for a collection of Gaussians is a rather difficult task, so we instead opt for a non-parametric solution in the form of rasterized lookup tables. We take our parametric Gaussian mixture map and compute a rasterized version by evaluating the log-likelihood at a fixed discretization, generating a rasterization for each grid cell. Upper bounds can then be *exactly* computed between neighboring grid cells by taking the `max` across each discretization in the rasterized lookup table. While localizing, likelihoods at the finest resolution are computed using the original Gaussian mixture maps and the rasterized maps only facilitate fast traversal through the search tree. See Fig. 2.4 for a visual representation of these maps.

For a pure localization task such as ours, lookup tables can be pre-computed offline. However, we decided to store *only* the parametrized Gaussian mixture maps on disk to avoid storing extremely large maps. We are then able to efficiently compute rasterized

Algorithm 3 Multiresolution Search

Input: Base and Multires GMM $\mathcal{G} = \{\mathcal{G}_z, \mathcal{G}_r\}$, Point Cloud $\mathcal{P}$, guess $(x_0, y_0, z_0, r_0, p_0, h_0)$, search space X, Y, H

Output: Best registration $= (\hat{x}, \hat{y}, \hat{h})$

// init. priority queue with search over coarse resolution
Initialize $PriorityQueue$ ▷ priority = log-likelihood
$coarsest = N$
$\mathcal{P}_{\texttt{rot}} = empty$ ▷ rotated point clouds
for h_i in $h_0 + H$ **do**
    // store rotated clouds — do transformations once
    $T = f(0, 0, z_0, r_0, p_0, h_i)$ ▷ $[x, y]$ applied later
    $\mathcal{P}_{\texttt{rot}}[h_i] = T \oplus \mathcal{P}$
    for x_i in $x_0 + X/2^{coarsest}$ **do**
        for y_i in $y_0 + Y/2^{coarsest}$ **do**
            $cur.layer = coarsest$
            $cur.[x, y, h] = [x_i, y_i, h_i]$
            $cur.\mathcal{L} = \mathcal{L}(\mathcal{P}_{\texttt{rot}}[h_i] | x_i, y_i, \mathcal{G}[coarsest])$ ▷ (2.11)
            $PriorityQueue.add(cur)$
        end for
    end for
end for
// iterate priority queue, branching into finer resolutions
while $prev = PriorityQueue.pop()$ **do**
    if $prev.layer == 0$ **then**
        // at finest resolution, can't explore anymore
        // this is the global optimum
        $(\hat{x}, \hat{y}, \hat{h}) = prev.[x_i, y_i, h_i]$
        $return(\hat{x}, \hat{y}, \hat{h})$
    end if
    // branch into next finer resolution
    for x_i in $\left[prev.x, prev.x + 2^{prev.layer-1}\right]$ **do**
        for y_i in $\left[prev.y, prev.y + 2^{prev.layer-1}\right]$ **do**
            $cur.layer = prev.layer - 1$
            $cur.[x, y, h] = [x_i, y_i, prev.h]$
            $cur.\mathcal{L} = \mathcal{L}(\mathcal{P}_{\texttt{rot}}[prev.h] | x_i, y_i, \mathcal{G}[cur.layer])$ ▷ (2.11)
            $PriorityQueue.add(cur)$
        end for
    end for
end while

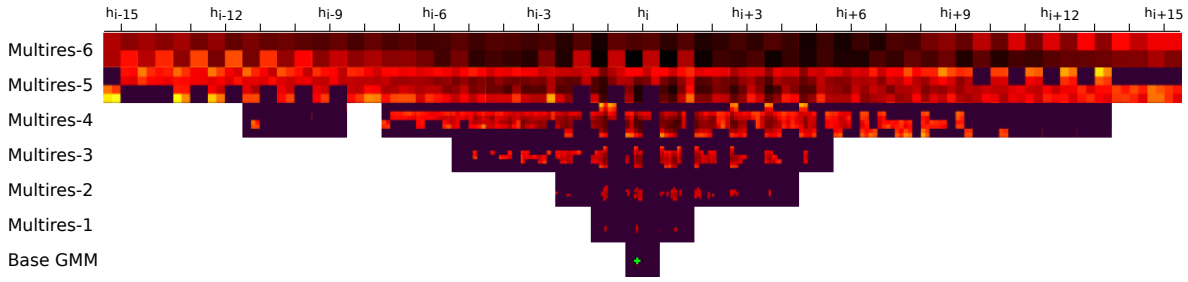


Figure 2.5: Sample multiresolution search space traversal. Top-bottom represents coarse-to-fine searching, left-right represents different slices through our *heading* search, and each pixel depicts an xy translation searched. Log-likehoods are colored increasingly yellow-black, purple and non-existent cells are areas not needed to be explored by the multiresolution search, and the optimal is indicated in green. We exhaustively search the coarsest resolution, then use branch-and-bound to direct our traversal through the tree. For typical scan alignments, we only have to search approximately 1% of the transformations in the finer resolutions, doing a majority of the work in the coarser resolutions.

multiresolution maps online from our parameterized Gaussian mixture map as a background job. This is done incrementally using each successive multiresolution layer to build the next.

Note that our rasterized multiresolution maps are a generic representation that can also be used with other map types including standard NDT maps, MLS maps, occupancy voxels, etc. After converting one of these maps to a rasterized multiresolution map, the remainder of our proposed pipeline can be used for fast registration of a point cloud.

A sample search through our multiresolution search space can be seen in Fig. 2.5. The shown example explores a 25 m $\times$ 25 m area at 16 cm resolution in approximately 2 seconds, while only needing to evaluate 1% of the transformations necessary in the exhaustive search.

2.5 Localization Filter

Our localization task is framed as an estimation problem over the full 6-degree of freedom (DOF) dynamics of our vehicle, where our state vector is $\boldsymbol{\mu}_k = [x_k, y_k, z_k, \phi_k, \theta_k, \psi_k]^\top$. We propose to use an extended Kalman filter (EKF) to estimate our vehicle state from various measurement sources. An EKF provides a simple way to fuse information from multiple sensing modalities; though in this section we only consider integrating our vehicle odometry and multiresolution registrations into our EKF estimation.

We define a discrete time process model and incorporate our registration corrections into

our state filter:

$$\begin{aligned}
\text{Predict} \quad & \bar{\boldsymbol{\mu}}_k = f(\boldsymbol{\mu}_{k-1}, \mathbf{u}_k) \\
& \bar{\Sigma}_k = \mathrm{F}_k \Sigma_{k-1} \mathrm{F}_k^\top + \mathrm{Q}_k \\
\\
\text{Update} \quad & \mathrm{K}_k = \bar{\Sigma}_k \mathrm{H}_k^\top (\mathrm{H}_k \bar{\Sigma}_k \mathrm{H}_k^\top + \mathrm{R}_k)^{-1} \\
& \boldsymbol{\mu}_k = \bar{\boldsymbol{\mu}}_k + \mathrm{K}_k \big(\mathbf{z}_k - \mathbf{h}_k(\bar{\boldsymbol{\mu}}_k) \big) \\
& \Sigma_k = (\mathrm{I} - \mathrm{K}_k \mathrm{H}_k) \bar{\Sigma}_k (\mathrm{I} - \mathrm{K}_k \mathrm{H}_k)^\top + \mathrm{K}_k \mathrm{R}_k \mathrm{K}_k^\top
\end{aligned}$$

Here, $f(\cdot)$ is our nonlinear plant model that integrates odometry measurements from an Applanix IMU, $\mathbf{u}_k$, with uncertainty Q_k and linearized Jacobian F_k; refer to Appendix A for further discussion on this proposed odometry model. H_k is a linear observation model (identity matrix) and K_k is the corrective Kalman gain induced by our registration measurement with uncertainty R_k. The measurement $\mathbf{z}_k$ is exactly the output of our multiresolution registration detailed in Section 2.4, $\mathbf{z}_k = T^*$. We use fixed measurement uncertainties, R_k, that were empirically determined (discussed in Section 2.6.4); however, one could fit a conservative covariance using the explored search space as shown by Olson (2009).

One issue with this formulation is that when measurements arrive there is a non-zero latency associated with *registering* the measurements. This presents a problem when we attempt to add the measurement to our EKF because the measurement would be applied to the future state of the robot. Thus, as soon as the measurement is received (near zero latency), we augment our EKF with a delayed-state (Leonard and Rikoski, 2000). In our discrete model, this leads to an expanded state belief as:

$$\mathbf{x}_k = \left[\boldsymbol{\mu}_k^\top, \boldsymbol{\mu}_{k-1}^\top \right]^\top .$$

Thus, we can continue to apply odometry prediction directly to the most recent state, but apply the registration correction on the associated delayed-state. The correlation inherent between temporal poses will allow the effect of this measurement to then propagate to the current state belief. We then marginalize this delayed-state as it is no longer necessary to maintain in our state vector. This formulation also facilitates the integration of more measurements from various sources to increase robustness, despite the fact that these sources can have varying latencies associated with them.

Our filter is initialized in a global frame from a single dual-antenna global positioning system (GPS) measurement with high uncertainty, which provides a rough initial guess of global pose with orientation. We adaptively update our multiresolution search bounds to ensure that we explore a 4σ window around our posterior distribution. This dynamic approach

(a) TORC ByWire XGV

(b) Ford Fusion Hybrid Autonomous Research Vehicle

Figure 2.6: Test platforms used for evaluation of multiresolution Gaussian mixture map localization: a TORC ByWire XGV and a Ford Fusion Hybrid Autonomous Research Vehicle. Both platforms are equipped with 4 Velodyne HDL-32E LIDAR scanners and an Applanix POS-LV 420 INS.

allows us to improve throughput as our posterior confidence increases, while leaving room to statistically eliminate outlier measurements by evaluating the corresponding measurement normalized innovation squared (NIS). Note that aside from using GPS for initializing the filter, our proposed localization method *only* uses input from inertial sensors, a wheel encoder, and 3D LIDAR scanners.

2.6 Evaluation

In this section, we present a thorough evaluation of our proposed theory covering a diverse set of real-world experiments. All algorithms were implemented in C/C++ using CUDA and, unless otherwise specified, experiments were run on a workstation computer equipped with an Intel Xeon E5-2670 CPU and an NVIDIA GeForce GTX TITAN X GPU. For parallelization in CUDA, we parallelized the inner-loop calculation of registration likelihood (implemented as a parallel sum reduction).

2.6.1 Platforms and Datasets

We evaluate our proposed methods using data collected on our autonomous platforms, a TORC ByWire XGV (Fig. 2.6(a)) and a Ford Fusion Hybrid Autonomous Research Vehicle (Fig. 2.6(b)). These automated vehicles are equipped with four Velodyne HDL-32E 3D LIDAR scanners and an Applanix POS-LV 420 IMU. Given the use of four independent LIDAR scanners, it is crucial to perform extrinsic calibration between these to establish a

rigid body transformation between the IMU and each sensor. This is achieved by formulating a pose-graph as in Appendix B and treating the calibration parameters as unknowns in the optimization.

Moreover, reflectivity measurements need to be calibrated against a known reference map such that measurements are consistent: (*i*) within each LIDAR scanner, (*ii*) between LIDAR scanners on the same platform, and (*iii*) between platforms, as mapping data must be consistent with data observed online. To achieve this, we use a method similar to that proposed by Levinson and Thrun (2014), which derives a map from observed reflectivity to true reflectivity. In their work, each beam rotates about an axis perpendicular to the ground-plane, which allows for learning a mapping $observed\ reflectivity \rightarrow real\ reflectivity$ to implicitly account for angle of incidence. However, our rotation axes are *not* perpendicular to the ground-plane, so we must alter their approach by learning the mapping $\{observed\ reflectivity, rotation\ angle\} \rightarrow real\ reflectivity$.

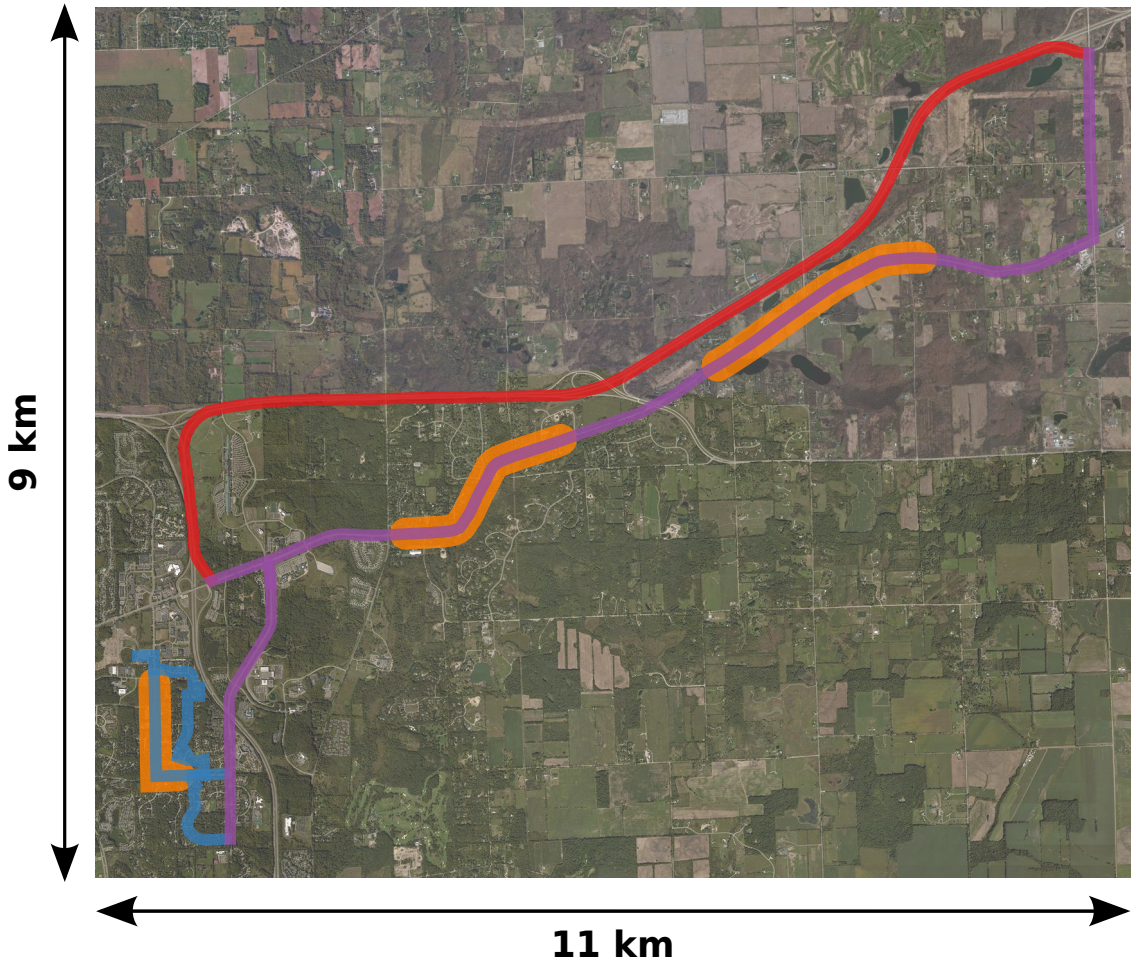
Experiments are presented on two primary datasets collected with our platforms:

- *PG14 Dataset*: Set of 14 logs collected over 3 months with the TORC and Fusion platforms, each log approximately 38 km in length covering a loop near Ann Arbor, Michigan, over **P**lymouth Road, **G**otfredson Road, and M-**14** highway. This dataset totals 525.72 km in length, covering common use cases including highway, rural, and residential areas at various times of day, including rush hour. Furthermore, there were 3 construction zones that evolved over the data collection; 2 of which resulted in full repavings of more than 0.5 km each. See Fig. 2.7 for a visual depiction of the route and an overview of each log.

- *Downtown Dataset*: Set of 5 logs collected with the TORC platform, each spanning a 3 km loop through downtown Ann Arbor, Michigan. This dataset totals 14.92 km of urban roadways and one of these logs was collected on a snowy day with snow actively falling and covering the ground, as depicted in Fig. 2.17(a). See Fig. 2.8 for a visual depiction of the route and an overview of each log.

Collectively, these datasets cover samplings through various environments including highway, urban, rural, and residential roadways during various conditions including heavy traffic, heavy snowfall, and construction zones.

In each of these datasets, the first 2 logs are used for map construction to remove stationary dynamic obstacles (e.g., parked cars). These two mapping logs are stitched together into a single pose-graph, then all data for map construction is compiled at the sparse histogram level, as detailed in Section 2.3.1. Each subsequent log was merged into this pose-graph to provide experimental ground-truth for each—more details can be seen in Appendix B.1. We assume



Ann Arbor—PG14							
ID	Date	Platform	Length	ID	Date	Platform	Length
P-M1	May 1, 2015	TORC	38.0 km	P-M2	May 1, 2015	Fusion	37.9 km
P-1	May 8, 2015	TORC	37.9 km	P-2	May 8, 2015	Fusion	33.4 km
P-3	May 15, 2015	TORC	37.8 km	P-4	May 15, 2015	Fusion	37.9 km
P-5	May 21, 2015	TORC	37.8 km	P-6	May 21, 2015	Fusion	37.9 km
P-7	June 5, 2015	TORC	37.9 km	P-8	June 5, 2015	Fusion	37.9 km
P-9	June 12, 2015	TORC	37.8 km	P-10	June 12, 2015	Fusion	37.9 km
P-11	July 24, 2015	TORC	37.9 km	P-12	July 24, 2015	Fusion	37.9 km
						Total:	525.72 km

Figure 2.7: *PG14 Dataset*: a dataset of 14 manually driven loops near Ann Arbor, Michigan, covering **P**lymouth Road, **G**otfredson Road, and M-**14** highway. This dataset was collected over a span of 3 months and covers residential roads (*blue*), rural roads (*purple*), and a highway (*red*). Moreover, we observed 3 construction zones over data acquisition (*orange*). The two zones on the right resulted in full repavings that included the addition of left turn lanes that were not completed until P-11 and P-12. In our evaluation with this dataset, P-M1 and P-M2 were used for map construction.

Ann Arbor—Downtown			
ID	Date	Platform	Length
D-M1	Nov. 19, 2013	TORC	3.0 km
D-M2	Nov. 19, 2013	TORC	3.0 km
D-1	Nov. 20, 2013	TORC	3.0 km
D-2	Nov. 20, 2013	TORC	3.0 km
D-3	Dec. 17, 2013	TORC	3.0 km
			Total: 14.92 km

Figure 2.8: *Downtown Dataset*: a dataset of 5 manually driven loops through downtown Ann Arbor, Michigan, covering urban driving scenarios. D-3 was collected while heavy snow was falling and covered significant portions of the roadway. In our evaluation with this dataset, D-M1 and D-M2 were used for map construction.

the accuracy of this ground-truth is an order of magnitude better than our localization errors and was manually verified by viewing the consistency of resulting LIDAR data reprojected into this common *map* reference frame.

2.6.2 Map Parameter Selection

This section intends to analyze the impacts of varying map parameters, both in terms of resulting localization errors as well as disk space requirements for storing maps. The primary user configurable map settings include: (*i*) grid resolution, (*ii*) number of Gaussians per cell, and (*iii*) reflectivity maps that contain full 3D appearance ($\mathcal{G}_r$) or appearance of the ground-plane only ($\mathcal{G}_{r,\texttt{grd}}$). Moreover, these first two settings can be tuned for each map type (structure versus appearance). All of these variabilities lead to a wide search space of possible map combinations.

To fully experiment and determine the optimal map parameters, we constructed 360 maps to run evaluation over (180 using the *PG14 Dataset* and 180 using the *Downtown Dataset*). Each of these sets of 180 maps were made by varying 12 grid resolutions (6.4 cm, 8.0 cm, 12.8 cm, 16.0 cm, 25.6 cm, 32.0 cm, 51.2 cm, 64.0 cm, 80.0 cm, 128.0 cm, 160.0 cm, and 256.0 cm), 5 different number of Gaussians per grid cell (1–5), and were generated for our 3 map types, ($\mathcal{G}_z$, $\mathcal{G}_r$, and $\mathcal{G}_{r,\texttt{grd}}$).

In our implementation, we store our maps on disk in 64 m $\times$ 64 m tiles. Thus, grid resolutions considered here were chosen to evenly divide these tiles. Moreover, to efficiently generate hundreds of maps, we construct our sparse histogram map representations at the

6.4 cm and 8.0 cm resolutions only, then build the remaining maps using these histograms (e.g., the 25.6 cm map is constructed by pooling a 4×4 window of the 6.4 cm histogram).

As a first reference to qualitatively demonstrate what is being captured in our maps, we looked at 5 versions of $\mathcal{G}_z$, $\mathcal{G}_r$, and $\mathcal{G}_{r,\texttt{grd}}$, at a resolution of 25.6 cm, in which we varied the number of Gaussians per cell from 1–5. Snapshots of these maps are visually depicted in Fig. 2.9, Fig. 2.10, and Fig. 2.11, respectively.

Within the z-height map, Fig. 2.9, we immediately see that increasing the number of Gaussians leads to overfitting. This is clear in the ground-plane and superstructure where many components share the same mean. There is a significant qualitative improvement from a 1-Gaussian map to a 2-Gaussian map. In the 1-Gaussian case, we see that there's a necessary blurring between ground and superstructure (trees, lightposts, etc.), while the 2-Gaussian case can easily capture a mode near the ground-plane and a mode covering superstructure. This trend continues through higher fidelity maps as we see the ground is captured in the lowest mean component and the increase in number of Gaussians allows for more overfitting to building facades and other superstructure. Keeping these modes separate and distinct is important for discarding obstacles that may appear in the void between ground and structure, thus we expect there to be a noticeable localization improvement between the 1-Gaussian and higher number maps.

Looking at the reflectivity maps, Fig. 2.10 and Fig. 2.11, we again notice overfitting beyond 2 Gaussians. Using 2 or more Gaussians is necessary for $\mathcal{G}_r$, as there appears to be two distinct modes per cell: the appearance of the ground and the appearance of above ground features. In both maps, multiple Gaussian components allows us to better capture edge effects (transitions from asphalt to road paint), where ground-plane road paint is typically much smaller than the 25.6 cm grid resolution. The state-of-the-art method (Levinson and Thrun, 2010), depicted as the 1-Gaussian ground only map in Fig. 2.11, leads to a more washed out image as these edges blur between high and low reflectivity—thus capturing a large Gaussian variance in these cells.

2.6.2.1 Map Parameter Sweep

We performed a series of evaluation over these 360 maps constructed in which we hold the experimental log fixed while evaluating against each map. Thus, we used P-3 to test against *PG14 Dataset* maps and D-2 to evaluate against *Downtown Dataset* maps. Further, we test using each map type independently ($\mathcal{G}_z$, $\mathcal{G}_r$, $\mathcal{G}_{r,\texttt{grd}}$), assuming that the resulting combination of the structure and appearance maps will yield more robust measurements, without directly optimizing over the exorbitant number of cross possibilities between all map types.

To benchmark our registration quality we took known ground-truth for our evaluation

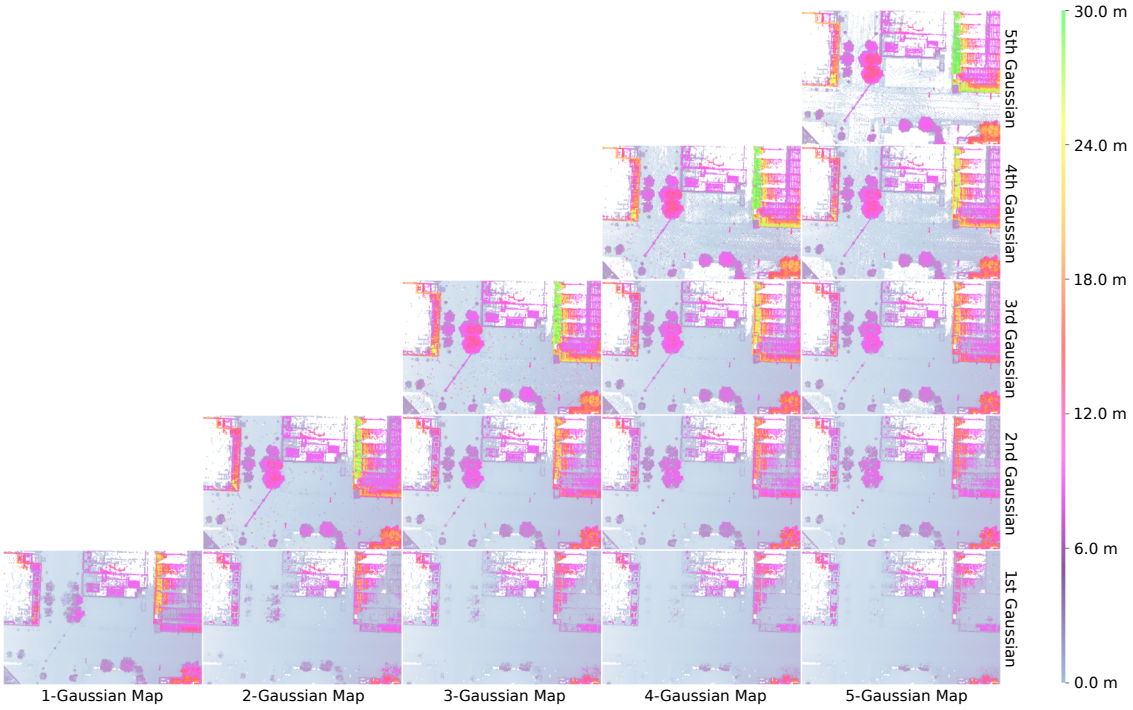


Figure 2.9: Gaussian mixture map components contained within the z-height map, $\mathcal{G}_z$, capturing ground-plane, buildings, trees, lightposts, and traffic lights. Left-to-right, each column in this figure represents a different map, ranging from a Gaussian mixture map containing only 1 component per cell to one containing 5 components per cell. Bottom-to-top, we display the mean of the i^{th} Gaussian component, ordered by increasing component mean (z-height); white cells indicate no Gaussian mixture component exists because no data was available during mapping or the Gaussian mixture resulting weight was less than 0.001.

log and generated a randomized offset every 40 m of road travel. This random offset was sampled uniformly within 2.5 m×2.5 m of the ground-truth pose. This randomly sampled point can then be viewed as the initial guess, T_0, into our Gaussian mixture map registration framework. The expectation is that the resulting registration event will converge on the ground-truth pose.

Results for the parameter sweep over $\mathcal{G}_z$, $\mathcal{G}_r$, and $\mathcal{G}_{r,\texttt{grd}}$ are presented in Fig. 2.12, Fig. 2.13, and Fig. 2.14, respectively. These figures show the longitudinal and lateral median absolute deviation with respect to ground-truth, and results are divided between *downtown*, *highway*, and *other* (encompassing rural and residential roads) portions, along with a summary over *all* roadways. Median absolute deviation was chosen over other statistics as it is a robust, outlier-proof measure of variability.

As expected, we see that error grows as a function of coarser grid resolution across all map types, and we are constrained far better laterally than we are longitudinally. Furthermore,

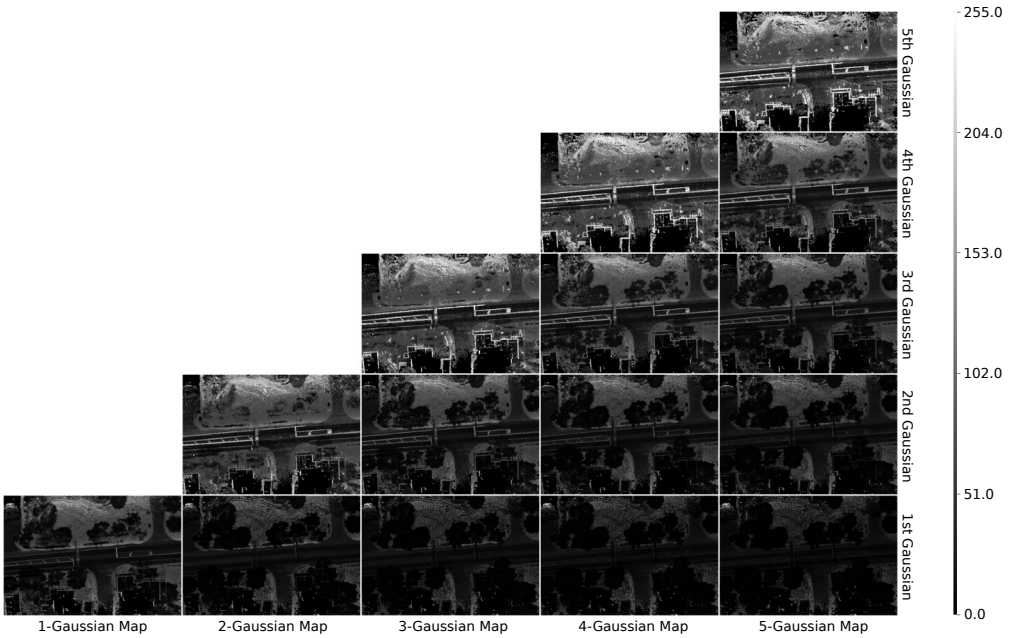


Figure 2.10: Gaussian mixture map components contained within the reflectivity map, $\mathcal{G}_r$, capturing appearance of ground-plane, foliage, etc. Left-to-right, each column in this figure represents a different map, ranging from a Gaussian mixture map containing only 1 component per cell to one containing 5 components per cell. Bottom-to-top, we display the mean of the i^{th} Gaussian component, ordered by increasing component mean (reflectivity).

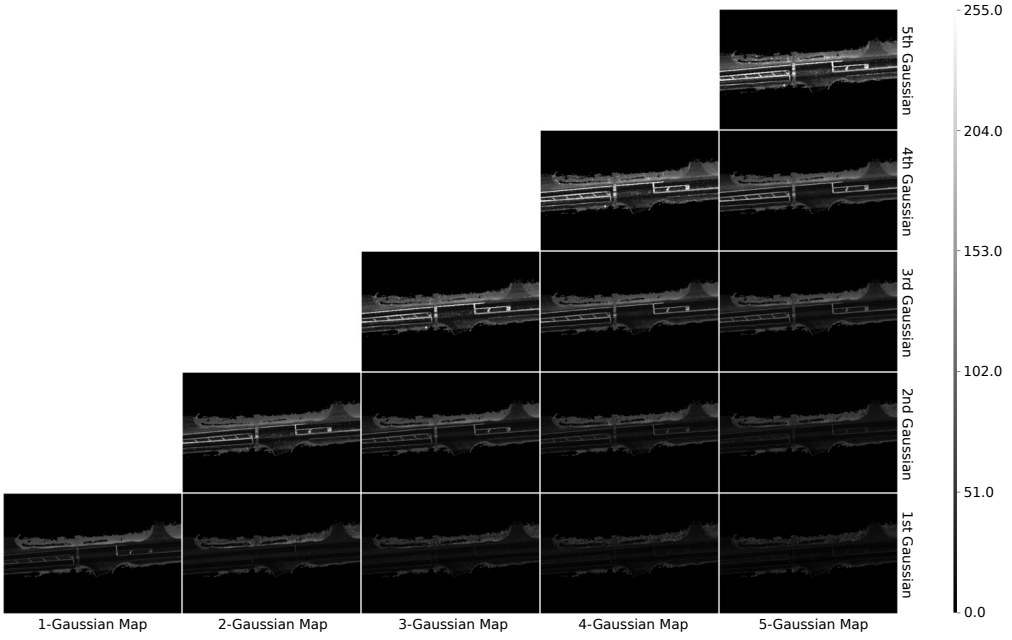


Figure 2.11: Gaussian mixture map components contained within the reflectivity map, $\mathcal{G}_{r,\mathtt{grd}}$, capturing appearance of ground-plane *only*. See Fig. 2.10 for more description.

we see that all map types perform markedly better on the *downtown* portions, as can be expected given the significant structure and well maintained road paint. Across most of these plots, we notice that there are significant error spikes at very fine resolutions. This is believed to be caused by overfitting as there is simply not enough training data to accurately learn the Gaussian mixture maps.

Looking at the evaluation over $\mathcal{G}_z$, we see a noticeable improvement from 1 to 2+ Gaussians in both lateral and longitudinal error. As predicted in the previous section, the ability to rule out obstacles between superstructure and ground-plane plays an important role here.

Considering the evaluation over $\mathcal{G}_r$, it is clear that 2+ Gaussians is necessary as anticipated—given that 2 modes are needed to capture ground-plane appearance and above-ground appearance. On the contrary, $\mathcal{G}_{r,\texttt{grd}}$ performs best with a single Gaussian, though only at fine resolutions. However, as grid cell size is increased, it is necessary to use 2+ Gaussians so that features are not blurred away.

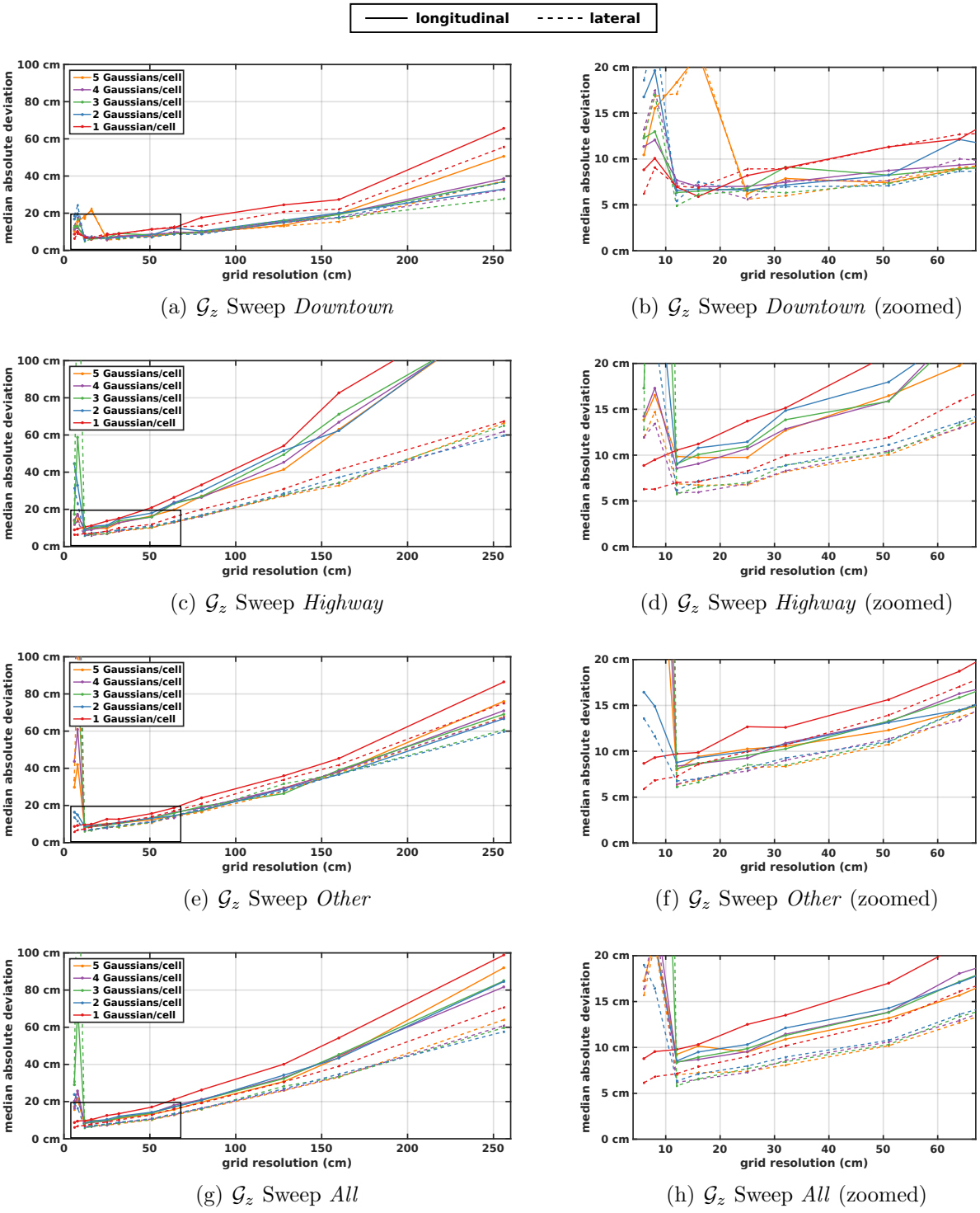
Throughout all of these sweeps, it is not immediately clear that more than 2 Gaussians is necessary as there is not a significant performance improvement by doing so.

Map Size: In addition to performance metrics, in many cases map parameter selection must also consider the required disk space for map storage. In Fig. 2.15, we look at the corresponding disk space required per km of map data. As expected, finer resolution maps get exponentially larger relative to coarser grid resolutions. Additionally, ground-only reflectivity maps are significantly smaller than those constructed using all points—this is because ground-only maps are restricted to areas within a few meters of the roadway, while full maps can include points over 50 meters away.

For our maps, all Gaussian mixture components (i.e., weight, mean, variance) are stored as single-precision floating point values and the maps are compressed using gzip. Therefore, more intricate compression schemes can be used and map sizes presented here should be viewed as a worst case scenario.

We envision that our maps can be streamed to an autonomous car, where our 64 m $\times$ 64 m tiles can be continuously downloaded over a 4G connection. We assume a network bandwidth of nominally 2 MBps and a vehicle certainly traveling less than 150 kph. Thus, we have an available streaming budget of roughly 48.0 MB/km.

Considering this budget, we decided on a 2-Gaussian, 25.6 cm z-height map ($\mathcal{G}_z$), and a 1-Gaussian, 6.4 cm ground-only reflectivity map ($\mathcal{G}_{r,\texttt{grd}}$). Note that our reflectivity map selection is roughly the same as Levinson's probabilistic appearance maps. The combination of these two maps found the best balance between performance, while falling under our required budget at roughly 44.3 MB/km. We found that the superior performance of the



(a) $\mathcal{G}_z$ Sweep *Downtown*

(b) $\mathcal{G}_z$ Sweep *Downtown* (zoomed)

(c) $\mathcal{G}_z$ Sweep *Highway*

(d) $\mathcal{G}_z$ Sweep *Highway* (zoomed)

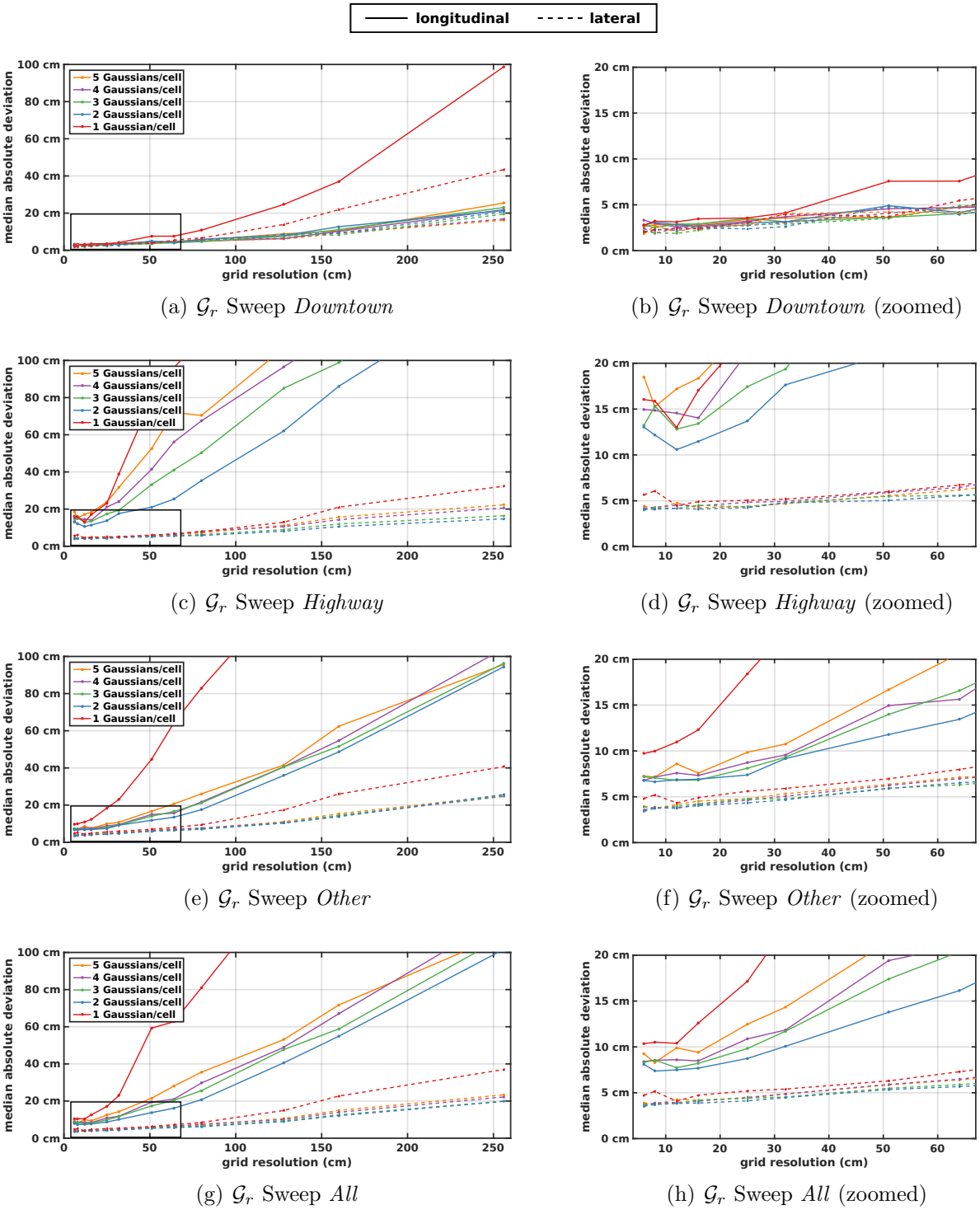
(e) $\mathcal{G}_z$ Sweep *Other*

(f) $\mathcal{G}_z$ Sweep *Other* (zoomed)

(g) $\mathcal{G}_z$ Sweep *All*

(h) $\mathcal{G}_z$ Sweep *All* (zoomed)

Figure 2.12: Summary of parameter sweep over grid resolution and number Gaussians in z-height Gaussian mixture maps, $\mathcal{G}_z$, and the resulting longitudinal (solid lines) and lateral (dashed lines) median absolute deviation (MAD)—partitioned into *downtown*, *highway*, *other* (rural and residential), and a summary over *all* data.



(a) $\mathcal{G}_r$ Sweep *Downtown*

(b) $\mathcal{G}_r$ Sweep *Downtown* (zoomed)

(c) $\mathcal{G}_r$ Sweep *Highway*

(d) $\mathcal{G}_r$ Sweep *Highway* (zoomed)

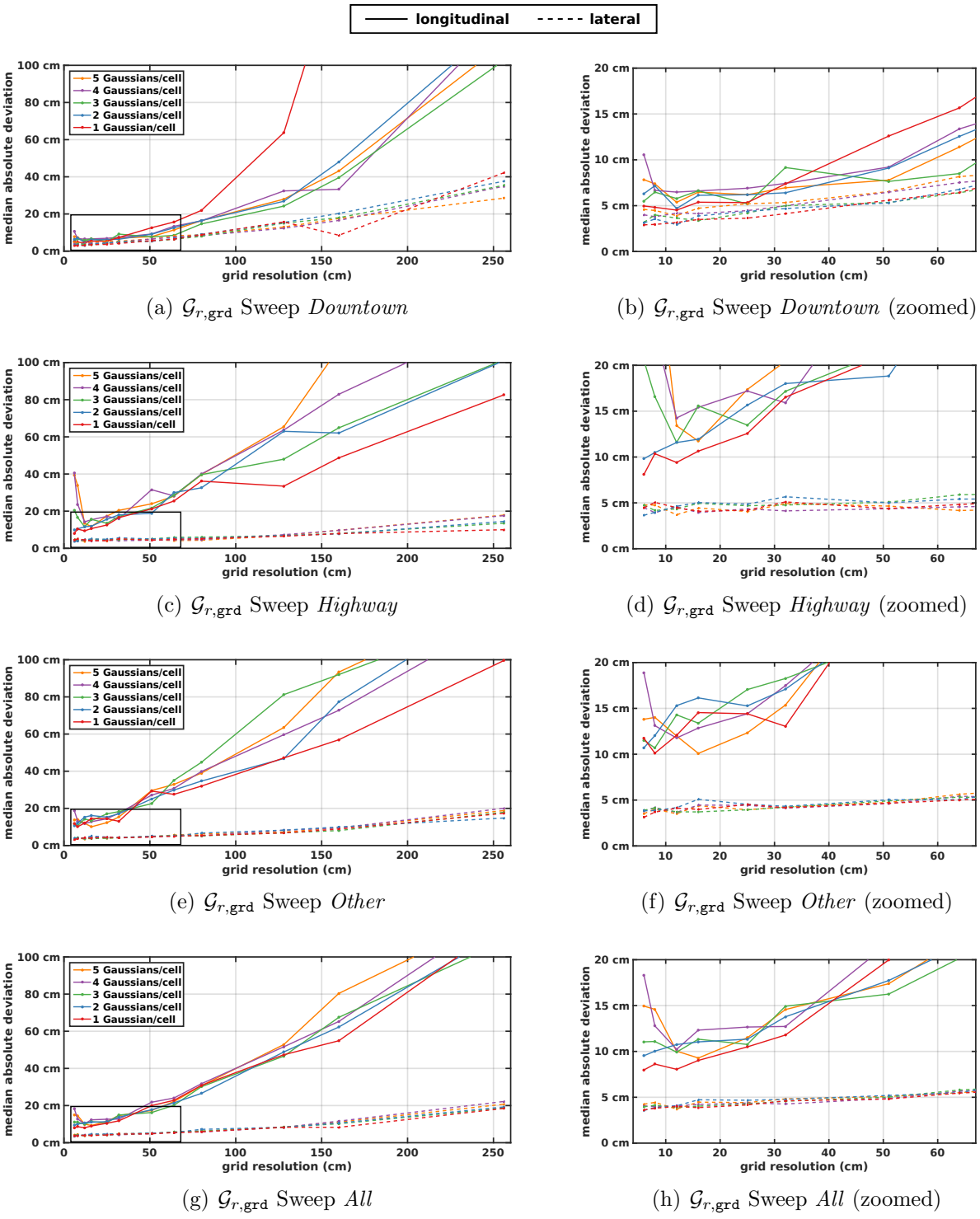
(e) $\mathcal{G}_r$ Sweep *Other*

(f) $\mathcal{G}_r$ Sweep *Other* (zoomed)

(g) $\mathcal{G}_r$ Sweep *All*

(h) $\mathcal{G}_r$ Sweep *All* (zoomed)

Figure 2.13: Summary of parameter sweep over grid resolution and number Gaussians in reflectivity Gaussian mixture maps, $\mathcal{G}_r$, and the resulting longitudinal (solid lines) and lateral (dashed lines) median absolute deviation (MAD)—partitioned into *downtown*, *highway*, *other* (rural and residential), and a summary over *all* data.



(a) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *Downtown*

(b) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *Downtown* (zoomed)

(c) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *Highway*

(d) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *Highway* (zoomed)

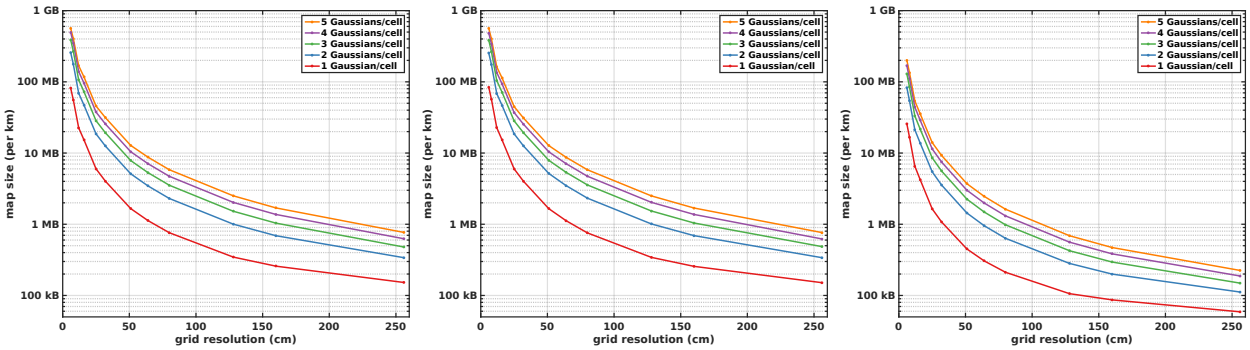
(e) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *Other*

(f) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *Other* (zoomed)

(g) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *All*

(h) $\mathcal{G}_{r,\texttt{grd}}$ Sweep *All* (zoomed)

Figure 2.14: Summary of parameter sweep over grid resolution and number Gaussians in ground-plane only, reflectivity Gaussian mixture maps, $\mathcal{G}_{r,\texttt{grd}}$, and the resulting longitudinal (solid lines) and lateral (dashed lines) median absolute deviation (MAD)—partitioned into *downtown*, *highway*, *other* (rural and residential), and a summary over *all* data.



(a) $\mathcal{G}_z$ (b) $\mathcal{G}_r$ (c) $\mathcal{G}_{r,\texttt{grd}}$

Figure 2.15: Map size for Gaussian mixture maps over z-height ($\mathcal{G}_z$), reflectivity ($\mathcal{G}_r$), and ground-plane reflectivity ($\mathcal{G}_{r,\texttt{grd}}$). Sizes are listed as a function of Gaussian mixture map grid resolution and number of Gaussians per cell. All sizes are per kilometer of road travel.

ground-only reflectivity maps on the highway are an added benefit as z-height is least effective there.

The remainder of this chapter will perform experiments over this map configuration alone.

2.6.3 Registration Experiments

Since our odometry source has significantly low drift-rates, registration deficiencies can be masked by a well-tuned filtering framework. Thus, this section looks directly at evaluating the unfiltered registrations that exploit structure and appearance within the vicinity of ground-truth results.

Identical in setup to our parameter sweep discussed in the previous section, we now look at a sweep over all logs in our dataset while holding map settings fixed—showing that our map is robust for localizing over time. Again we randomly sample a point uniformly within 2.5 m $\times$ 2.5 m of ground-truth every 40 m, and evaluated the resulting transformation from our multiresolution registration framework relative to this ground-truth. To fully understand the contributions of each map type, we perform 3 registrations per ground-truth sample: (*i*) using structure alone ($\mathcal{G}_z$), (*ii*) using appearance alone ($\mathcal{G}_{r,\texttt{grd}}$), and (*iii*) using structure and appearance jointly ($\mathcal{G}_z$, $\mathcal{G}_{r,\texttt{grd}}$).

Errors are summarized per data log in Fig. 2.16, where we show median absolute deviation bars for longitudinal and lateral errors, along with first and third quartile error whiskers. Over most of *PG14 Dataset*, we see high longitudinal errors when using reflectivity alone that becomes well constrained with the addition of 3D structure. In most logs, we see that the joint cost function yields an improvement in our registrations. However, in the case of

the *Downtown Dataset*, it is not surprising that the joint cost function is heavily dictated by z (seeming to ignore the more accurate reflectivity measurements) because the significant number of point returns off of 3D structure that dominate the cost function.

Results are summarized for all datasets in Table 2.1.

	Longitudinal		**Lateral**	
Map	Median	$1^{\text{st}}/3^{\text{rd}}$ Qtr.	Median	$1^{\text{st}}/3^{\text{rd}}$ Qtr.
$\mathcal{G}_z$	8.5 cm	(3.8 cm, 16.7 cm)	8.0 cm	(3.4 cm, 15.9 cm)
$\mathcal{G}_{r,\texttt{grd}}$	10.9 cm	(3.8 cm, 35.0 cm)	4.6 cm	(2.0 cm, 8.6 cm)
$\mathcal{G}_z$,$\mathcal{G}_{r,\texttt{grd}}$	7.7 cm	(3.4 cm, 15.2 cm)	5.3 cm	(2.4 cm, 9.8 cm)

Table 2.1: Comparison of errors between Gaussian mixture map types, showing the median, first quartile, and third quartiles of absolute deviation (longitudinally and laterally).

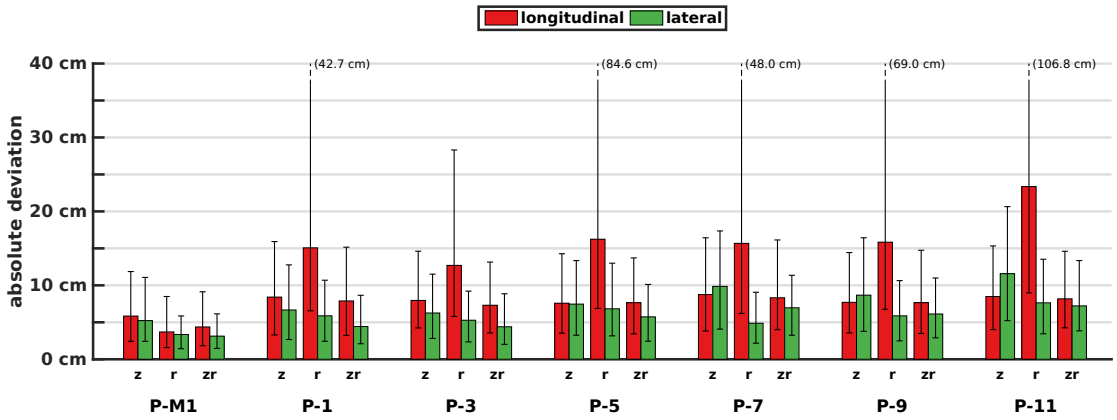
2.6.3.1 Heavy Snowfall Registrations

We more thoroughly looked at z-height registrations alone in the snow-filled *Downtown Dataset*, D-3, by randomly sampling within 10 m of the ground-truth pose. We present these results in two ways. First, we compiled the results into a histogram, as shown in the top row of Fig. 2.17(b). Here we see that our proposed solution is able to return to within 25 cm of the ground-truth with minimal outliers. Additionally, we see that because our method exploits the 3D structure, it is not impacted by harsh weather and significant amounts of falling snow, as depicted in Fig. 2.17(a).

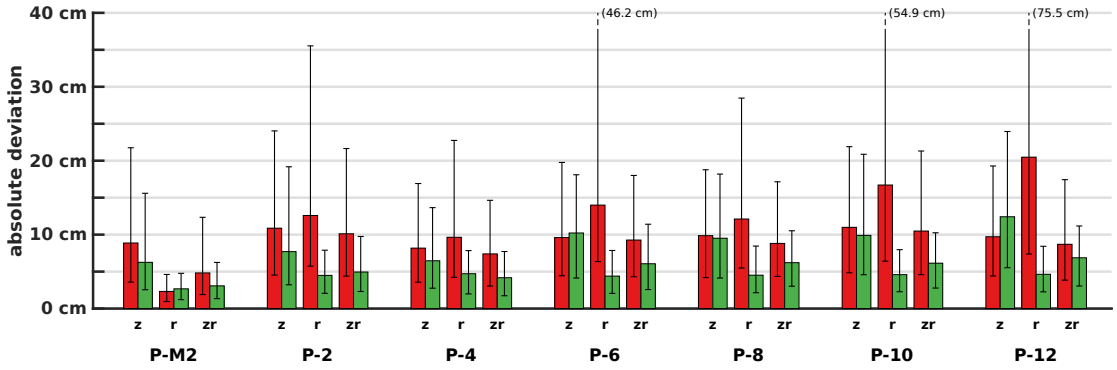
Second, we display this same registration error as a function of initial offset input to the scan matcher, as displayed in the bottom row of Fig. 2.17(b). We show that our registration success is not dictated by distance from the optimum, as long as our search space is able to enclose the true transformation.

2.6.3.2 Construction Zone Registrations

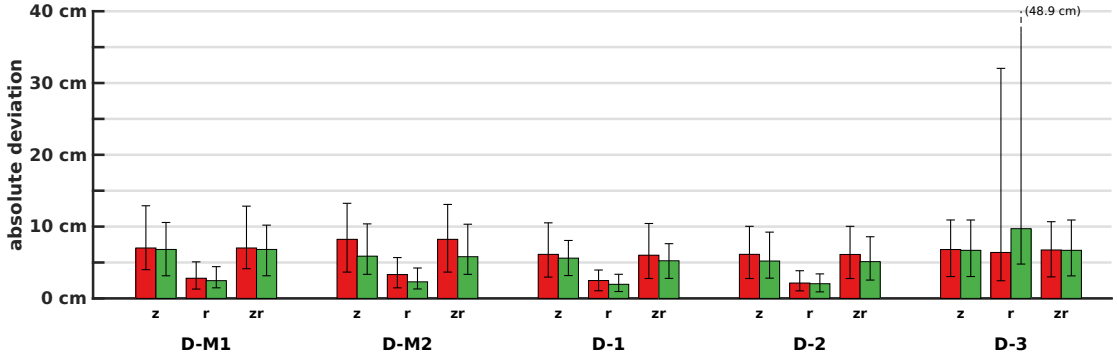
We further look at registration errors through one of the three construction zones that was repaved during our dataset collection, see Fig. 2.18. In this figure, we display our single map containing $\mathcal{G}_z$ and $\mathcal{G}_{r,\texttt{grd}}$ built using P-M1 and P-M2 data in Fig. 2.18(a). Figures (b)-(d) shows registration results evaluated against $\mathcal{G}_z$, $\mathcal{G}_{r,\texttt{grd}}$, and the joint measurement over both. Further, each of these figures are drawn over a reference reflectivity map that was built using data on each day to demonstrate the changes over time—note that the map in (a) was still used for all experiments.



(a) *PG14 Dataset*: TORC Logs

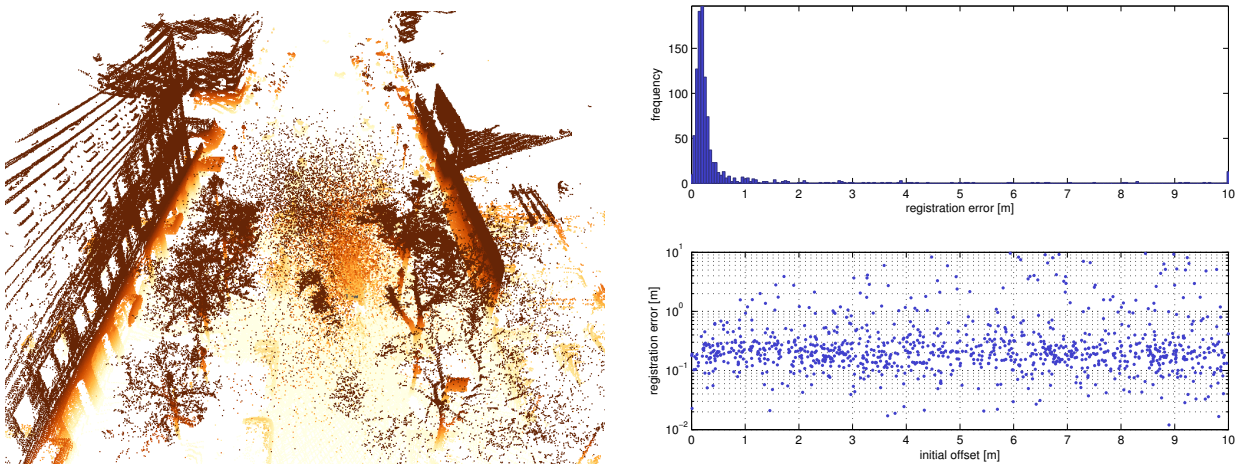


(b) *PG14 Dataset*: Fusion Logs



(c) *Downtown Dataset*

Figure 2.16: This figure shows the registration errors from randomly sampled points along each log using $\mathcal{G}_z$, $\mathcal{G}_{r,\texttt{grd}}$, and $\{\mathcal{G}_z, \mathcal{G}_{r,\texttt{grd}}\}$, marked by **z**, **r**, and **zr**, respectively. Bars indicate the longitudinal (*red*) and lateral (*green*) median absolute deviation (MAD) and the error whiskers mark the first and third quartiles of absolute deviation. In most logs, the joint likelihood measure over structure and appearance yields improved performance relative to the likelihood measure over structure or appearance alone. Moreover, the use of structure prevents large longitudinal errors during the *PG14 Dataset* and allows for consistent localization during the snow dataset, D-3. The reflectivity alone does quite well in some circumstances, such as the *Downtown Dataset* where road paint is well maintained, though the joint measure still results in errors less than 10 cm.



(a) Heavy Snowfall Point Cloud

(b) *Downtown Dataset*—D-3

Figure 2.17: In (a), we show a point cloud rendering of typical snowfall (with ground-plane removed) during the D-3 dataset. Orange and brown points located at the center of the figure shows the dense snow returns. In (b), we demonstrate registration error using z-height alone on the snow-filled dataset, D-3. The top row shows a histogram of our L^2 error, demonstrating good registration performance. The bottom row shows a plot of initial offset versus registration error, where we show that our scan matching errors are independent of initial guess.

Even before construction began in Fig. 2.18(b), reflectivity poorly constrains longitudinally due to limited features—the method relies on cutouts into driveways and roads for success. Over time, the ability to localize using reflectivity alone becomes impossible because the appearance is fundamentally different; however, localization using $\mathcal{G}_z$ remains effective and the joint measurement is not distracted by erroneous reflectivity measurements.

2.6.4 Filtered Experiments

We integrated our registration algorithm into the EKF localization framework described in Section 2.5. The only measurements used were those from a GPS unit for initialization, our IMU for vehicle odometry, and our multiresolution scan matches considering structure and appearance initialized around our 4σ posterior belief. Standard deviation for these scan registrations was set relative to our median absolute deviation derived in the previous section, $1.4826 \cdot \text{MAD}$; this scaling is so that MAD can be viewed as a consistent estimator of normally distributed variance (Rousseeuw and Croux, 1993). Further, we evaluate the measurement NIS and only include measurements that are 99% likely to be consistent with our filter—this allows our filtering to be robust to outliers.

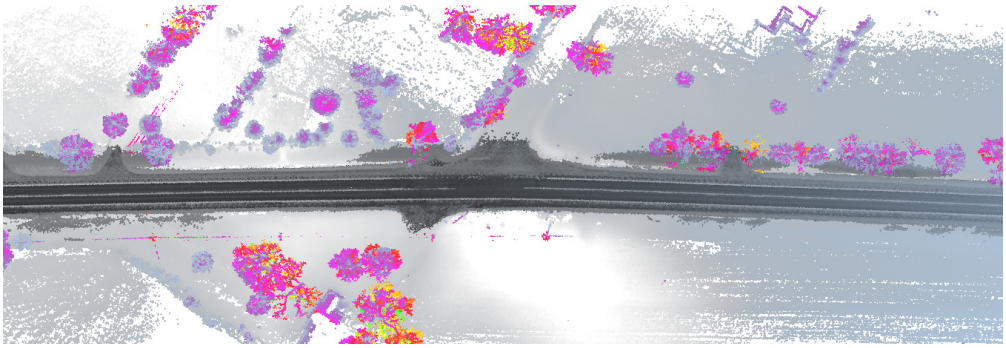
Results are tabulated in Table 2.2, where we present longitudinal, lateral, and heading errors relative to ground-truth. Errors are shown in terms of RMS errors as well as percentages of filtered poses that fall within 5 cm, 25 cm, and 1 m (longitudinally and laterally). Overall, we see that our measurements result in better constraints laterally than longitudinal; lateral RMS errors are typically less than 10 cm and longitudinal RMS errors are within the range of 10–13 cm.

Further, we demonstrate each log graphically over satellite imagery in Fig. 2.19, Fig. 2.20, and Fig. 2.21, where each log's trajectory is colored by L_2 error. We see that errors are frequently along highway and rural roads where longitudinal constraints become dependent on visible 3D structure (unconstrained via reflectivity as showin in Fig. 2.18). A common problem occurs when passing large semi-trailer trucks that fully occlude field of view of informative 3D structure beside the road, often leading to noisy measurements.

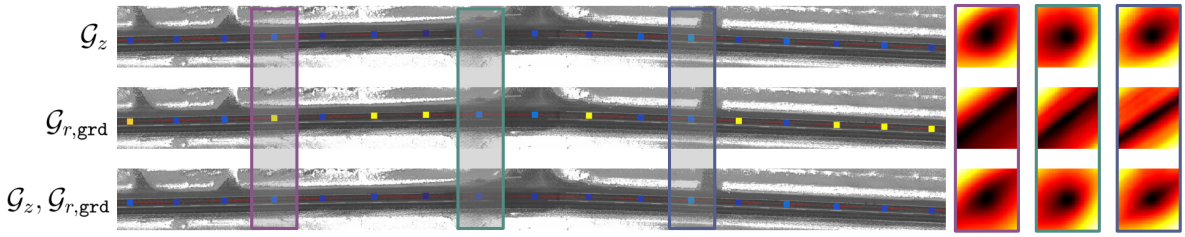
Additionally, we see our method is robust to radical appearance changes. This can be seen Fig. 2.19(g) and Fig. 2.20(g) where our method can remain localized through repavings that completely altered the appearance of 0.5-1.0 km stretches of road. There are occasional spikes of inaccuracy through these regions, though we still maintain localization through these periods of drastic appearance changes. Moreover, we demonstrate in Fig. 2.21(e) that we are able to remain localized through heavy snowfall that was present during the D-3 log.

	RMS Error			p(err<5 cm)		p (err<25 cm)		p(err<1 m)	
Session	Long.	Lat.	Hdg.	Long.	Lat.	Long.	Lat.	Long.	Lat.
P-M1	10.1 cm	6.5 cm	0.09 °	60.9 %	58.3 %	96.8 %	99.4 %	100.0 %	100.0 %
P-M2	6.6 cm	5.4 cm	0.09 °	68.6 %	69.7 %	99.1 %	99.8 %	100.0 %	100.0 %
P-1	13.3 cm	8.4 cm	0.13 °	33.2 %	56.9 %	94.3 %	98.4 %	100.0 %	100.0 %
P-2	10.3 cm	7.6 cm	0.11 °	40.3 %	56.1 %	98.0 %	99.3 %	100.0 %	100.0 %
P-3	10.7 cm	8.2 cm	0.13 °	41.0 %	50.4 %	97.7 %	98.9 %	100.0 %	100.0 %
P-4	10.8 cm	8.3 cm	0.16 °	38.5 %	46.0 %	98.5 %	99.4 %	100.0 %	100.0 %
P-5	11.9 cm	9.3 cm	0.20 °	36.7 %	44.4 %	96.2 %	98.4 %	100.0 %	100.0 %
P-6	8.4 cm	8.2 cm	0.15 °	46.6 %	45.3 %	99.1 %	99.4 %	100.0 %	100.0 %
P-7	12.7 cm	9.1 cm	0.12 °	34.3 %	41.5 %	94.4 %	99.0 %	100.0 %	100.0 %
P-8	12.2 cm	9.5 cm	0.12 °	45.1 %	40.6 %	94.4 %	98.5 %	100.0 %	100.0 %
P-9	16.2 cm	9.8 cm	0.13 °	34.8 %	47.3 %	91.1 %	98.1 %	100.0 %	100.0 %
P-10	12.2 cm	10.6 cm	0.12 °	41.5 %	43.0 %	95.7 %	96.6 %	99.9 %	100.0 %
P-11	15.0 cm	12.3 cm	0.15 °	30.3 %	41.4 %	92.2 %	96.8 %	99.9 %	99.9 %
P-12	11.0 cm	12.7 cm	0.12 °	40.3 %	34.3 %	97.4 %	95.6 %	100.0 %	100.0 %
D-M1	10.7 cm	8.9 cm	0.17 °	30.0 %	43.1 %	98.3 %	99.3 %	100.0 %	100.0 %
D-M2	12.5 cm	10.3 cm	0.17 °	30.4 %	50.9 %	96.4 %	97.1 %	100.0 %	100.0 %
D-1	10.8 cm	9.8 cm	0.18 °	39.7 %	39.4 %	98.7 %	97.3 %	100.0 %	100.0 %
D-2	11.6 cm	9.1 cm	0.16 °	26.8 %	45.6 %	95.4 %	98.2 %	100.0 %	100.0 %
D-3	11.8 cm	10.9 cm	0.15 °	42.5 %	36.4 %	96.8 %	97.7 %	100.0 %	100.0 %

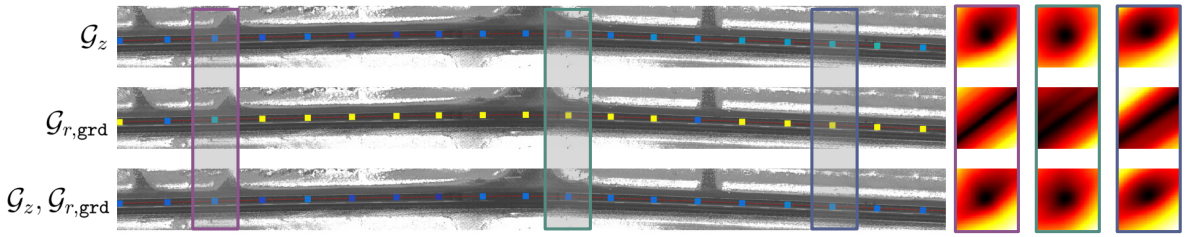
Table 2.2: Filtered results using joint measurements over structure and appearance for all datasets. We show longitudinal, lateral and heading RMS errors in addition to percentage of filtered poses that are within 5 cm, 25 cm, and 1 m of ground-truth, longitudinally and laterally. Top half of table tabulates results over the *PG14 Dataset*, and the bottom half covers the *Downtown Dataset*.



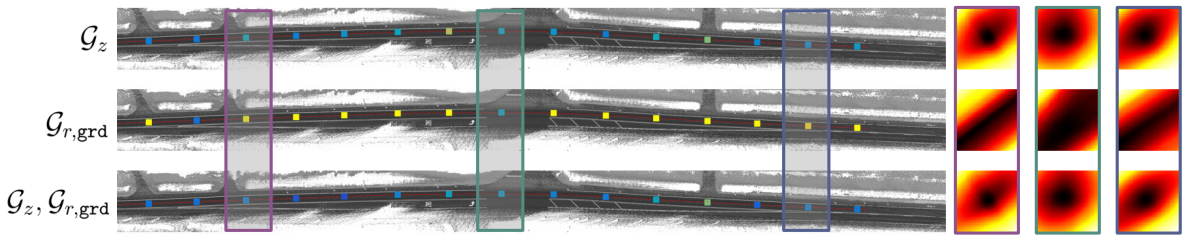
(a) $\mathcal{G}_z$ and $\mathcal{G}_{r,\text{grd}}$ Maps



(b) Fusion, May 8

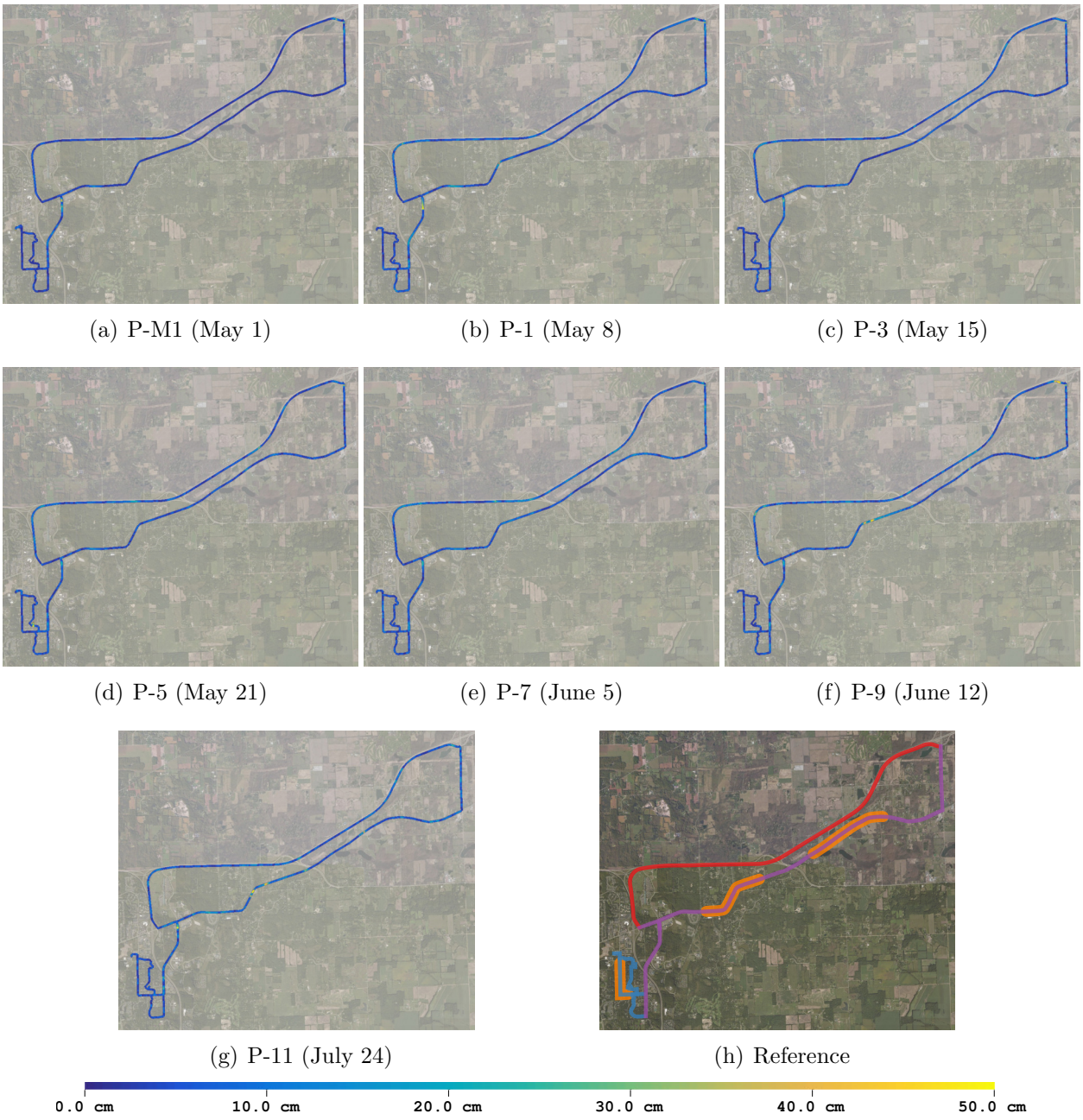


(c) Fusion, June 5



(d) Fusion, July 24

Figure 2.18: This figure demonstrates the registration quality of our joint measurement function over a segment of road that is poorly constrained by appearance *and* undergoes significant construction. In (a), we show the map constructed for our experiments (showing the maximum mean component), visualizing the structure (*purple-green*) and appearance (*black/white*) together. In (b)-(d), we show registrations performed over each map type spanning different logs, rendering: a map that reflects the appearance *on that day*, registration L_2 error (colored dots), and sample cost function sweeps (right). Despite appearance cost functions that poorly constrain our pose and radically change, our z-height measurements and resulting joint measurements remain well constrained.

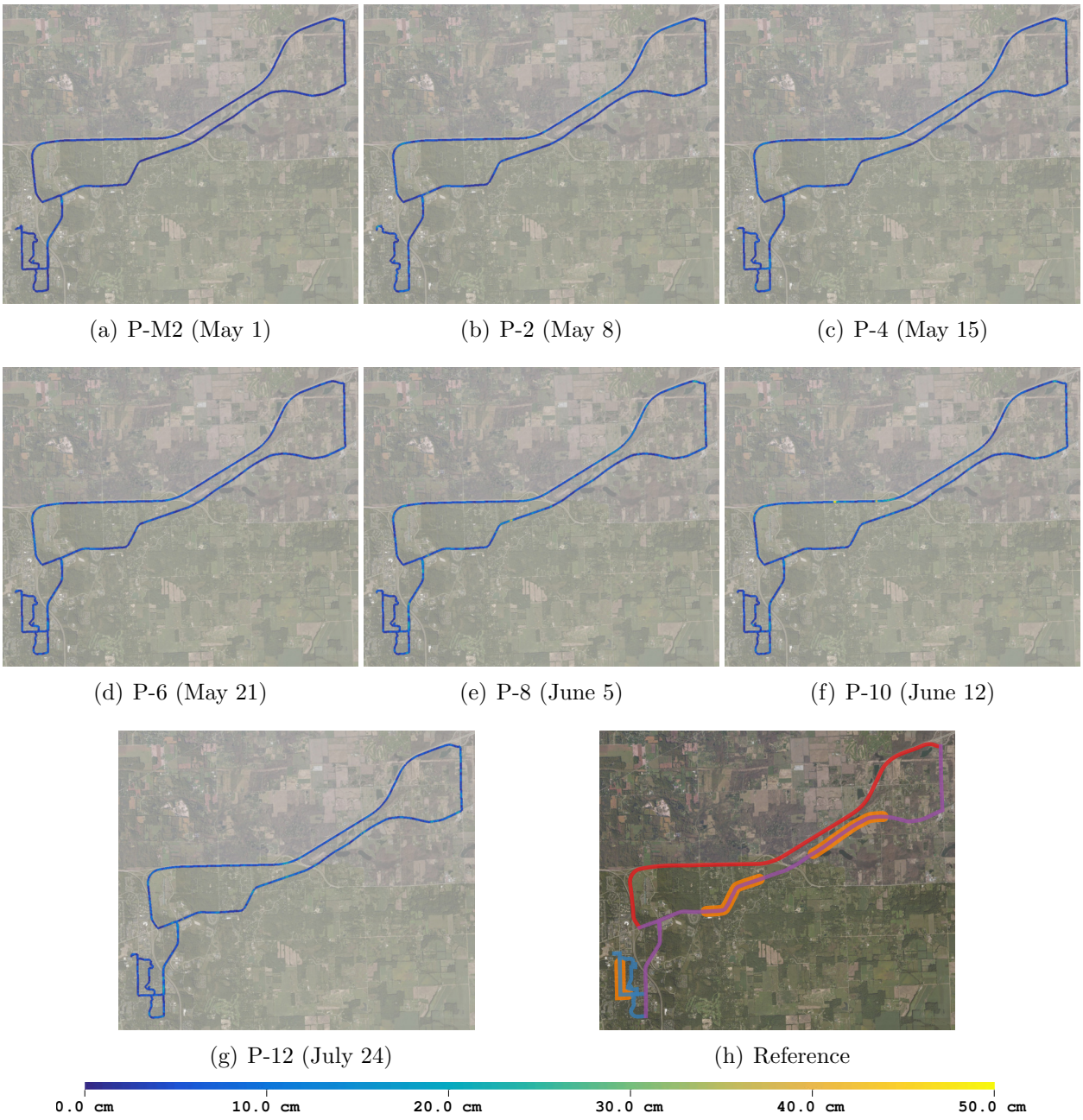


(a) P-M1 (May 1) (b) P-1 (May 8) (c) P-3 (May 15)

(d) P-5 (May 21) (e) P-7 (June 5) (f) P-9 (June 12)

(g) P-11 (July 24) (h) Reference

Figure 2.19: Filtered results using joint measurements over structure and appearance for the TORC logs of the *PG14 Dataset*, where the trajectory is colored by L_2 error. Most noticeable errors are longitudinal—primarily on the highway or other long, straight stretches with little variation in that dimension. Further, note the slight increase in errors through construction zones. Despite these increases, our filter does not diverage and remains localized within acceptable tolerances. A route reference is provided in (h) highlighting residential roads (*blue*), rural roads (*purple*), highways (*red*), and construction zones (*orange*).



(a) P-M2 (May 1) (b) P-2 (May 8) (c) P-4 (May 15)

(d) P-6 (May 21) (e) P-8 (June 5) (f) P-10 (June 12)

(g) P-12 (July 24) (h) Reference

Figure 2.20: Filtered results using joint measurements over structure and appearance for the Fusion logs of the *PG14 Dataset*, where the trajectory is colored by L_2 error. Most noticeable errors are longitudinal—primarily on the highway or other long, straight stretches with little variation in that dimension. Further, note the slight increase in errors through construction zones. Despite these increases, our filter does not diverge and remains localized within acceptable tolerances. A route reference is provided in (h) highlighting residential roads (*blue*), rural roads (*purple*), highways (*red*), and construction zones (*orange*).

(a) D-M1 (Nov. 19) (b) D-M2 (Nov. 19) (c) D-1 (Nov. 20)

(d) D-2 (Nov. 19) (e) D-3 (Dec. 17)

Figure 2.21: Filtered results using joint measurements over structure and appearance for the *Downtown Dataset*, where the trajectory is colored by L_2 error. Aside from initial convergence time for the filter (the bright *yellow* segment on the bottom of each figure), our method does quite well in the urban environment. This includes staying well localized through heavy snowfall during D-3 log.

2.6.5 Run-time Analysis

Given that our registration approach is a function of desired search space, we analyze the run-time performance at several search windows of: $1\text{ m} \times 1\text{ m}$, $2\text{ m} \times 2\text{ m}$, $4\text{ m} \times 4\text{ m}$, $8\text{ m} \times 8\text{ m}$, and $16\text{ m} \times 16\text{ m}$. Fig. 2.22 shows framerate for each of these search spaces at a *single* rotational search, in which each registration likelihood is evaluated using approximately $300,000$ points. Results are presented for the CPU implementation and we show a 30–40× speedup for both the exhaustive and multiresolution branch-and bound search when implemented on a GPU. Relative to our work presented in (Wolcott and Eustice, 2015), we no longer have to significantly downsample our point cloud to achieve real-time localization. Note the time speedup is more pronounced when more than one rotational offset is considered because the multiresolution search can short-circuit quicker.

During online performance, we initially need to perform a dense search over a window of $\sim 10\text{ m} \times 10\text{ m}$, which can be achieved in a little over a second. Over time, these search windows gradually shrink according to our posterior pose belief such that we can perform online localization using *all* point cloud points at roughly $5-10$ Hz

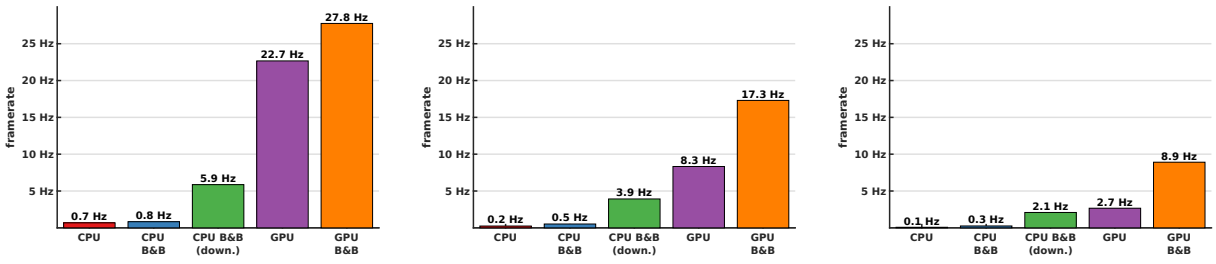
Furthermore, rasterizing the Gaussian mixture maps into multiresolution lookup tables must be carefully managed when implemented on the CPU as these take 4562 ms to construct. However, this is dramatically improved to 114 ms when implemented on a GPU (40× speedup).

2.6.6 Alternative Uses of GMM

In addition to localization, Gaussian mixture maps over z-height can be used for other purposes. In this section, we briefly present two possible use cases: point cloud compression and obstacle background subtraction.

2.6.6.1 Point Cloud Compression

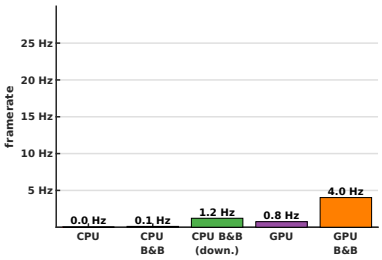
Storing raw point clouds can require more than 500 MB per km of road. As an alternative, Gaussian mixture maps over z-height can be a parametric method for compactly storing terrestrial maps. In Fig. 2.23, we show the efficacy of our method for retaining the true point cloud distribution using 1, 2, 3, 4, and 10 Gaussians per grid cell—this figure shows points that are within 2.5 standard deviations of each mixture component. It is clear that a single Gaussian per cell would be insufficient, though as few as 2 appears to well capture the building facade and foliage. These maps require roughly 10 MB, 20 MB, 30 MB, 40 MB, and 100 MB, respectively, per km of road.



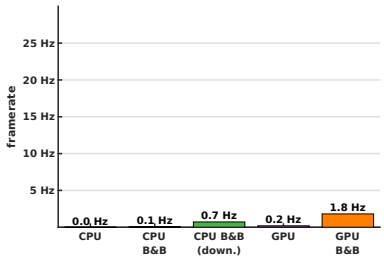
(a) 1 m×1 m

(b) 2 m×2 m

(c) 4 m×4 m



(d) 8 m×8 m



(e) 16 m×16 m

Figure 2.22: This figure shows registration framerates of our proposed localization method using Gaussian mixture maps over various search spaces. Left-to-right we show framerates using the CPU for exhaustive search (*red*), multiresolution branch-and-bound (*blue*), and again using a downsampled point cloud that was necessary in our previous work to meet acceptable localization framerates (Wolcott and Eustice, 2015) (*green*). We further show the exhaustive search (*purple*) and the branch-and-bound search (*orange*) when implemented on a GPU. Note, these figures were generated only searching over a *single* rotational offset.

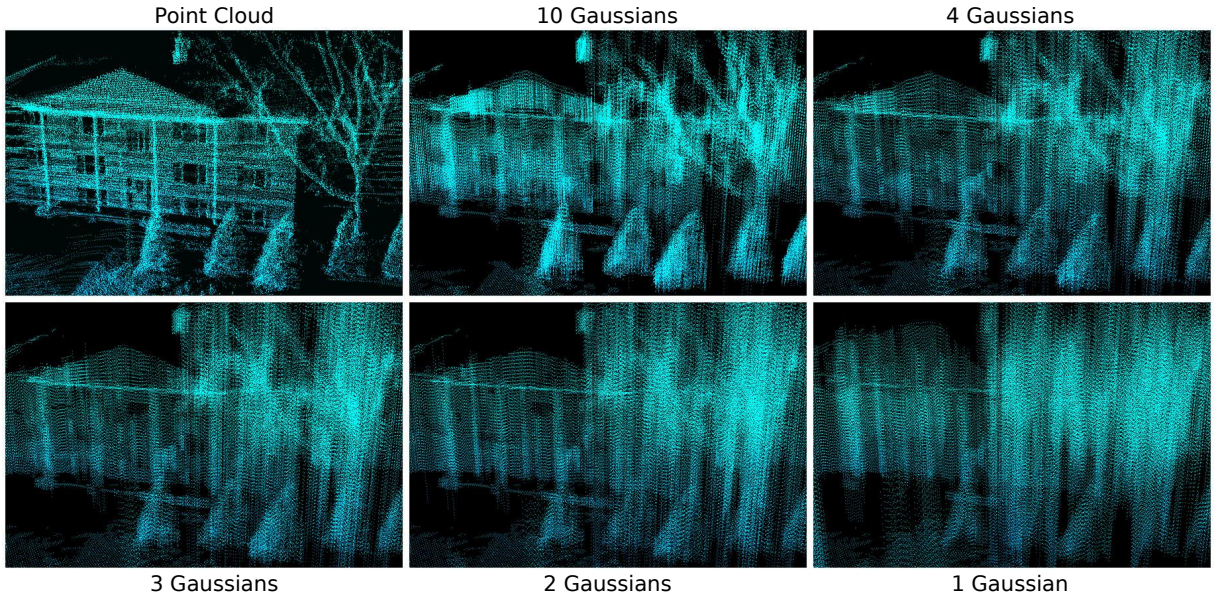


Figure 2.23: Gaussian mixture maps over z-height can be used for point cloud compression. In this figure, we demonstrate the initial point cloud, seen in the top-left and corresponding reconstructions using various $\mathcal{G}_z$ maps.

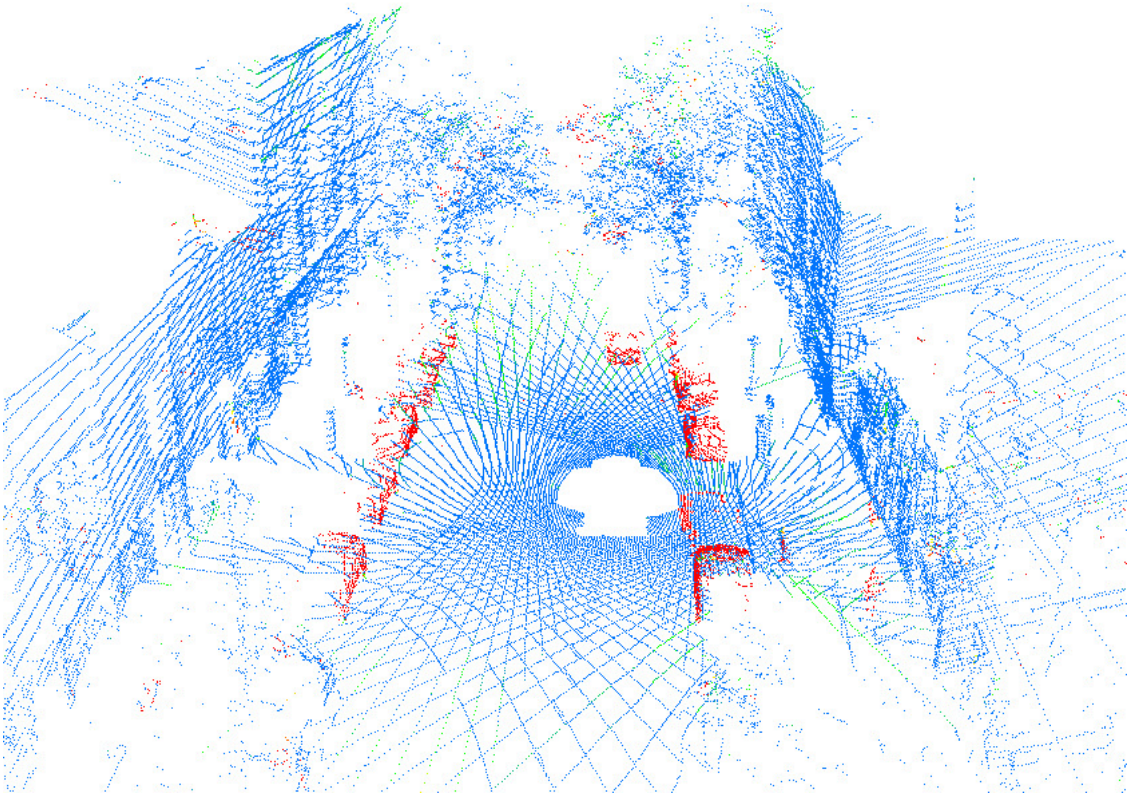


Figure 2.24: Sample point cloud colored by Mahalanobis distance from the underlying map's Gaussian mixture. Note the parked cars in *red* and agreeing prior map in *blue* (including ground-plane, building facades, trees, and lightposts). Our method allows us to expand our obstacle sensing horizon, as we can not sense the ground-plane beyond 40 m.

2.6.6.2 Obstacle Background Subtraction

Another benefit of using structure in our automated vehicle's localization pipeline is that it provides a probabilistic method to classify point cloud points as dynamic obstacles or belonging to the background environment. In generating the likelihood for a registration, we evaluate the likelihood of each scan point against the prior map, which tells us how likely each scan point is to be part of the map. By looking at points that *poorly* align to the prior map (i.e., those with low likelihoods), we can perform a classification. We do this by setting a Mahalanobis distance threshold and labeling points that exceed this threshold as obstacles—this selection is precisely the outlier thresholds set to minimize effect of outliers in our robust cost formulation of (2.9). Our formulation allows us to do this classification on a frame-by-frame basis and extend our sensing range of obstacles. Visualization of point cloud classification can be seen in Fig. 2.24.

2.7 Conclusion

In this chapter, we demonstrated Gaussian mixture maps that reduce large point clouds into a compact, parametric representation that maintains expressibility over structure and appearance. Through the use of multiresolution rasterized maps that can be computed online, we can efficiently traverse these maps to find the guaranteed optimal registration using branch-and-bound search, rather than finding local optima as with modern scan matchers. Finally, we integrated this into an EKF to demonstrate that our autonomous platform can remain well localized in a prior map over more than 500 km of road data. Our proposed system is able to handle harsh weather and poorly textured roadways, which is a significant advantage over the current state-of-the-art methodologies for automated vehicle localization. We further demonstrated that localization can be done through construction zones undergoing drastic appearance changes, allowing us in future work to consider maintaining and updating these maps as they change.

CHAPTER 3

Visual Localization within LIDAR Maps

3.1 Introduction

While the predominant strategy for localizing an autonomous vehicle is through the use of three-dimensional (3D) light detection and ranging (LIDAR) scanners, in this chapter we look at trying to solve the same problem using only a monocular camera. Quite likely the greatest near-term enabler for self-driving cars is the increased use of camera systems in addition to expensive LIDAR scanners. We envision that the future of perception on autonomous vehicles will be a mix of low-cost LIDAR and radar sensors, with a complementary suite of several cameras.

Our approach leverages a graphics processing unit (GPU) so that we can generate several synthetic, pin-hole camera images, which we then *directly* compare against streaming vehicle imagery. This differs from other visual localization approaches, like Wu and Ranganathan (2013), which rely on sophisticated feature sets. This significantly simpler approach avoids over-engineering the problem by formulating a slightly more computationally expensive solution that is still real-time tractable on a mobile-grade GPU and capable of high accuracy localization.

In this chapter, we propose exploiting 3D prior maps augmented with surface reflectivities constructed with a survey vehicle equipped with 3D LIDAR scanners. We localize a vehicle by comparing imagery from a monocular camera against several candidate views, seeking to maximize normalized mutual information (NMI) (as outlined in Fig. 3.1). The key contributions of this chapter are:

- We present a multi-modal approach that allows us to use LIDAR-based ground maps, which accurately depicts the metric and surface reflectivity of the ground, for localization with a monocular camera.
- We benchmark our visual localization method with state-of-the-art LIDAR-based localization strategies.

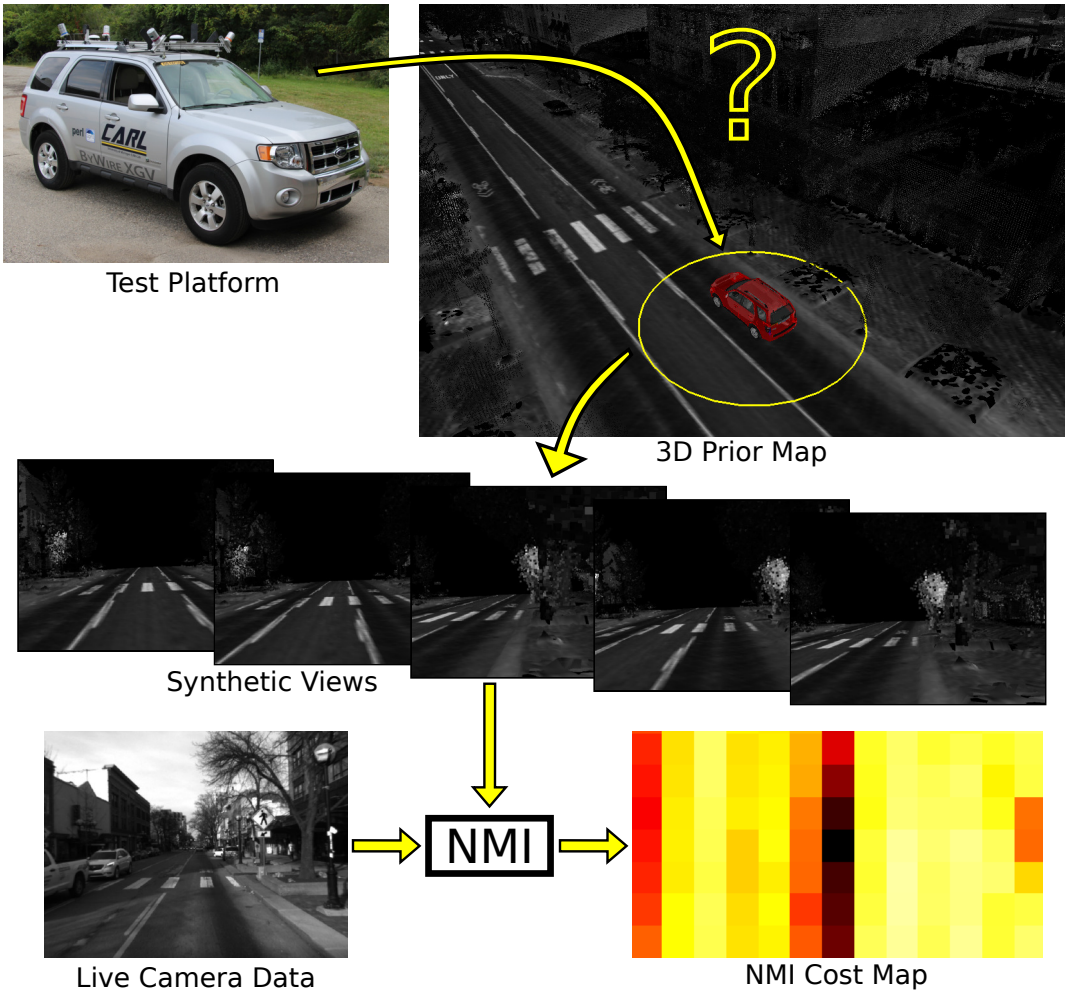


Figure 3.1: Overview of our proposed visual localization system. We seek to localize a monocular camera within a 3D prior map (augmented with surface reflectivities) constructed from 3D LIDAR scanners. Given an initial pose belief, we generate numerous synthetic views of the environment, which we then evaluate using normalized mutual information against our live view from camera imagery.

- We show a GPU implementation that can provide real-time localization at ~ 10 Hz.

3.2 Related Work

Early visual simultaneous localization and mapping (SLAM) methodologies employ filtering frameworks in either an extended Kalman filter (EKF) (Davison et al., 2007) or FastSLAM framework (Eade and Drummond, 2006), to generate a probability distribution over the belief pose and map of point features. In order to accurately localize within these point feature maps, one relies on co-observing these features. However, these features frequently vary with time of day and weather conditions, as noted by Napier and Newman (2012), and cannot be used without an intricate observability model, similar to that presented by Carlevaris-Bianco and Eustice (2012).

In the context of autonomous vehicles, Wu and Ranganathan (2013) and Ranganathan, Ilstrup, and Wu (2013) try to circumvent this by identifying and extracting higher order features from road markings in images that are far more robust and representative of static infrastructure. Their method is able to densely and compactly represent a map by using a sparse collection of features for localization. However, their method also assumes reliability of features and requires a flat ground, whereas our projective registration requires no features extraction and allows for more complex geometries.

Rather than relying on specific image *features* in our prior map (and complicated, hand-tuned feature extractors), our method is motivated by the desire to avoid point features *entirely* and do whole image registration relative to a static, 3D map captured by survey vehicles.

In work by Stewart and Newman (2012), the use of a 3D map for featureless camera-based localization that exploits the 3D structure of the environment was explored. They were able to localize a monocular camera by minimizing normalized information distance between the appearance of 3D LIDAR points projected into multiple camera views. Further, McManus et al. (2013) used a similar 3D map with reflectivity information to generate synthetic views for visual distraction suppression.

This approach has been previously considered, but methods thus far rely on the reconstruction of the local ground plane from a stereo camera pair. Senlet and Elgammal (2011) create a local top-view image from a stereo pair and use chamfer matching to align their reconstruction to publicly available satellite imagery. Similarly, Napier and Newman (2012) use mutual information to align a live camera stream to pre-mapped local orthographic images generated from the same stereo camera. With both of these methods, small errors in stereo pair matching can lead to oddly distorted orthographic reconstructions, thus confusing

the localization pipeline. Further, our multi-modal approach allows us to take advantage of LIDAR scanners to actively capture the true reflectivity of our map, meaning our prior map is not susceptible to time of day changes in lighting and shadows.

The use of mutual information for multi-modal image registration has been widely used in the medical imaging domain for several decades (Maes et al., 1997; Pluim, Maintz, and Viergever, 2003). More recently, the idea has been transferred to robotics for calibration of visual cameras to LIDAR scanners (Pandey et al., 2012; Taylor, Nieto, and Johnson, 2013). This sensor registration has mostly been considered an offline task due to the expense of generating synthetic views for calibration.

To move this into real-time localization, we propose using a GPU to generate synthetic views, which we can then use a normalized measure of mutual information to optimize over our vehicle's pose. The GPU has been frequently used in robot localization for precisely this reason, including: Kinect depth-SLAM (Fallon, Johannsson, and Leonard, 2012), image feature correspondence search for SIFT features (Charmette, Royer, and Chausse, 2010), and line features (Kitanov, Bisevac, and Petrovic, 2007).

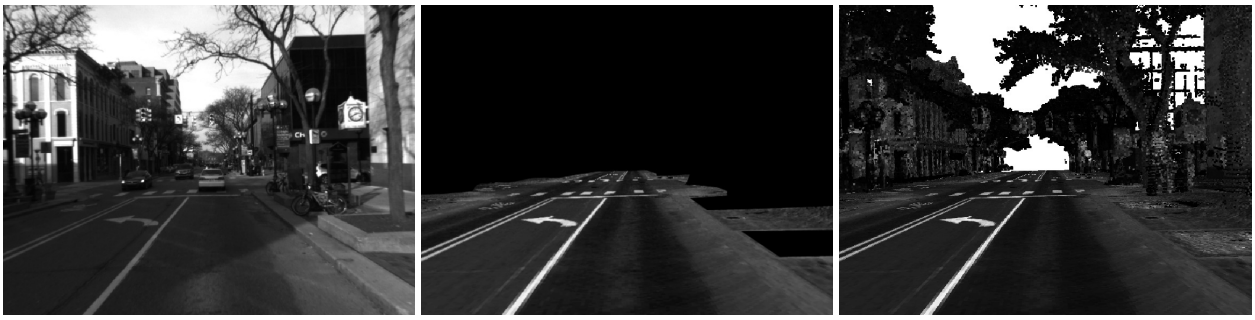
More recently, Pascoe et al. (2015) expanded from our work to consider a full 3D optimization in a method called FARLAP. Their method derives analytic gradients that allow them to use gradient-based methods to optimize their image registration cost function.

3.3 Prior Map

In order to develop a prior map suitable for localization, our survey robots make a traversal through the environment equipped with a 3D LIDAR scanner. As detailed in Appendix B, we solve an offline SLAM problem to generate a map to be used for online localization. This provides us with an optimized pose-graph, where each pose in our survey is metrically accurate to the route driven.

From the optimized pose-graph, we construct a dense ground-plane mesh. For each pose in our optimized set of poses, $X = \{\mathbf{x}_i\}_{i=0}^{M}$, we take each corresponding point cloud, $\mathcal{P}_i = \{\mathbf{p}_j\}_{j=1}^{n}$, where $\mathbf{p}_j = [x_j, y_j, z_j, r_j]^\top$ is the metric position and reflectivity of a point, and extract the local ground-plane from these using a region growing method starting at known ground-plane near the vehicle; resulting in ground-plane point clouds, $\mathcal{Q}_i \subseteq \mathcal{P}_i$. Note, this region growing method is identical to that used in Section 2.3 to extract the ground-plane for Gaussian mixture maps of the ground-plane reflectivities.

We then transform each point cloud into the global frame, $\mathcal{Q}'_i = \mathbf{x}_i \oplus \mathcal{Q}_i$, where $\oplus$ is the head-to-tail composition operation (Smith, Self, and Cheeseman, 1990). All of these point clouds are then accumulated into a global point cloud, $\mathcal{Q} = \{\mathcal{Q}'_i\}_{i=0}^{M}$. We then reduce this



(a) Camera Image (b) Ground-only Synthetic Image (c) Full 3D Synthetic Image

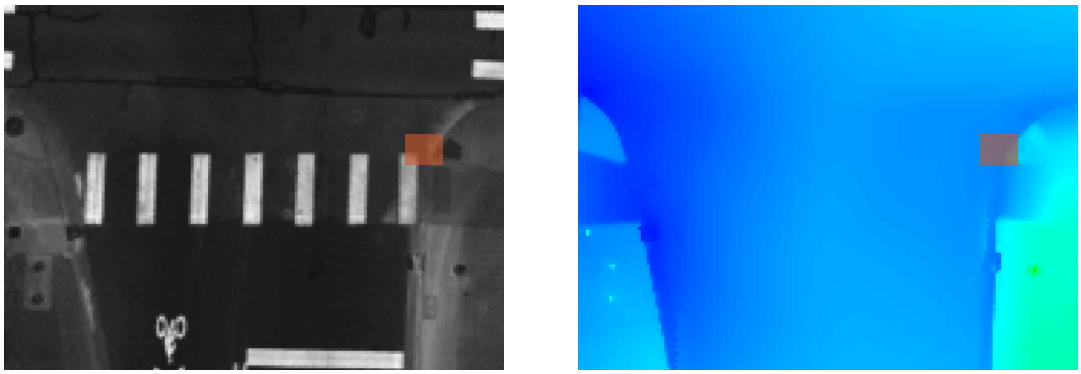
Figure 3.2: In this chapter, we are interested in registering the camera image shown in (a) by generating synthetic LIDAR images of our prior map. This can be done using (b) only the ground mesh or with (c) the ground mesh and 3D structure point cloud. We chose to use ground maps only as it significantly increased scene prediction time.

point cloud into two grid maps $\mathbf{Z}$ and $\mathbf{R}$ to model the z-height and reflectivity of the ground surface, respectively. We chose grids that are 10 cm in resolution to ensure capturing road paint features and each cell's value is taken as the median value of point cloud points that fell into the grid cell. A sample of these grid maps can be seen in Fig. 3.3(a).

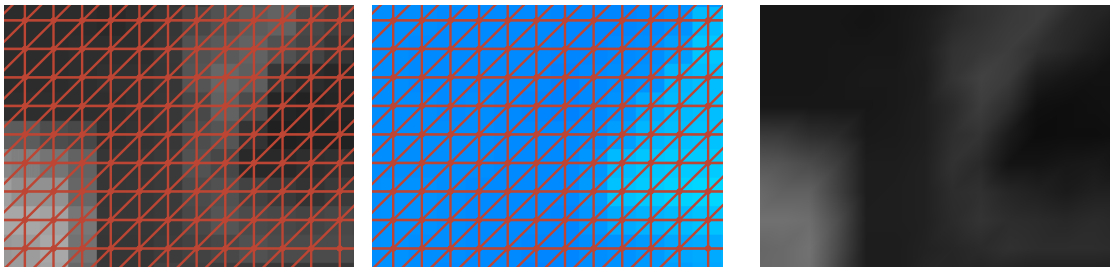
We then use these maps to generate triangle meshes *online* for localization. As our robot traverses the world, the z-height and reflectivity grids are loaded from disk. For each grid cell in these overlapping grids, we generate a vertex at the center of each cell, where z-height and reflectivity is obtained from $\mathbf{Z}$ and $\mathbf{R}$, respectively. These vertices are then connected into triangles as shown in Fig. 3.3(b), resulting in the full triangle mesh rendered in Fig. 3.3(c). This algorithm is logically equivalent to extracting the ground-plane at each point and draping an orthographic texture over a varying z-height map; see Algorithm 4 for more details.

Note that our system is not limited to ground-only maps. We applied our method to incorporate the full 3D point cloud in our prior map (e.g., buildings, street poles, trees) as depicted in Fig. 3.2, but found that the added structure did not appreciably increase registration quality enough to warrant the additional rendering cost (the 3D structure more than *doubled* scene prediction time). Furthermore, our two grid approach for modelling the ground-plane results in a compact representation when stored on disk. On the other hand, generating a scalable method to model the 3D world at a high enough fidelity that is usable for vision systems is incredibly difficult (point clouds are quite large).

However, we did find that it was extremely important to use a mesh surface as opposed to a strict planar texture because the planar texture did not accurately depict the curvature of the road (e.g., the crown of the road), as can be seen in the map colored by z-height in Fig. 3.4(c).

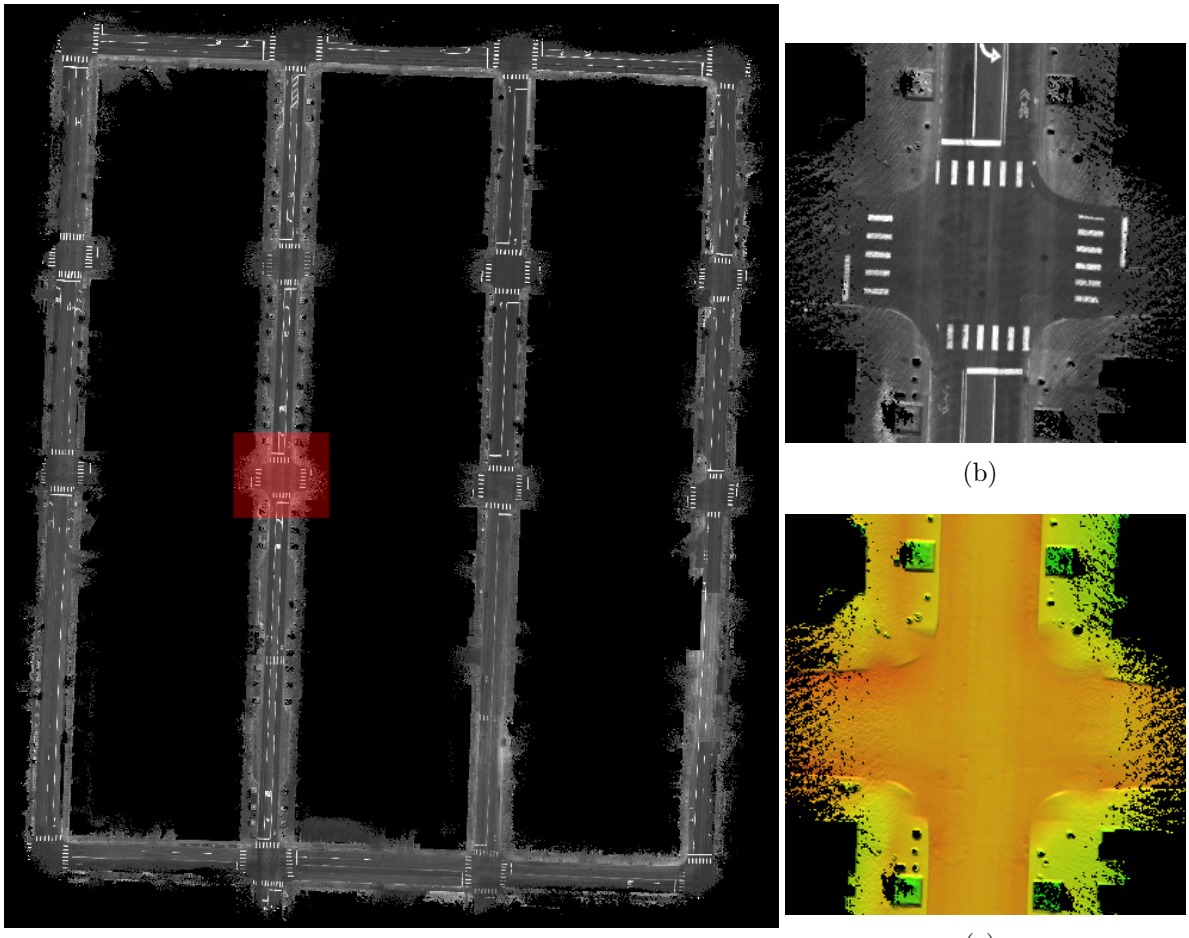


(a) Reflectivity and z-height Grids



(b) Tessellation of Reflectivity and z-height Grids (c) Resulting Mesh Texture

Figure 3.3: This figure demonstrates the tessellation performed online to generate 3D textures for localization. Offline we generate two grids during mapping: one where each pixel contains a reflectivity value and another where each pixel contains the z-height of the ground-plane; these are shown in (a). Online, these maps are tessellated as shown in (b) to generate a mesh where each vertex position is derived from the z-height grid and the appearance is derived from the reflectivity grid. The resulting textured mesh can be seen in (c).



(a) (b) (c)

Figure 3.4: Sample ground mesh used to generate synthetic views of the environment. In (a), we show a 400 m $\times$ 300 m ground mesh colored by surface reflectivity, with a zoomed in view shown in (b) (this region is highlighted in red in (a)). We show the same zoomed view, colored by z-height to demonstrate the height variation we are able to capture with our ground-mesh in (c); yellow-to-red represents $\Delta z = 30$ cm.

Algorithm 4 Pose-Graph to Ground-Mesh

Input: Optimized pose-graph, $X = \{\mathbf{x}_0, \cdots, \mathbf{x}_M\}$, and point clouds, $\mathcal{P} = \{\mathcal{P}_0, \cdots, \mathcal{P}_M\}$,
Output: Triangle ground-mesh, $T = \{t_0, \cdots, t_N\}$

// extract ground-plane point cloud
Initialize global, ground-plane point cloud, $\mathcal{Q} = \{\}$
for $\mathbf{x}_i$ in X **do**
    // extract ground-plane point cloud
    $\mathcal{Q}_i = \texttt{ExtractGround}(\mathcal{P}_i)$
    // for each point in point cloud, transform from body frame to global frame
    $\mathcal{Q}'_i = \mathbf{x}_i \oplus \mathcal{Q}_i$
    // append to global point cloud
    $\mathcal{Q} \leftarrow \texttt{Append}(\mathcal{Q}'_i)$
end for
// build grids
Initialize empty 10 cm grids, $\mathbf{Z}$ and $\mathbf{R}$ for z-height and reflectivity
for each point $\mathbf{q}_j$ in $\mathcal{Q}$ **do**
    $[x_j, y_j, z_j, r_j]^\top = \mathbf{q}_j$
    // update median in each grid cell
    $\mathbf{Z}[x_j, y_j] \leftarrow \texttt{UpdateMedian}(\mathbf{Z}[x_j, y_j], z_j)$
    $\mathbf{R}[x_j, y_j] \leftarrow \texttt{UpdateMedian}(\mathbf{R}[x_j, y_j], r_j)$
end for
// tessellate grids to form ground-plane mesh
Initialize ground mesh, $T = \{\}$
for x_k, y_k in $\{\mathbf{Z}, \mathbf{R}\}$ **do** ▷ for each pixel in these grids
    if $\texttt{MaxDiff}(\mathbf{Z}[x_k, y_k]$, $\mathbf{Z}[x_k, y_{k+1}]$, and $\mathbf{Z}[x_{k+1}, y_{k+1}]) < z_{\text{threshold}}$ **then**
        $v_0 = \{x_k, y_k, \mathbf{Z}[x_k, y_k], \mathbf{R}[x_k, y_k]\}$
        $v_1 = \{x_k, y_{k+1}, \mathbf{Z}[x_k, y_{k+1}], \mathbf{R}[x_k, y_{k+1}]\}$
        $v_2 = \{x_{k+1}, y_{k+1}, \mathbf{Z}[x_{k+1}, y_{k+1}], \mathbf{R}[x_{k+1}, y_{k+1}]\}$
        $T \leftarrow \texttt{AddTriangle}(v_0, v_1, v_2)$
    end if
    if $\texttt{MaxDiff}(\mathbf{Z}[x_k, y_k]$, $\mathbf{Z}[x_{k+1}, y_{k+1}]$, and $\mathbf{Z}[x_{k+1}, y_k]) < z_{\text{threshold}}$ **then**
        $v_0 = \{x_k, y_k, \mathbf{Z}[x_k, y_k], \mathbf{R}[x_k, y_k]\}$
        $v_1 = \{x_{k+1}, y_{k+1}, \mathbf{Z}[x_{k+1}, y_{k+1}], \mathbf{R}[x_{k+1}, y_{k+1}]\}$
        $v_2 = \{x_{k+1}, y_k, \mathbf{Z}[x_{k+1}, y_k], \mathbf{R}[x_{k+1}, y_k]\}$
        $T \leftarrow \texttt{AddTriangle}(v_0, v_1, v_2)$
    end if
end for

3.4 Projective Image Registration

Given an initial pose prior $T_0 = [x_0, y_0, z_0, r_0, p_0, h_0]^\top$, the goal of our image registration problem is to find the relative two-dimensional (2D) offset $\Delta T = [\Delta x, \Delta y, 0, 0, 0, \Delta h]^\top$ that optimally aligns the projected map, $\mathcal{M}$, against our camera image, I. Our approach can be formulated as,

$$\Delta T^* = \underset{\Delta T}{\operatorname{argmax}} \operatorname{NMI}(I, L), \tag{3.1}$$

where $L = \texttt{proj}(\mathcal{M}, T_0 \oplus \Delta T)$ is the synthetic LIDAR image generated by projecting the map into our hypothesized camera location at $T_0 \oplus \Delta T$ and $\oplus$ is the head-to-tail composition. This optimization is framed as a local search problem within the vicinity of T_0 and could be done in an exhaustive manner by generating a predicted view for the entire $dom(x) \times dom(y) \times dom(h)$ search volume to avoid local maxima of hill-climbing searches. The remainder of this section details our method for efficiently generating these predicted views (L), how we can improve on the exhaustive search approach, and our NMI evaluation metric.

3.4.1 Generating Predicted Views

Given a query camera pose parameterized as $[\mathrm{R}|\mathbf{t}] = f(T_0 \oplus \Delta T)$, where R and $\mathbf{t}$ are the camera's rotation and translation, respectively, our goal is to provide a synthetic view of our world from that vantage point. We use OpenGL, which is commonly used for visualization utilities, in a robotics context to simulate a pin-hole camera model, similar to Fallon, Johannsson, and Leonard (2012).

All of our ground-plane mesh triangles are drawn in a world frame using indexed vertex buffer objects. These triangles are incrementally passed to the GPU as necessary as the robot traverses the environment—though the maps in our test set can easily fit within GPU memory. We pass the projection matrix,

$$P = \mathrm{M} \cdot \mathrm{K} \cdot \begin{bmatrix} \mathrm{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}, \tag{3.2}$$

to our OpenGL Shading Language (GLSL) vertex shader for transforming world vertex coordinates to frame coordinates. Here,

$$\mathrm{M} = \begin{bmatrix} \frac{2}{w} & 0 & 0 & -1 \\ 0 & -\frac{2}{h} & 0 & 1 \\ 0 & 0 & -\frac{2}{z_f - z_n} & -\frac{z_f + z_n}{z_f - z_n} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

and

$$\mathrm{K} = \begin{bmatrix} f_x & \alpha & -c_x & 0 \\ 0 & f_y & -c_y & 0 \\ 0 & 0 & z_n + z_f & z_n \times z_f \\ 0 & 0 & -1 & 0 \end{bmatrix}, \tag{3.4}$$

where w and h are the image's width and height, z_n and z_f are the near and far clipping planes, and the elements of K correspond to the standard pinhole camera model. Note that the negative values in K's third column are the result of inverting the z-axis to ensure proper OpenGL clipping.

For efficient handling of these generated textures, we render to an offscreen framebuffer that we then directly transfer into a CUDA buffer for processing using the CUDA-OpenGL Interoperability. Sample synthetic views can be seen in Fig. 3.5. In our system, we further use frustum culling to limit the number of triangles drawn; this allows our pipeline to easily generate over $1,000$ synthetic frames per second.

3.4.2 Simplified Rotational Search

A naïve approach to this local search problem would be to use the OpenGL pipeline to generate a synthetic view for each discrete step within the search volume, $dom(x) \times dom(y) \times dom(h)$. However, this would result in generating $n_x \times n_y \times n_h$ synthetic views. Because the predicted view rasterization is the primary bottleneck of the system (taking nearly 1 ms for each render), here we propose a method of warping the camera measurement to explore the heading space (warpings can be performed at 0.1 ms instead and can be parallelized with the serial OpenGL rasterizations).

For each homogeneous point in our source camera image, $\underline{\mathbf{u}}^i$, we can warp the point by a given rotation R and translation $\mathbf{t}$ via,

$$\underline{\mathbf{u}}^{i'} = \mathrm{KRK}^{-1}\underline{\mathbf{u}}^i + \mathrm{K}\mathbf{t}/z^i, \tag{3.5}$$

where K is the camera calibration matrix and z^i is the scene depth at pixel i. Considering only rotations, this point transfer can be done without regard to scene depth and is therefore no longer a function of what the camera is imaging. This allows us to precompute a bank of rotational mappings that we can then apply to each source image,

$$\underline{\mathbf{u}}^{i'} = \mathrm{KRK}^{-1}\underline{\mathbf{u}}^i. \tag{3.6}$$

This technique allows us to use the OpenGL pipeline to generate only $n_x \times n_y$ synthetic

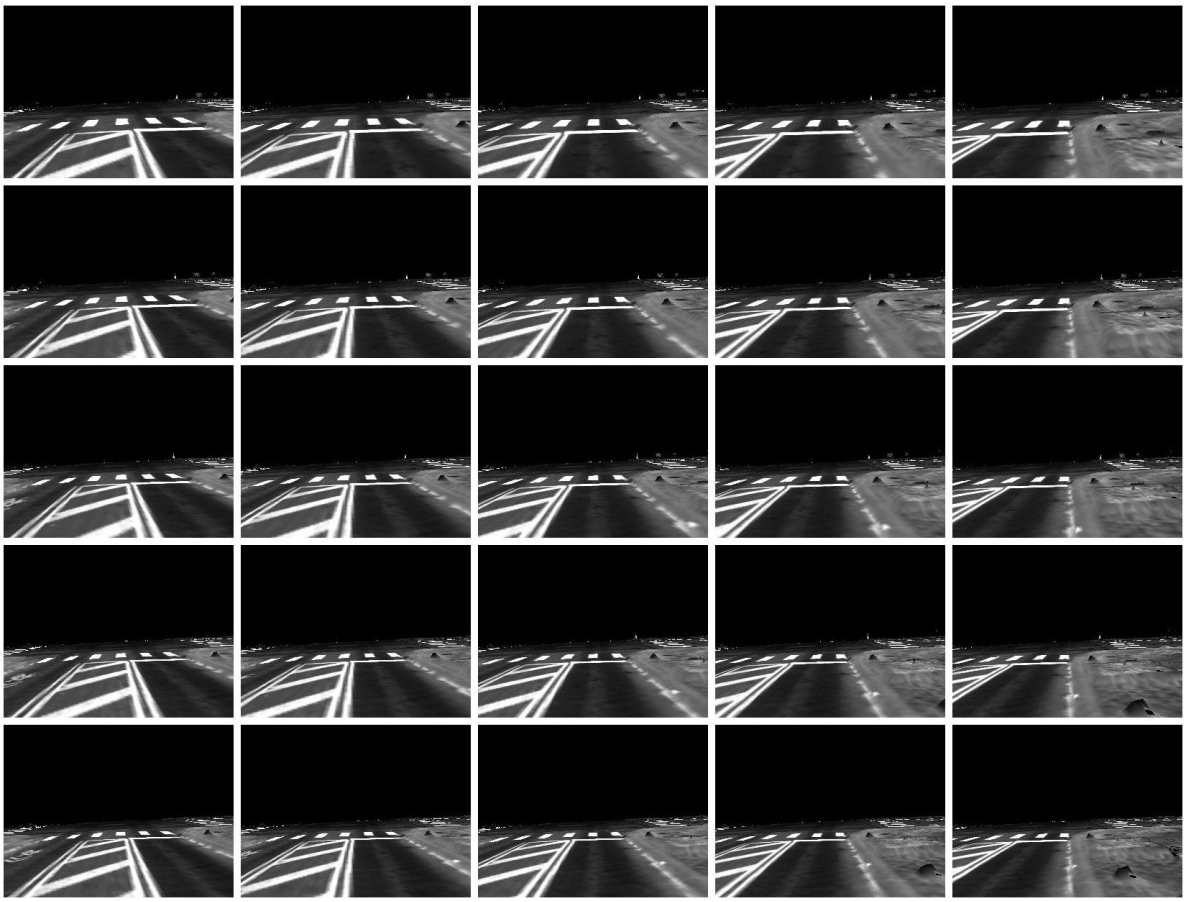


Figure 3.5: Sample synthetic views generated by our OpenGL pipeline. These views were generated by varying longitudinal and lateral translation around the optimally aligned image (*center*).

views, *first*, then compare each against n_h (warped) measurements. We still evaluate the same number of candidate pairs, though we significantly reduce our OpenGL pipeline overhead. A sample of these rotations can be seen in Fig. 3.6. The combination of our methods allow us to evaluate over $10,000$ candidate registrations per second.

3.4.3 Normalized Mutual Information Image Registration

Mutual information has been successfully used in various fields for registering data from multi-modal sources. Mutual information provides a way to statistically measure the mutual dependence between two random variables, A and B. Most commonly, mutual information is defined in terms of the marginal and joint entropies of each:

$$MI(A, B) = H(A) + H(B) - H(A, B), \tag{3.7}$$

Figure 3.6: Sample warping applied to images to reduce the overall search space for image registration; pictured here are warps of $\Delta h = \{-6.0°, -4.5°, -3.0°, -1.5°, 0.0°, 1.5°, 3.0°, 4.5°, 6.0°\}$ By rotating each *source* image in place, we can optimally pull out as much information from a single OpenGL rendered image.

where these entropies can be realized by evaluating the Shannon entropy over the random variables A and B:

$$H(A) = -\sum_{a \in A} p(a) \log p(a), \tag{3.8}$$

$$H(B) = -\sum_{b \in B} p(b) \log p(b), \tag{3.9}$$

$$H(A, B) = -\sum_{a \in A} \sum_{b \in B} p(a, b) \log p(a, b). \tag{3.10}$$

This mutual information formulation clearly demonstrates that maximization of mutual information is achieved through the minimization of the joint entropy of A and B. This

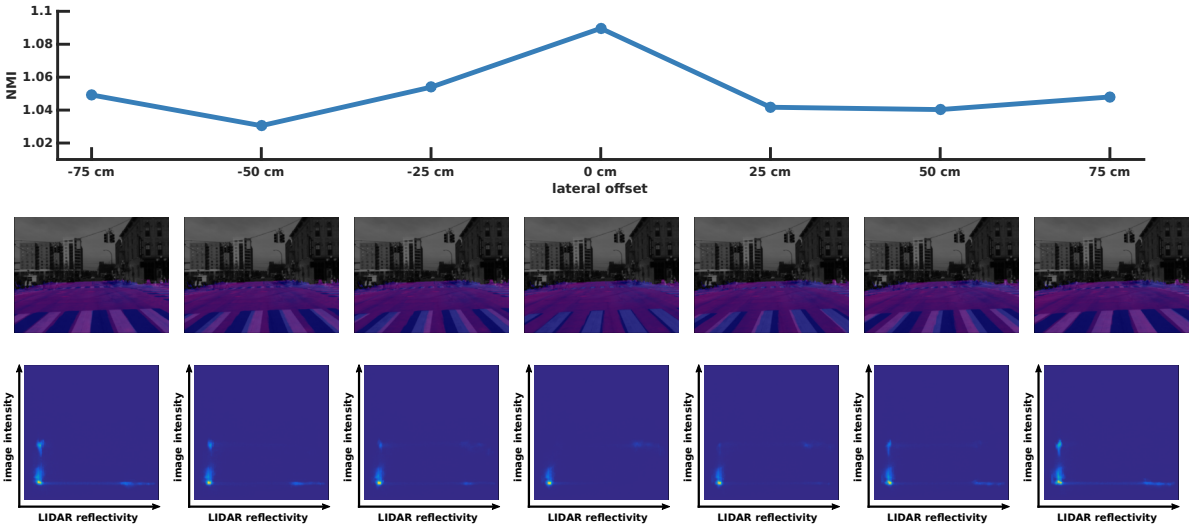


Figure 3.7: This figure demonstrates a sample registration to provide intuition into the mutual information (MI) measurement. Consider an image in which we generate a synthetic view at fixed lateral offsets to ground-truth. The top row shows the corresponding normalized mutual information at each offset. The middle row shows a blending of the image and the synthetic LIDAR image (colored *purple-blue*). To compute the mutual information, a joint histogram table is computed for each offset and MI is maximized when these joint histograms are minimally dispersed; signifying underlying signal agreement. Note that MI is actually *higher* at ± 75 cm than ± 50 cm because MI is concerned with finding the best overlap even if that overlap yields negative correlation (the road markers are perfectly out of phase here)—the partial overlap at ± 50 cm yields more dispersion in the joint histogram table relative to ± 75 cm where two modes are distinctly made.

optimality coincides with minimizing the dispersion in the joint histogram for the two random variables. A sample demonstrating this with sample joint histogram tables can be seen in Fig. 3.7.

By viewing the problem in this information theoretic way, we are able to capture more interdependency between random variables than with simple similarity or correlation-based measures. For example, tar strips in the road frequently appear dark in LIDAR reflectivity, yet bright in visual imagery. Correlative methods can only measure either a negative or positive correlation and often fails under varying illumination. However, because maximization of mutual information is concerned with seeking tightly compact joint distributions, we can successfully capture this mutual dependence (see Fig. 3.10(b)). Note that it would be quite difficult to create a hand-tuned feature detector that could identify this type of information for localization.

Because our source imagery and predicted views have varying amount of overlap (largely due to our warping technique and our synthetic views not being fully dense images), we

instead employ a normalized mutual information measure. The amount of overlap between two candidate images can bias the standard mutual information measure toward *lower* overlap image pairs. To avoid these effects, Studholme, Hill, and Hawkes (1999) proposed an overlap invariant measure of mutual information, normalized mutual information (NMI):

$$NMI(A,B) = \frac{H(A) + H(B)}{H(A,B)}. \tag{3.11}$$

This measure shares the same desirable qualities of the typical mutual information shown in (3.7), but is more robust to overlap changes.

In summary, our image registration results in the following optimization:

$$\Delta T^* = [\Delta x^*, \Delta y^*, \Delta h^*]^\top = \underset{\Delta x, \Delta y, \Delta h}{\operatorname{argmax}} \operatorname{NMI}(W, L), \tag{3.12}$$

where $W = \texttt{warp}(I, \Delta h)$ is the warping of the source imagery, I, that spans the rotational search space, and $L = \texttt{proj}(\mathcal{M}, T_0 \oplus [\Delta x, \Delta y]^\top)$ refers to the synthetic views that are generated over the translational search space.

3.5 Results

We evaluated our theory through data collected on our TORC ByWire XGV autonomous platform, as seen in Fig. 3.1. This automated vehicle is equipped with four Velodyne HDL-32E 3D LIDAR scanners, a single Point Grey Flea3 monocular camera, and an Applanix POS-LV 420 inertial navigation system (INS).

Algorithms were implemented using OpenCV (Bradski and Kaehler, 2008), OpenGL, and CUDA and all experiments were run on a laptop equipped with a Core i7-3820QM central processing unit (CPU) and mid-range mobile GPU (NVIDIA Quadro K2000M).

As in Section 2.6, we made two passes through the same environment (on separate days) and aligned the two together using our offline SLAM procedure outlined in Appendix B.1. This allowed us to build a prior map ground-mesh on the first pass through the environment. Then, the subsequent pass would be well localized with respect to the ground-mesh, providing sufficiently accurate ground-truth in the experiment (accuracy an order of magnitude greater than our localization errors). Experiments are presented on two primary datasets:

- *Downtown*: 3.0 km trajectory through downtown Ann Arbor, Michigan in which multiple roads are traversed from both directions and the dataset contains several dynamic obstacles. Note, this is referred to as M-D1, M-D2, and D-1 in the previous chapter.

- *Stadium*: 1.5 km trajectory around Michigan Stadium in Ann Arbor, Michigan. This dataset presents a complicated environment for localization as half of the dataset is through a parking lot with infrequent lane markings.

3.5.1 Image Registration

We first present the results of our raw image registration to detach the registration quality from a filtered estimate. To evaluate our image registration alone, we took our ground truth pose belief over the *Downtown* dataset and tried to perform an image registration to our map once a second. Ideally, we should be able to perfectly register our prior map, however, due to noise or insufficient visual variety in the environment, we end up with a distribution of lateral and longitudinal errors.

We present these results in two ways. First, we show our vehicle's trajectory through the prior map in which we color our longitudinal and lateral errors at each ground-truth pose, shown in Fig. 3.9. In this figure, larger and brighter markers indicate a larger error in registration at that point. One can immediately notice that we are not perfectly aligned longitudinally on long, straight stretches; during these stretches, the system frequently relies on a double, solid lane marking to localize off of. To maintain accuracy, the system requires occasional cross-streets, which provide more signal for constraining our pose belief.

Second, we show the same results in histogram form, as can be seen in Fig. 3.8, where we see that our registration is primarily concentrated within ± 30 cm of our ground-truth. A common mode can be found in the tails of the histograms. This is caused by areas that are *visually* feature poor or obstructed by significant obstacles; for example, lane markings can often be perceived by the survey vehicle's LIDAR scanners and captured in our prior map, yet the subtle transition between pavement and faded lane markings cannot be observed by our camera. In these scenarios, the optimal normalized mutual information will try to pull the registration toward the edges of our prior map—the edges are often feature poor as well, and this alignment minimizes the joint entropy of the two signals.

Finally, we present several scenarios of our image registration succeeding (Fig. 3.10) and common causes of failure (Fig. 3.11). These figures were generated by exploring within a local window around known ground truth.

In Fig. 3.10(b), we see the benefit of using a mutual information derived cost function over computing image correlation. Mutual information is able to exploit the statistical dependence between the two signals regardless of some linear relationship; this is demonstrated by finding positive correlation dependence in most of the image and negative correlation in matching bright tar strips to their dark appearance from the LIDAR. Further, because our prior map

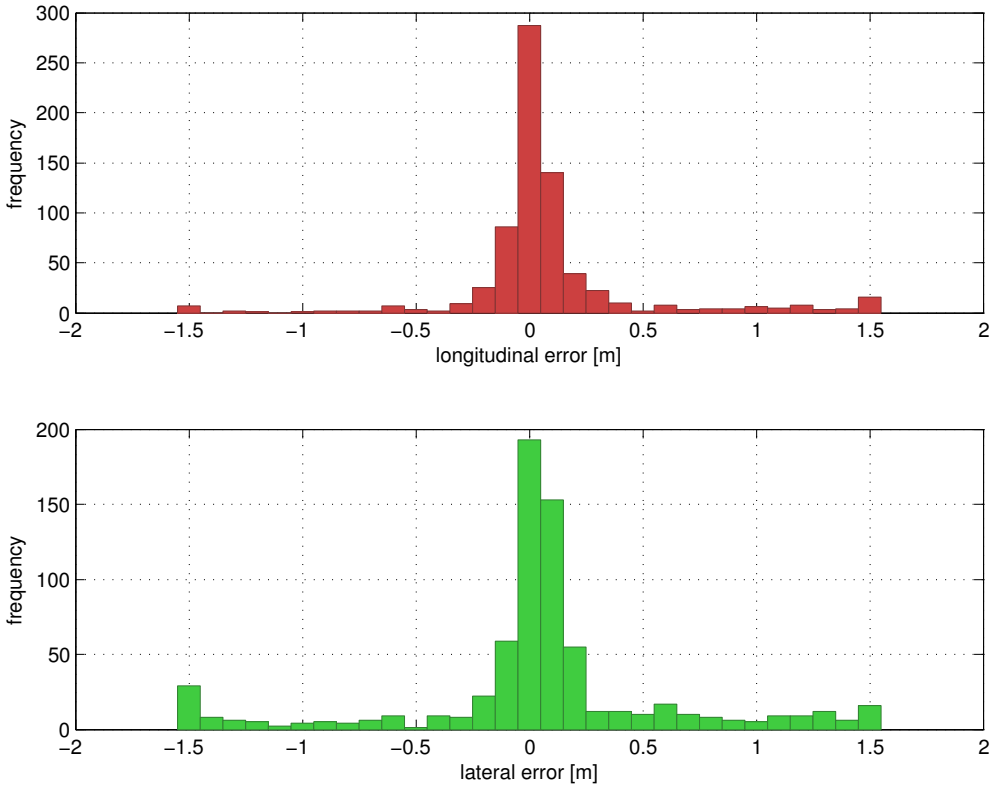
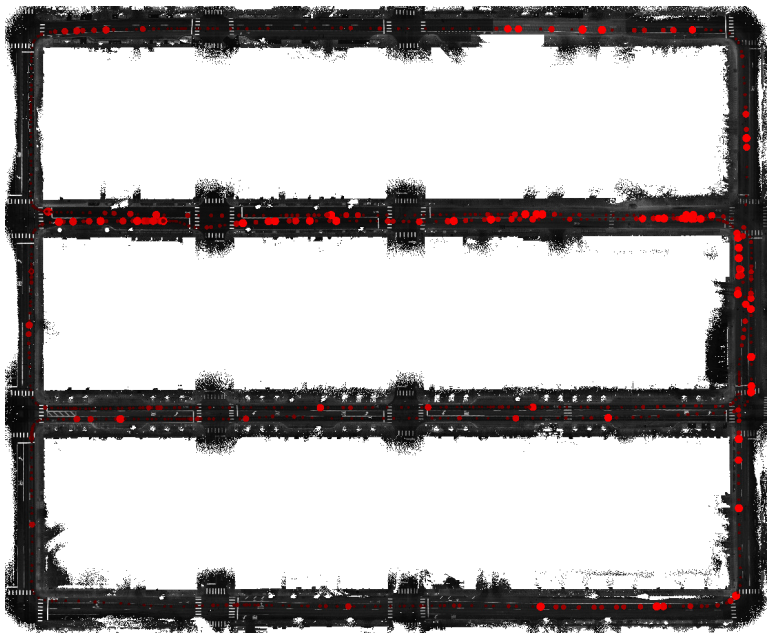


Figure 3.8: Histograms of longitudinal and lateral error that our projective image registration produces (i.e., we are not using our localization filter to generate this). Most frequently, our proposed method is able to stay within ±30 cm of ground truth. Large frequency in the tails of the histograms are caused by feature poor regions where the NMI biases towards the edges of the search limits (which was ±1.5 m here).

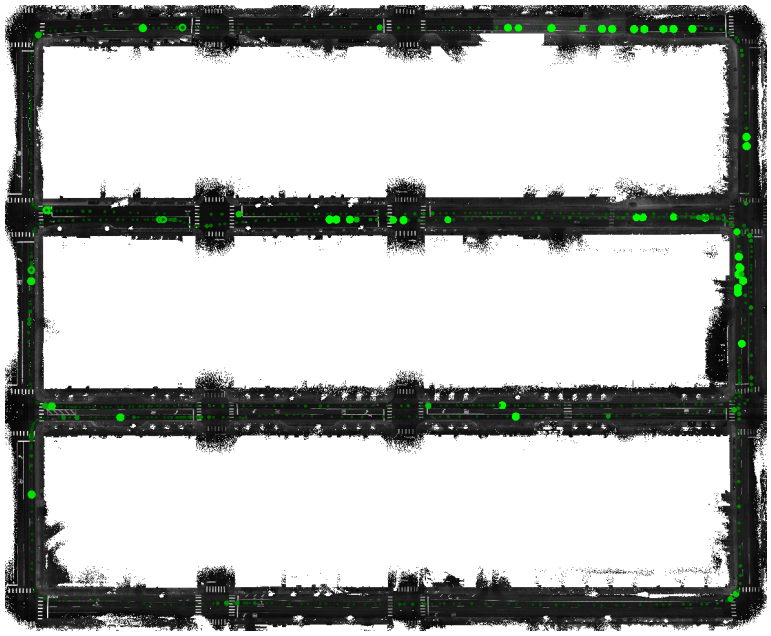
is not corrupted by shadows, our localization can withstand shadow in our images as shown in Fig. 3.10(c). This is because multiple modes will build up in our joint histogram table for a given LIDAR value—one for the illuminated parts of the image and one for the shadowed regions.

While we do see some resilience to obstacles in Fig. 3.10(a), this is because a majority of the image still exhibits a statistical dependency to the LIDAR image. As the image becomes more overwhelmed with obstacles, as in Fig. 3.11(a), the joint likelihoods become erroneously sampled and our registrations fail.

Additionally, our method can only be effective when the underlying map has enough visual variation. Our proposed approach fails in Fig. 3.11(b) because the underlying map is quite feature poor and in Fig. 3.11(c) where we can only be constrained laterally as there is limited longitudinal discrepancy.

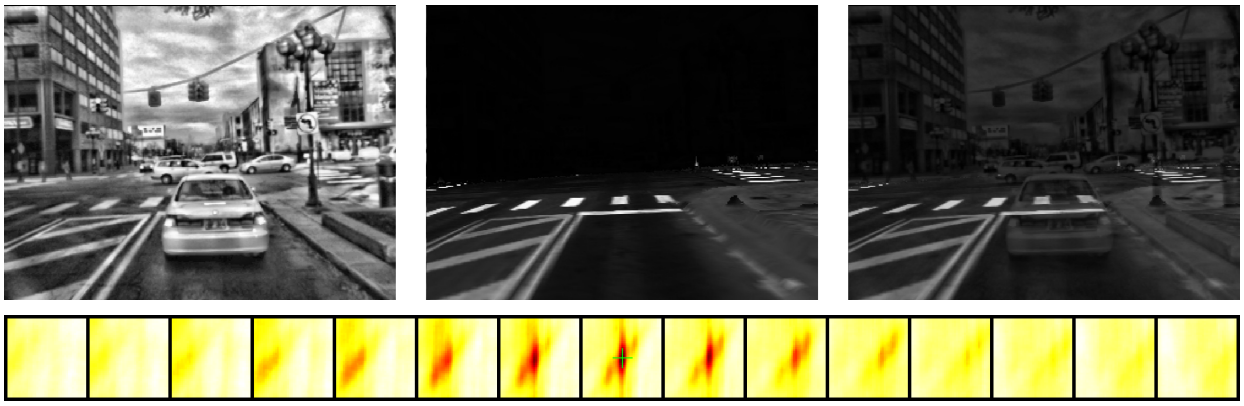


(a) Longitudinal Errors

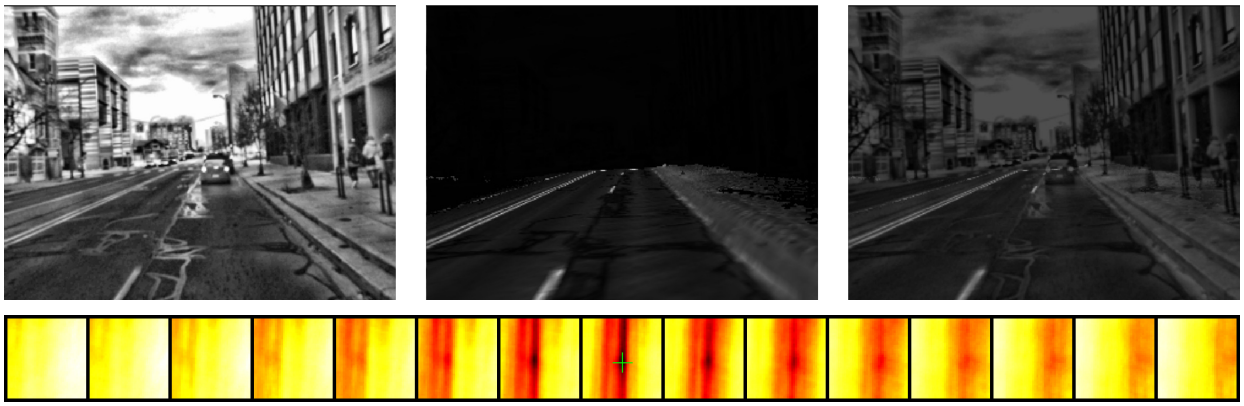


(b) Lateral Errors

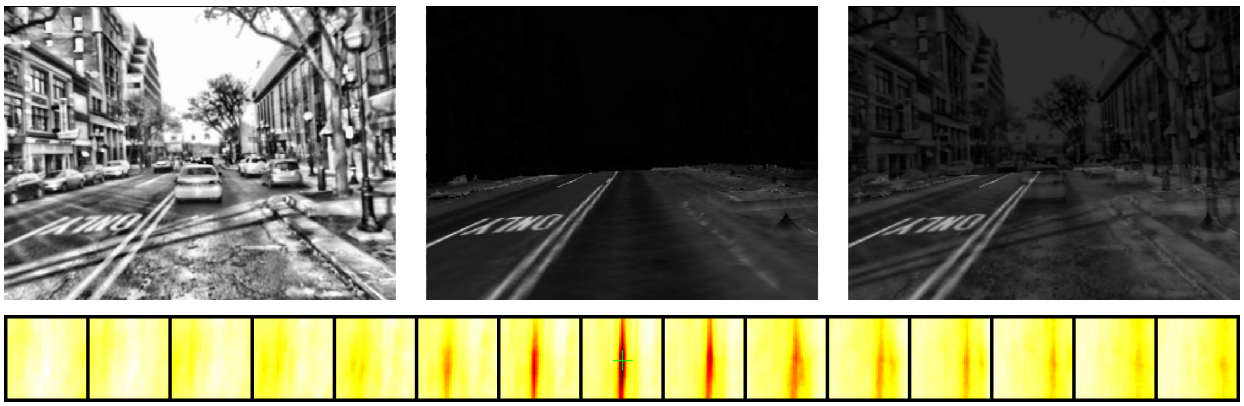
Figure 3.9: Longitudinal and lateral errors in our image registration, sampled at each second of our trajectory; larger and brighter markers indicate regions where image registration produced higher errors longitudinally (a) or laterally (b). In (a), we see that, especially on the third street from the bottom, we are only well constrained longitudinally in and around intersections; quite often, we are only constrained laterally due to a double, solid lane divider being the only feature in view. In (b), we see that our method provides good lateral registration—the few bright spots seen are when the vehicle is stopped at an intersection with a vehicle obstructing our view. Note that in these two figures, perfect longitudinal or lateral registration is indicated by *dark* red or green, respectively.



(a) Typical observation, unaffected by dynamic obstacle

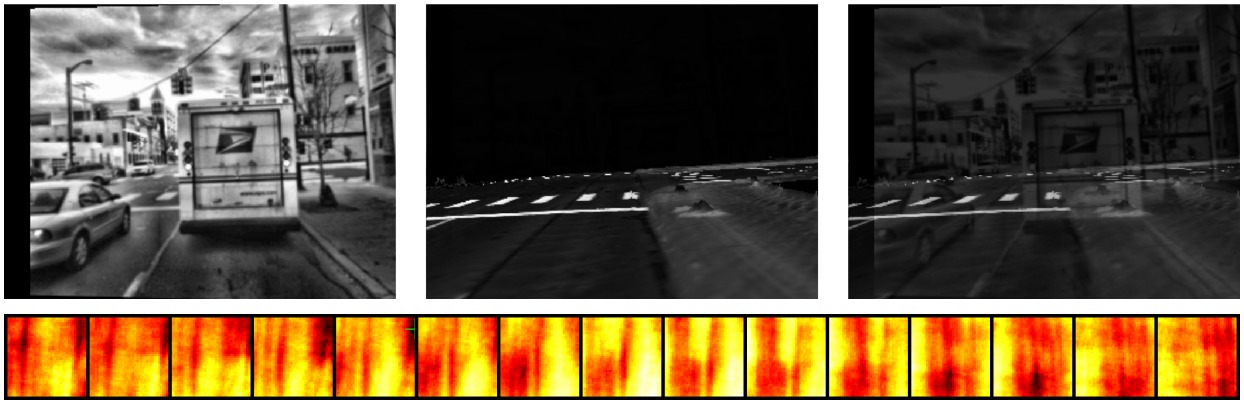


(b) Positive and negative correlation captured by cost function (bright tar strips in imagery aligns with dark prior map)

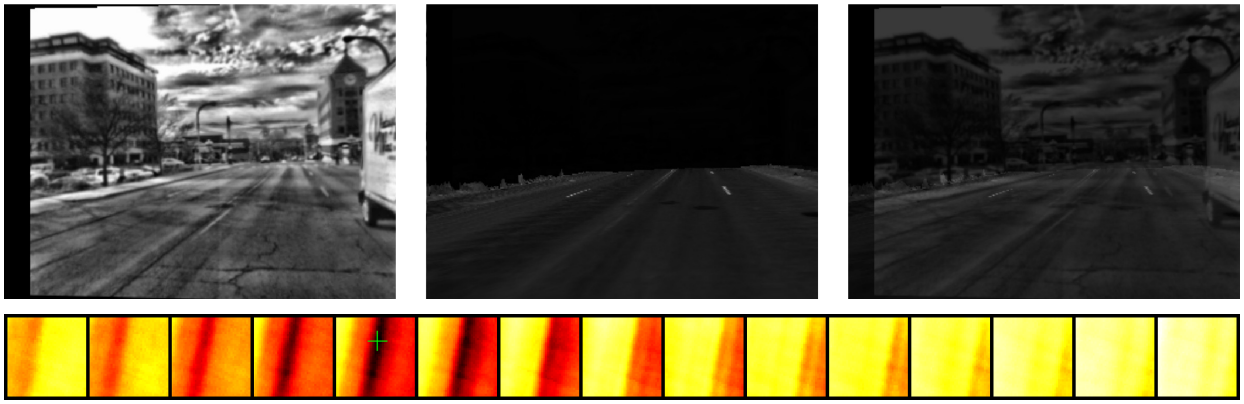


(c) Our method is robust to uniform shadowing

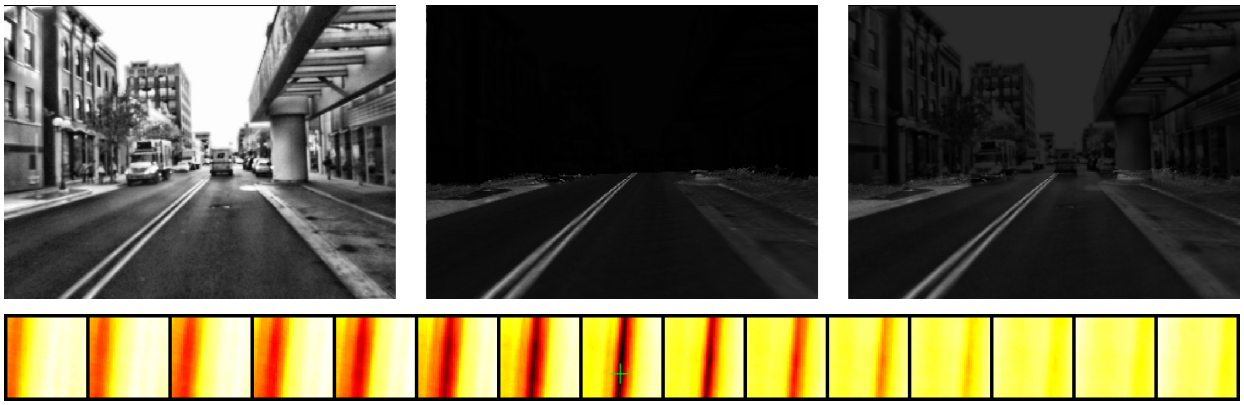
Figure 3.10: Successful image registrations. From left-to-right, we show three images: the source image, the best predicted synthetic LIDAR image, and an alpha-blending of the source and synthetic image. The normalized mutual information cost map is also shown, where each tile in the cost map represents a different *heading* slice of the 3D cost surface, each pixel then represents an xy translation, and the maximum found is marked with a green '+'. Note, the cost surface should be maximized at the center pixel of the center tile.



(a) Our method is not robust to all dynamic obstacles



(b) Poor imagery and low feature content in prior map



(c) Only constrained laterally by double lane marker

Figure 3.11: Failure modes of our image registration. From left-to-right, we show three images: the source image, the best predicted synthetic LIDAR image, and an alpha-blending of the source and synthetic image. The normalized mutual information cost map is also shown, where each tile in the cost map represents a different *heading* slice of the 3D cost surface, each pixel then represents an xy translation, and the maximum found is marked with a green '+'. Note, the cost surface should be maximized at the center pixel of the center tile.

3.5.2 Filtered Localization

We next looked at the filtered response of our system that incorporates the projective image registration into the EKF localization framework described in Section 2.5, where we include measurements *only* from the inertial sensors, a wheel encoder, and our monocular camera. Image registration uncertainty, R_k, is estimated by fitting a covariance to the explored cost surface, as is done in Olson (2009).

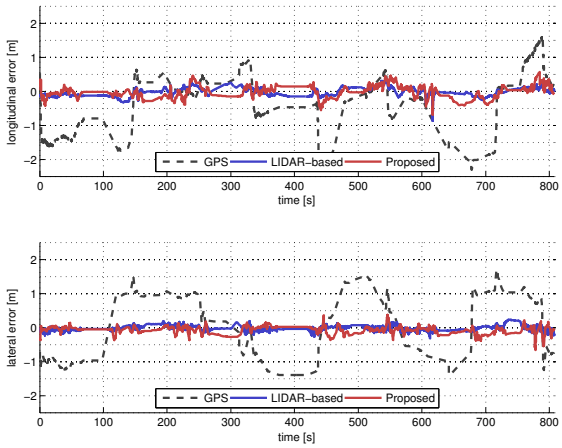
Moreover, we compare our localization performance against our own implementation of the state-of-the-art LIDAR-based localization proposed by Levinson and Thrun (2010). Our LIDAR-based system builds orthographic ground images using the four Velodyne HDL-32E's onboard; these orthographic ground images can then be aligned to an orthographic prior map built using an accumulation of these scans.

We present longitudinal and lateral errors over time for global positioning system (GPS), LIDAR-based localization, and our proposed single camera algorithm within the *Downtown* and *Stadium* datasets (see Fig. 3.12). These results are summarized in Table 3.1. Here we show that we are able to achieve longitudinal and lateral root mean squared (RMS) errors of 19.1 cm and 14.3 cm, respectively, on the *Downtown* dataset. Further, we obtain longitudinal and lateral RMS errors of 45.4 cm and 20.5 cm, respectively, on the *Stadium* dataset. Our proposed solution is able to maintain error levels at a similar order of magnitude as the LIDAR-based options, while using a sensor that is several orders of magnitude cheaper.

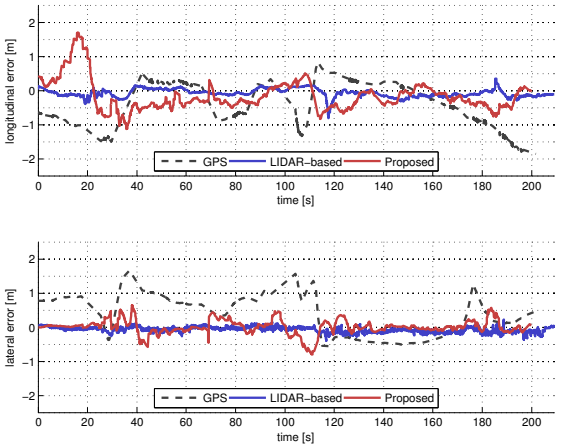
Note that the *Stadium* results show a rather large variance in longitudinal error; this is because half of the dataset is through a parking lot containing little visual variation. Also, we are slow to initially converge longitudinally because the first 20 s of the run is on a two-lane road containing only a double, solid lane marker.

	***Downtown* RMS Error**		***Stadium* RMS Error**	
Method	Longitudinal	Lateral	Longitudinal	Lateral
GPS	91.0 cm	100.5 cm	81.7 cm	73.4 cm
LIDAR-based	12.4 cm	8.0 cm	14.3 cm	10.9 cm
Proposed	19.1 cm	14.3 cm	45.4 cm	20.5 cm

Table 3.1: Comparison of RMS errors for GPS, LIDAR-based localization, and our proposed vision-only localization. Our method is able to remain sufficiently well localized for use in an automated vehicle.



(a) *Downtown*—Filtered Results.



(b) *Stadium*—Filtered Results.

Figure 3.12: Here we present our localization accuracy in terms of longitudinal and lateral error relative to SLAM-optimized ground-truth over time. Our proposed solution achieves a similar order of magnitude performance as the state-of-the-art LIDAR-based solutions while being several orders of magnitude cheaper. GPS alone is presented to show that it cannot provide reliable localization for automated vehicles. Despite significant longitudinal errors in the *Stadium* dataset, we are still able to maintain lateral alignment, which is critically important for lane-keep.

3.6 Conclusion

In this chapter, we showed that a single monocular camera can be used as an information source for visual localization in a 3D LIDAR map containing surface reflectivities. By maximizing normalized mutual information, we are able to register a camera stream to our prior map. Our system is aided by a GPU implementation, leveraging OpenGL to generate synthetic views of the environment; this implementation is able to provide corrective positional updates at ~ 10 Hz. Moreover, we compared our algorithm against the state-of-the-art LIDAR-only automated vehicle localization, revealing that our approach can achieve a similar order of magnitude error rate, with a sensor that is several orders of magnitude cheaper.

CHAPTER 4

Probabilistic Obstacle Partitioning for Improved Localization

4.1 Introduction

Localization is a key task for autonomous cars; systems such as the Google driverless car rely on precise and detailed maps for safe operation (Urmson, 2015). Light detection and ranging (LIDAR) sensors are capable of providing rich information—including metric range and point appearance. Robust methods can use this data for vehicle localization, for example by extracting the ground-plane for alignment to a prior map, as done by Levinson et al. (Levinson and Thrun, 2010; Levinson, Montemerlo, and Thrun, 2007).

Due to decreased cost and the ability to have robust, redundant sensing, vision sensors as part of the localization pipeline can be a great enabler for autonomous platforms. Contrary to LIDAR approaches, identifying the ground-plane from a camera image is a much more challenging task. In Chapter 3, we considered localizing with just a monocular camera by aligning the whole image to a prior map. This can be problematic as the ground-plane can frequently be obscured by obstacles within view of the camera. As indicated in Fig. 3.11(b), our visual localization system can easily be distracted when the image is dominated by obstacles, leading to a degradation in localization.

In this chapter, we are interested in partitioning an image stream into obstacles and prior map as shown in Fig. 4.1, with the goal of only using the portions of the image containing the prior map for localization. This addition will lend itself to a more robust end-to-end visual localization system.

We propose to leverage our textured prior map, consisting of a ground-plane mesh, to formulate a Markov random field (MRF) that models the image partition between the obstacles and the ground-plane. We present several probabilistically motivated energy functions that can be fused in this MRF framework. Specifically, the prior map allows us to evaluate ground likelihood by conditioning our belief on the expected appearance from the prior map. This

(a) (b)

Figure 4.1: In this chapter, we propose to extract optical flow vectors and probabilistically evaluate them against expected flow vectors. We present an MRF framework that fuses various energy potentials, in addition to optical flow likelihood, such that minimizing the energy results in an optimal partition of the image space into ground-map and obstacles, as denoted by the green line through (b).

distribution is captured by the joint histograms used when computing mutual information. Moreover, we present a probabilistic method to evaluate optical flow likelihood against our three-dimensional (3D) prior map, taking into account expected motion parallax.

Our proposed approach is evaluated on a challenging urban dataset where lighting is non-uniform and our camera is an 8-bit monochrome sensor (note, explicitly no color used to demonstrate effectiveness of our approach). We demonstrate our proposed algorithms by looking at errors with respect to hand-labeled groundtruth. Additionally, we look at bringing this obstacle partition into the visual localization pipeline and present results that demonstrate improved registration when obstacle masks are used.

4.1.1 Related Work

Our problem of road segmentation can be cast as a more general scene segmentation problem, of which there has been a vast amount of research. Semantic labels can typically be learned using extracted features (Sturgess et al., 2009), features that are learned as well (Alvarez et al., 2012), or using some coarse prior knowledge of the environment (Irie and Tomono, 2013). Felzenszwalb et al. (2010) present a deformable parts based model that uses a trained latent SVM over HOG-like features to detect various object categories. These methods were later extended by Held, Levinson, and Thrun (2012), realizing that vehicles are constrained to the ground plane, formulating a scale and context weighting.

In our work, we are less interested in developing a classifier that relies on feature extraction and offline-learning; instead we are interested in using the full image with minimal feature extraction (nothing more than gradients) for road segmentation. Our motivation is to augment any potential gaps in learning or mis-tuned feature extractors to improve robustness in our approach.

Modeling the ground plane appearance distribution directly from image data has been successful in many domains. Ulrich and Nourbakhsh (2000) build a histogram appearance model for the ground plane, learning this distribution with the assumption that the lower window of the image is mostly ground plane. Dahlkamp et al. (2006) improves on this by restricting appearance learning to within co-registered laser range finder returns. In addition, they and Álvarez and Ĺopez (2011) use an RGB colorspace transform to minimize the effect of shadows by actively removing them from their appearance model. These works heavily rely on color images that are clearly more discriminative than grayscale images.

Others have looked to exploit camera motion to infer scene structure and motion. Considering a temporal stream of images, Zhang et al. (2006) looked at the residual error from focus of expansion estimation. Similar to our proposed work, others have assumed a locally planar ground in which motion can be inferred (Lourakis and Orphanoudakis, 1998) or provided via odometry (Braillon et al., 2006). Moreover, Wedel et al. (2007) proposed classifying between foreground and background by warping sequential images onto multiple plane hypotheses.

In this work, we are instead interested in computing dense optical flow fields as it lends itself to a probabilistic formulation that can capture uncertainties in camera motion and can more easily be used to detect dynamic obstacles. Most early work using optical flow for obstacle detection had an intended use with a stationary camera for surveillance tasks. These works typically focus on segmenting the dynamic parts of a scene aside from an otherwise static image. Haag and Nagel (1999) look at image edge elements, where the optical flow is more accurate, to guide their model-based tracking. Work by Rosales and Sclaroff (1999) tracked objects in 3D using an extended Kalman filter (EKF), while also using a background subtraction mechanism. Many of these methods can not translate to our domain as the key assumption of a stationary camera is violated.

The use of optical flow for obstacle detection of dynamic obstacles from a dynamic vehicle was first looked at in the joint works by Enkelmann et al. (1994) and Krüger, Enkelmann, and Rössle (1995). In these works, optical flow vectors are *sparsely* extracted and compared against estimated model flow vectors. The latter of these two works extends the former with robust filtering and statistical point classification into one of three sets—ground plane, static obstacle, and dynamic obstacle—noting the probabilistic model of each. Roberts and Dellaert (2013) performed a similar classification employing dense flow fields, though found problems

when faced with textureless image regions. Our work intends to extend theirs because we acknowledge the benefits of dense flow fields in obstacle estimation, though we introduce a penalty to discredit optical flow measured in texturesless areas.

Similar to our work, McManus et al. (2013) applied optical flow for background detection on an autonomous vehicle assuming an already known localization within a 3D prior map. They evaluate the likelihood of this optical flow by computing optical flow twice—first on the raw images then on the image warped via the 3D prior map—and comparing the flow vectors.

Badino, Franke, and Pfeiffer (2009) proposed a novel idea called the "Stixel World" in which image processing demands can be significantly reduced under the context of on-vehicle cameras. The representation is such that the world can be decomposed into a set of vertical *stixels* that directly correspond to a column in image space. A key insight here is that the pixels between the bottom of the image and the first obstacle in each column is strictly identified as free-space—thus imposing a 1D image space partitioning that can be efficiently solved using dynamic programming.

Our work is quite similar in underlying machinery to more recent work by Yao et al. (2015) and Levi, Garnett, and Fetaya (2015), both closely resemble the stixel-world formulation, while using a monocular camera. In Yao et al. (2015), they propose inference in a 1D MRF that incorporates various cues including pixel appearance, image edges, temporal consistency, and spatial smoothness. However, many of these cues are severely biased towards the bottom of the images, leading to a brittle system when faced with difficult imagery. In Levi, Garnett, and Fetaya (2015), they use a convolutional neural network (CNN) learn the appearance of the image partition offline. We propose a set of cues that are probabilistically motivated, allowing joint reasoning over appearance and inferred motion.

4.2 Preliminaries

In our work, we use a survey vehicle equipped with 3D LIDAR scanners to construct a detailed prior map for localization. As detailed in Section 3.3, we build a 3D mesh of the ground-plane that we texturize using reflectivity measurements from the LIDAR, as shown in Fig. 4.2.

We then localize an image, I_t, taken at time t from a monocular camera within this prior map, $\mathcal{M}$, by exploiting the statistical dependency between camera intensity values and LIDAR reflectivities. Using a coarse prior (such as that from GPS), we generate several

(a) Prior Map (b) Synthetic Image (c) Image with Expected Depth

Figure 4.2: Using a survey vehicle equipped with 3D LIDAR scanners, we can offline generate a rich mesh of the ground-plane colored by LIDAR reflectivity, as shown in (a). OpenGL is used to generate synthetic viewpoints and expected depth of this prior map, (b) and (c). The synthetic image and depth are relied upon for obstacle partitioning.

synthetic views of the prior map, maximizing normalized mutual information (NMI):

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}}{\operatorname{argmax}} \, \mathrm{NMI}\left(I_t, L_t\right), \tag{4.1}$$

where $L_t = \texttt{proj}(\mathcal{M}, \mathbf{x})$ is the synthetic LIDAR image generated by projecting $\mathcal{M}$ into the camera frame at $\mathbf{x} = [x, y, z, r, p, h]^\top$, using the standard pinhole camera model. NMI is a normalized variant of mutual information that is maximized by minimizing the dispersion between two random variables (a metric that is evaluated with the entropy of the joint and marginal histograms of the two signals).

The projections for localization can be done efficiently within OpenGL using custom shaders. Further, the OpenGL rendering process populates a depth buffer to determine screen ordering of drawn triangles. This depth buffer can be scaled by the *near* and far clipping planes to generate an expected depth image $\hat{Z}_t$. Thus, the localization process provides expected depths for a given camera location, which we will leverage for obstacle partitioning.

In the following sections, we will detail how we can estimate prior map likelihood and then how we can incorporate these likelihoods into a Markov random field (MRF) smoothing framework. A goal of this chapter is to generate image partitions so that NMI can be computed over those pixels believed to be imaging the prior map.

4.3 Probabilistic Obstacle Partitioning

Our proposed formulation is heavily motivated from the Stixel World presented by Badino, Franke, and Pfeiffer (2009) and a similar monocular approach for free space estimation (Yao

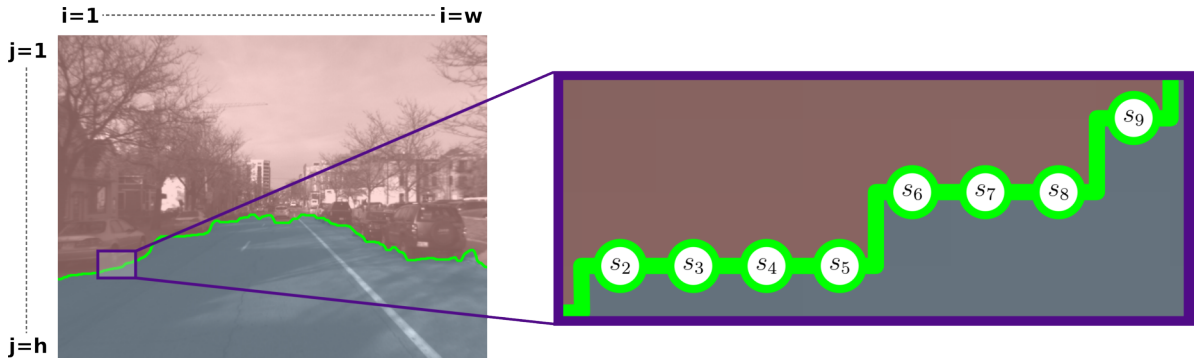


Figure 4.3: In probabilistic obstacle partitioning, we propose to use a 1D MRF to partition an image into two sets: ground-plane (*blue*) and not ground-plane (obstacles, *red*). Each variable in our MRF (*green*) models a partition point for each column in the image, and has a unary potential associated with it that is computed using a set of partition likelihoods. These variables are connected with their pairwise neighbors to enforce smoothness, resulting in an MRF that can be solved efficiently with dynamic programming.

et al., 2015). Realizing the structure of the roadway as viewed in a camera image, we assume that there is a distinct separation between free space and obstacles. This defined partition regularizes the task of identifying obstacles in a camera image. We propose to use a sequence of camera images to derive probabilistic appearance and inferred motion likelihoods to find this partition.

Given an image I_t taken at time t, probabilistic obstacle partitioning seeks an optimal seam that traverses the image left-to-right, $S = \{s_i\}_{i=1}^{w}$, where s_i can take the value of $h+1$ labels, $s_i \in \{0, \cdots, h\}$ (w and h denote the width and height of I_t). Considering the i^{th} column of I_t, $\mathbf{c_i} = \{I_t(i,j)\}_{j=1}^{h}$, the cut s_i implies a partitioning of this column into two disjoint sets such that $\{I_t(i,j)\}_{j=1}^{s_i}$ is sampled from the obstacle set, $\mathcal{O}$, and $\{I_t(i,j)\}_{j=s_i+1}^{h}$ is sampled from the prior map, $\mathcal{M}$. In our framework, $i=1$ indicates the leftmost column and $j=1$ indicates the topmost row of the image. An illustrative example of this setup is provided in Fig. 4.3.

We formulate obstacle partitioning as the maximum *a posteriori* (MAP) estimation of the set of column seams conditioned on the previous n camera images,

$$S^* = \underset{S}{\operatorname{argmax}}\, p(S|I_t, \cdots, I_{t-n+1}). \tag{4.2}$$

Assuming a Markov factorization, we can factor the posterior as

$$\begin{aligned} p(S|I_t, \cdots, I_{t-n}) &\propto p(I_t, \cdots, I_{t-n}|S)p(S) \\ &= \prod_i p\left(I_t, \cdots, I_{t-n}|s_i\right) \prod_j p\left(s_j|s_{j-1}\right), \end{aligned} \tag{4.3}$$

where we assume independence between columns $\mathbf{c_i}$. Applying the negative log-likelihood, the MAP inference results in the following energy function to be minimized:

$$E = \sum_i \sum_{k \in K} \underbrace{w_k \phi_k\left(s_i\right)}_{\text{unary}} + \sum_j \underbrace{w_p \phi_p(s_j, s_{j-1})}_{\text{pairwise}}, \tag{4.4}$$

where K represents the set of unary potentials, $\{a, f, e, l, r\}$, and w_n represents the weighting for each potential, which can be learned using training data. The MRF forms a chain connecting neighboring columns and can be efficiently solved using dynamic programming as a Viterbi problem (Viterbi, 1967).

The pairwise potential is modeled as a truncated quadratic to enforce smoothness across the seam,

$$\phi_p\left(s_j, s_{j-1}\right) = \min(|s_j - s_{j-1}|, T_p)^2, \tag{4.5}$$

where T_p is a threshold that allows the potential to enforce local smoothness without excessively penalizing large jumps, as should be allowed with objects near the camera. The remainder of this section details the unary potentials that exploit appearance and motion in the images.

4.3.1 Unary Potentials

4.3.1.1 Appearance Potential

We derive an appearance based potential that can be learned online using a monochrome camera. The theory could easily be applied to color imagery, though we opted against to demonstrate the effectiveness of our motion potential presented next (color can be an extremely discriminative feature in this context).

The motivation for this potential is to maximize the likelihood of the class assignments (obstacle and prior map) using image intensities. The potential is defined as

$$\phi_a\left(s_i\right) = -\log p\left(\mathbf{c_i}|s_i\right), \tag{4.6}$$

where $\mathbf{c_i}$ is the set of pixels in the i^{th} column and the likelihood term is derived assuming

independence and recalling the strict partitioning of the data at s_i:

$$p\left(\mathbf{c_i}|s_i\right) = \prod_{j=1}^{h} p(I_t(i,j)|s_i) \tag{4.7}$$

$$= \prod_{j=1}^{s_i} \underbrace{p(I_t(i,j)|\mathcal{O})}_{\substack{\text{obstacle} \\ \text{likelihood}}} \prod_{j=s_i+1}^{h} \underbrace{p(I_t(i,j)|\mathcal{M})}_{\substack{\text{prior map} \\ \text{likelihood}}}. \tag{4.8}$$

The obstacle appearance model is computed using a joint histogram to maintain,

$$p(I_t(i,j)|\mathcal{O}) = p(I_t(i,j)|i,\mathcal{O}). \tag{4.9}$$

This can be thought of as a 2D histogram with image intensity on one axis and image column on the other. We convolve this with a Gaussian kernel so as to avoid over-fitting and smooth the likelihoods. See Fig. 4.4(h) for a sample of this conditional distribution.

The prior map appearance model is slightly more intricate in that each intensity is conditioned on the reflectivity of the projected prior map; therefore,

$$p(I_t(i,j)|\mathcal{M}) = p(I_t(i,j)|L_t(i,j)). \tag{4.10}$$

This conditional distribution is managed via the joint histogram over image intensity and LIDAR reflectivity—this is the same distribution used to compute NMI for localization, Fig. 4.4(f). See Fig. 4.4(h) for a sample of this conditional distribution.

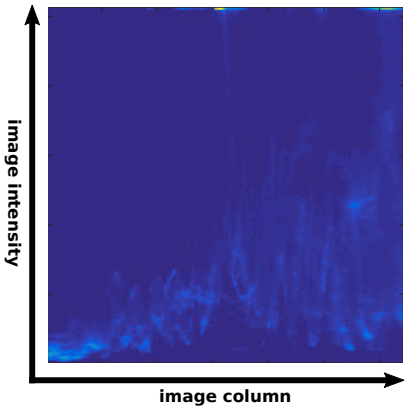
Both of these conditional histograms are learned online using the previous n pairs of images and extracted seams. Combined with the other potentials and the smoothing pairwise potential, the appearance prior continuously learns the obstacle and prior map distributions. In this work, we used a sliding time window over the last several seconds of data—this should be kept short so distributions can adapt to lighting changes.

4.3.1.2 Optical Flow Potential

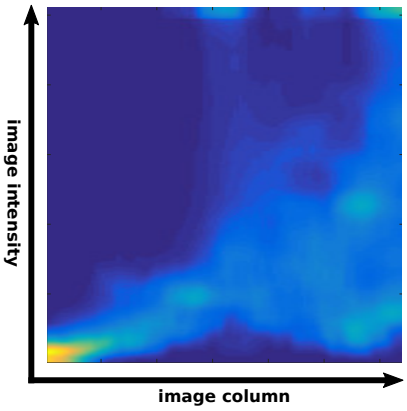
Appearance potentials alone can perform quite poorly in complex environments where partial illumination can distract the measure; moreover, 8-bit grayscale imagery makes it difficult to differentiate between cars and roadways. In this section, we present a motion potential derived from evaluating the likelihood of optical flow vectors—with the expectation that this can invalidate distracted areas due to parallax and physically moving objects. This illumination robust measure can further aid the appearance potential by maintaining the extracted partition through complex lighting transitions so that the appearance likelihoods

(a) Image

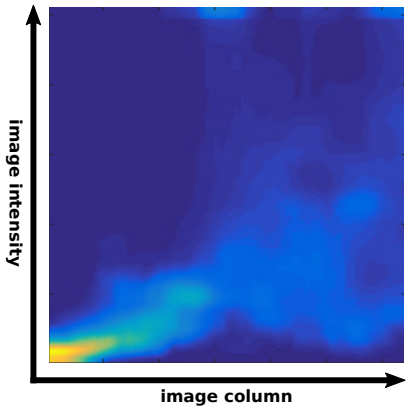
(b) Synthetic LIDAR View



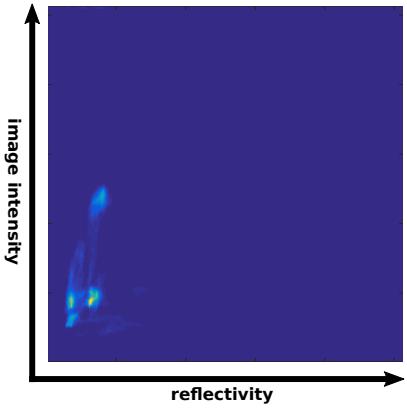
(c) Joint $p(I, i|\mathcal{O})$



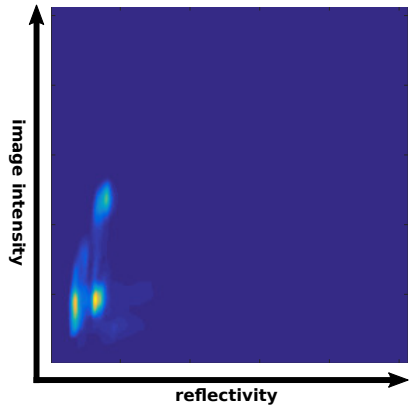
(d) Blurred (c)



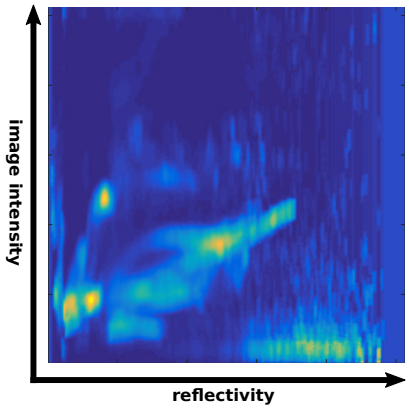
(e) Conditional $p(I|i, \mathcal{O})$



(f) Joint $p(I, L)$



(g) Blurred (f)



(h) Conditional $p(I|L)$

Figure 4.4: This figure demonstrates the histograms used for computing appearance-based likelihoods for the image and synthetic LIDAR view shown in (a) and (b), respectively. Figures (c)-(e) demonstrate likelihoods used for obstacle appearance conditioned on image column, while (f)-(h) show likelihoods used for ground appearance conditioned on the prior map reflectivity. Left-to-right, we start with a joint realization derived over the previous temporal time window, which we then blur to generalize the distributions. Finally, we convert the joint histograms into conditional histograms by normalizing by histogram column in (e) and (h). Note that (f) is identical to the joint histogram used for localization, (3.10).

can adapt to new lighting distributions.

Optical Flow Likelihood: We first extract optical flow vectors $\mathcal{U}_t = \{\mathbf{u}_1, \dots \mathbf{u}_w\}$, where $\mathbf{u}_i$ denotes a column of optical flow vectors, $\mathbf{u}_i = \{\mathbf{f}_{i,1}, \dots, \mathbf{f}_{i,h}\}$ and $\mathbf{f}_{i,j} = [u_{i,j}, v_{i,j}]^\top$ is the optical flow at pixel (i, j). We use known egomotion $\mathbf{x}_e = [x, y, z, r, p, h]^\top$ derived from vehicle odometry, an estimate on the motion uncertainty Σ_e (as detailed in Appendix A), and the expected scene depth, $\hat{Z}_t$, as outlined in Section 4.2, to calculate the expected optical flow measurement using the homogeneous point transfer (Hartley and Zisserman, 2004):

$$\mathbf{v}_{t-1} = \mathrm{KRK}^{-1}\mathbf{v}_t + \mathrm{K}\mathbf{t}/\hat{Z}_t(i, j), \tag{4.11}$$

where $\mathbf{v}_{t-1} = [x, y, 1]^\top$ represents the expected homogeneous pixel location in I_{t-1} of $\mathbf{v}_t = [i, j, 1]^\top$ (a homogeneous pixel in the current image I_t). Further, K represents the pinhole camera calibration matrix and $[\mathrm{R}|\mathbf{t}]$ is the camera motion derived from $\mathbf{x}_e$. Therefore, the expected optical flow measurement is

$$\hat{\mathbf{f}}_{i,j} = \mathbf{v}_{t-1} - \mathbf{v}_t. \tag{4.12}$$

Additionally, we can use the unscented transform (UT) to propagate motion uncertainty, Σ_e, and scene depth uncertainty at each pixel, σ_z^2, through the nonlinear point transfer of (4.11), yielding Σ_{UT}. This process of predicting optical flow is quite similar to pose-constrained correspondence search (PCCS) (Eustice, Pizarro, and Singh, 2004)) and is visually detailed in Fig. 4.5.

The uncertainty estimate, Σ_{UT}, only accounts for optical flow uncertainty induced by errors in odometry or expected scene depth. We extend this by estimating the uncertainty of measuring optical flow at each pixel considering the spatial image gradients and uncertainties in the spatio-temporal gradients, yielding Σ_g—we adopted the method proposed by Simoncelli, Adelson, and Heeger (1991). This allows us to make use out of poorly constrained flow vectors (such as those on image edges), yet still fully account for its inaccuracies. We can finally characterize the expected optical flow as a normally distributed measurement of the form:

$$\mathbf{f}_{i,j} \sim \mathcal{N}\left(\mathbf{v}_{t-1} - \mathbf{v}_t, \Sigma_{\mathrm{UT}} + \Sigma_g\right). \tag{4.13}$$

See Fig. 4.6 for visual depictions of this distribution.

Optical Flow Partition: Similar to the appearance potential, the optical flow partition potential is formulated as a function of the likelihood of class assignments (obstacle and prior

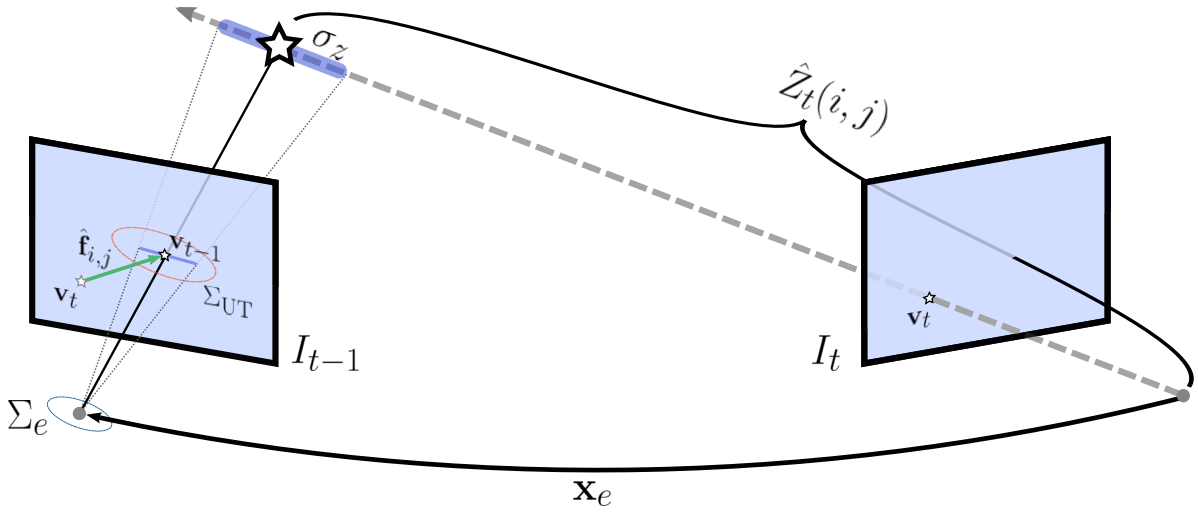


Figure 4.5: This figure demonstrates our method for generating expected optical flow measurements. Using known egomotion, $\mathbf{x}_e$, and scene depth, $\hat{Z}_t(i,j)$, we can reproject each pixel in our current image into the previous image. Further, we use the unscented transform to transform egomotion uncertainty, Σ_e, and scene depth uncertainty, σ_z, to derive optical flow uncertainty, Σ_{UT}.

map), such that minimization of the potential maximizes the associated likelihood:

$$\phi_f\left(s_i\right) = -\log p_f\left(\mathbf{u}_i|s_i\right). \tag{4.14}$$

Following a similar derivation as (4.8), we arrive at the likelihood decomposition,

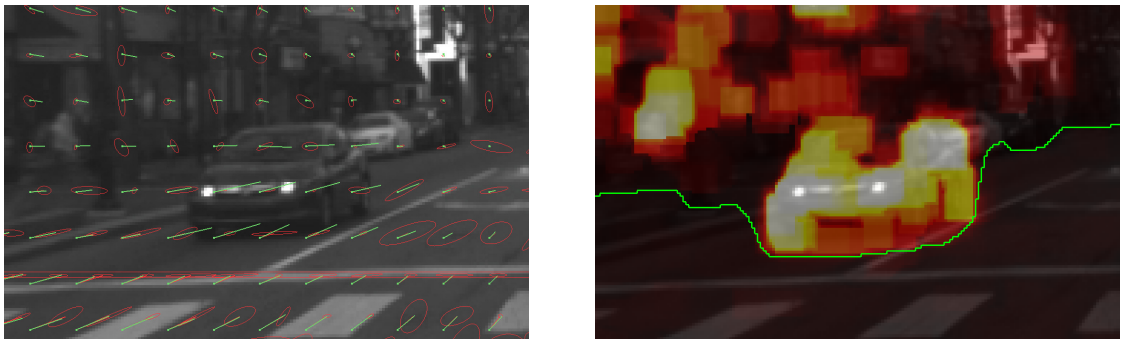
$$p\left(\mathbf{u_i}|s_i\right) = \prod_{j=1}^{s_i} p(\mathbf{f}_{i,j}|\neg\mathcal{M}) \prod_{j=s_i+1}^{h} p(\mathbf{f}_{i,j}|\mathcal{M}). \tag{4.15}$$

The prior map likelihood, $p(\mathbf{f}_{i,j}|\mathcal{M})$, is computed by evaluating against the Gaussian in (4.13). However, the term on the left we decompose even further into,

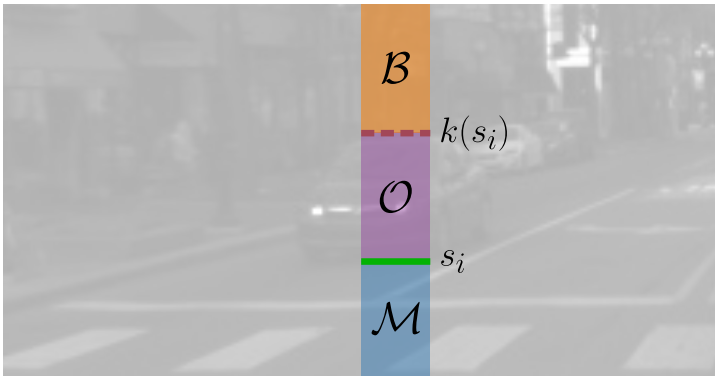
$$\prod_{j=1}^{s_i} p(\mathbf{f}_{i,j}|\neg\mathcal{M}) = \prod_{j=1}^{k(s_i)} p(\mathbf{f}_{i,j}|\mathcal{B}) \prod_{j=k(s_i)}^{s_i} p(\mathbf{f}_{i,j}|\mathcal{O}), \tag{4.16}$$

to partition the non-map elements into a background set, $\mathcal{B}$, and an obstacle set, $\mathcal{O}$, at $k(s_i)$. This split at $k(s_i)$ is necessary to divide the very dissimilar flow sets generated by $\mathcal{B}$ and $\mathcal{O}$; these 3 disjoint sets are visualized in Fig. 4.6(c).

Given a nominal world-frame height in meters of target obstacles, H_{obs}, we use known



(a) (b)



(c)

Figure 4.6: Overview of optical flow likelihood formulation. In (a), we show the optical flow vectors (*green*), where the closed tail is the location in the current image, and the expected optical flow vectors and uncertainties (*red*). Note the shape of the uncertainty ellipses following the gradients of the image. We then show the likelihood evaluation overlaid the camera imagery in (b), along with the output seam. Here, brighter indicates a lower likelihood and black/transparent indicates a higher likelihood. These likelihoods are then used in our optical flow partition likelihood; in (c) we show the partitioning into 3 disjoint sets including the map, $\mathcal{M}$, obstacles, $\mathcal{O}$, and the background, $\mathcal{M}$.

camera geometry and scene depth to derive the height in pixels of the obstacle,

$$h(s_i) = f \cdot H_{\text{obs}} / \hat{Z}_t(i, s_i), \tag{4.17}$$

where f is the camera focal length. This can then be used to determine the pixel location for splitting $\mathcal{B}$ and $\mathcal{O}$,

$$k(s_i) = s_i - h(s_i). \tag{4.18}$$

The nominal obstacle height is a tuning parameter, though we have found the algorithm to

be insensitive to selection of H_{obs} and is chosen based on minimum acceptable obstacle height ($H_{\text{obs}} = 1.5$ in our experiments).

Within an image column, elements of an obstacle are at a constant depth, resulting in flow vectors that are quite similar over the column—with noticeable deviation between flow vectors belonging to $\mathcal{O}$ and $\mathcal{B}$. Therefore, we estimate $p(\mathbf{f}_{i,j}|\mathcal{B})$ and $p(\mathbf{f}_{i,j}|\mathcal{O})$ by fitting a uniform distribution over the flow vectors within their respective column segment,

$$p(\mathbf{f}_{i,j}|\mathcal{B}) = \mathcal{U}_{\mathbf{f}_{i,1:k(s_i)}}(\mathbf{f}_{i,j}), \tag{4.19}$$

$$p(\mathbf{f}_{i,j}|\mathcal{O}) = \mathcal{U}_{\mathbf{f}_{i,k(s_i):s_i}}(\mathbf{f}_{i,j}). \tag{4.20}$$

Given that we have no further information to condition on, fitting a uniform distribution provides the maximum likelihood estimate of the corresponding flow vectors within each segment.

We compute this potential over multiple image sequences so that we can capture fast moving objects, yet still maintain observability for slow moving objects (such as those within the focus of expansion). We chose to use Farneback's optical flow algorithm (Farnebäck, 2003) and perform forward-backward flow to discard inconsistent measurements (these discarded measurements provide no influence in the likelihood computations). It is important to note that while stationary, the optical flow potential only provides input to the MRF if something else is moving (a roughly uniform prior over all partitionings otherwise). This is a byproduct of the formulation as stationary flow vectors observed from a stationary platform yields near constant likelihoods derived in (4.15).

4.3.1.3 Additional Potentials

In this section, we highlight three additional potentials that can be included into the MRF formulation for improved robustness.

Edge Potential: There is typically a strong gradient between obstacles and the road, thus we introduce an edge potential to bias cutting along spatial image gradients:

$$\phi_e(s_i) = -\nabla I_t(i, s_i)^2. \tag{4.21}$$

LIDAR Potential: While our primary motivation is an image-only solution, the MRF provides a convenient method to fuse online LIDAR measurements. Given a LIDAR point in the camera frame, $p = [x, y, z]^\top$, we project into the camera frame, $[i, j]^\top$ (here i and j are the projected column and row, respectively). Using the expected ground-plane depth image, $\hat{Z}_t$, we find the expected ground point $\hat{s}_i$ by traversing *down* the image column and

minimizing,

$$\hat{s}_i = \underset{\hat{s}_i}{\operatorname{argmin}} \left\| \hat{Z}_t\left(i, \hat{s}_i\right) - z \right\|. \tag{4.22}$$

The resulting potential is a truncated quadratic:

$$\phi_l\left(s_i\right) = \min(|s_i - \hat{s}_i|, T_l)^2, \tag{4.23}$$

where T_l is a threshold controlling the region of influence of the LIDAR potential. To prune obstacles that are too small and eliminate spurious returns, this potential is only added for each LIDAR point that meets a minimum height above ground threshold,

$$(\hat{s}_i - j) \cdot \hat{Z}_t(i, s_i)/f > H_{\text{LIDAR}}, \tag{4.24}$$

where $H_{\text{LIDAR}} = 0.5$ in our experiments. Fusing LIDAR data at this level improves our overall method because the sparse point returns can heavily dictate the models learned by the dense, image-based methods.

Recursive Potential: We introduce a recursive potential that propagates the full energy functional from the previous time step into the current frame, as proposed by Yao et al. (2015), which acts as a temporal smoothing of potentials. With our known ground-model and egomotion, we use the homogeneous point transfer (4.11) to propagate the sum over unary potentials of the previous frame into the current frame, generating $\phi_r\left(s_i\right)$.

4.4 Results

We evaluated our proposed method on our autonomous platform, a TORC ByWire XGV, that is equipped with Velodyne LIDAR scanners and a Point Grey Flea3 monochrome camera. The LIDAR scanners, unless otherwise specified, were used only offline for generating prior maps. Majority of the algorithms presented were implemented in CUDA and all experiments were run on a laptop equipped with a Core i7-4910MQ and a laptop GPU (NVIDIA Quadro K4100). The resulting implementation runs at 5–8 Hz.

4.4.1 Quantitative Analysis

Our approach is first evaluated against a hand-labeled dataset in which we have 240 ground-truth image partitions. In addition to our vision only solution, we also demonstrate the effectiveness of including a simple 2D LIDAR scanner to our system. Note that while our platform is not equipped with such a planar scanner, we simulated this with the Velodyne

scanners by only using point returns within a 40 cm window, 1 m off the ground in the *body frame*.

Following the metrics presented by Fritsch, Geiger, and Kühnl (2013), we project our image partition into the world to create a Bird's Eye View (BEV) before performing analysis—results are tabulated in Table 4.1. We see that our proposed method is competitive relative to prior work (Levi, Garnett, and Fetaya, 2015; Yao et al., 2015) and the addition of the LIDAR scanner dramatically improves obstacle detection.

Method	F1	Precision	Recall	FPR
Proposed	87.85 %	90.07 %	85.74 %	9.93 %
Proposed+2D LIDAR	93.18 %	94.65 %	91.75 %	5.35 %

Table 4.1: The F1-score, precision, recall, and false positive rate for our proposed method and our proposed method with the addition of 2D LIDAR measurements.

4.4.2 Qualitative Analysis

To demonstrate the contributions of each unary potential that is a part of the MRF model, we present several candidate image partitions along with an overlay of each potential, see Fig. 4.7—in these images, lighter (white) colors indicate a lower energy state. In this figure, all potentials presented in this chapter were enabled *except* the LIDAR potential.

In the first row, we see our platform exiting a brightly illuminated region into an area cast in shadow. Throughout the illuminated region, the appearance models overfit to this bright distribution and its potential is biased toward shaded/illuminated edges. Despite this, the image partitioning is still successful because of the optical flow potential. Several frames later, depicted in row 2, we see the appearance models have quickly adjusted to the new lighting.

The second and third row demonstrates the flexibility of our model to be able to perfectly follow the sharp contours of a pedestrian and a lightpost, respectively. One significant drawback of the optical flow potential is the effect of cast shadows from moving platforms, as shown in the third row—there is a gap of falsely detected obstacles triggered by the moving shadow to the right of the vehicle.

4.4.3 Obstacle Aware Localization

In our previous chapter (Chapter 3), we made no effort to eliminate non-static elements from the image, relying on the strong surface reflectivity to predominate the mutual information score. In many cases, the method was robust to sparse obstacles in view. However, this relied on the statistical robustness of the underlying mutual information cost metric. As

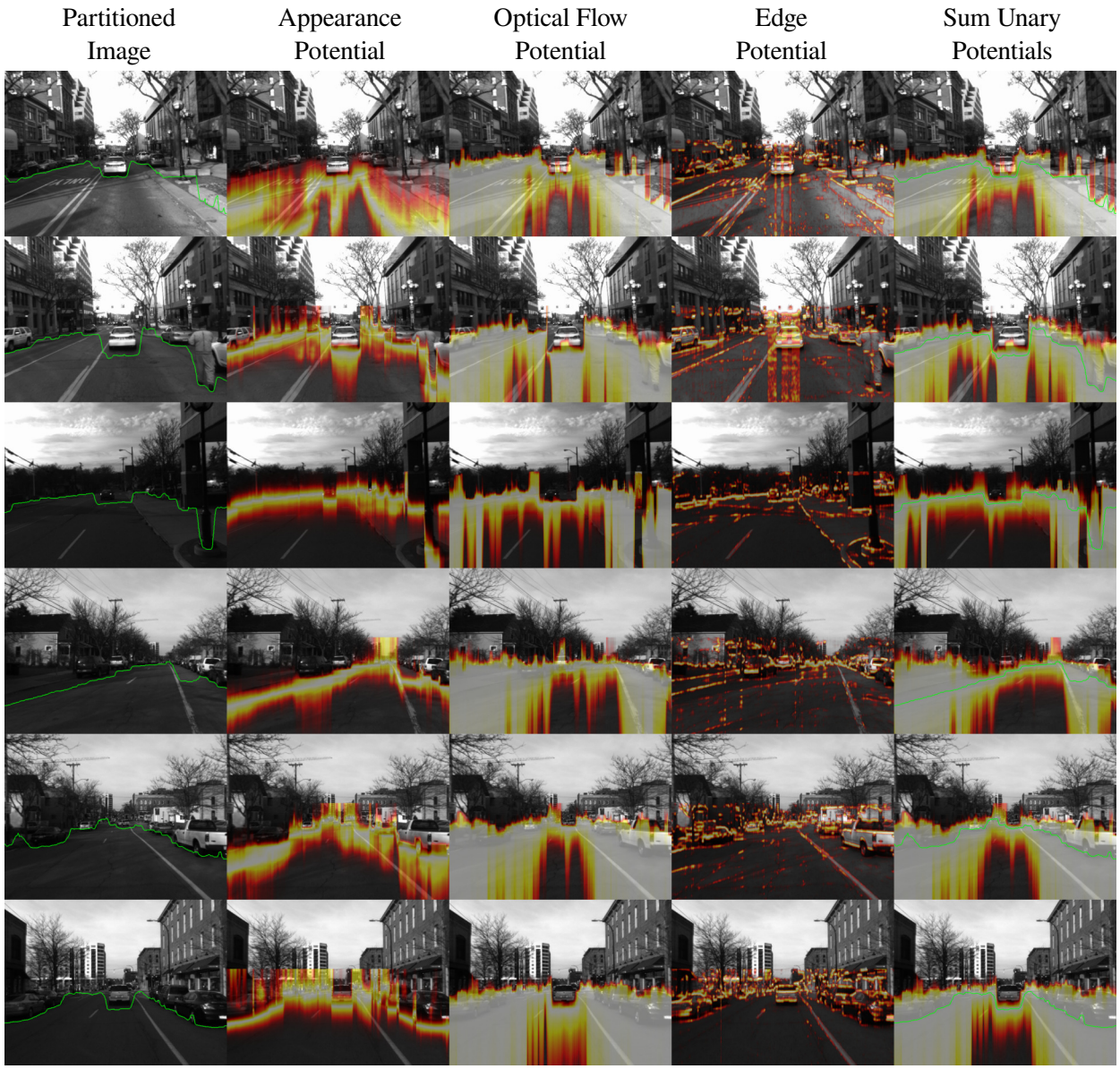


Figure 4.7: This figure shows sample results of our probabilistic obstacle partitioning. Left-to-right, we show the raw camera imagery overlaid with the extracted partition, then three primary potential functions, and finally the sum of all unary potentials. In each of these, lighter (white) colors indicate a lower energy state. See the text for a more detailed discussion.

more obstacles overtake the image, the algorithm can be distracted and lead to erroneous registrations.

In this section, we look at incorporating our image partitions into the localization pipeline. To do so, we only use pixels *below* the obstacle partition when computing the joint histogram tables—all other pixels are discarded. We performed a set of registration experiments similar to our previous chapter. Every second in our dataset, we attempted several registrations

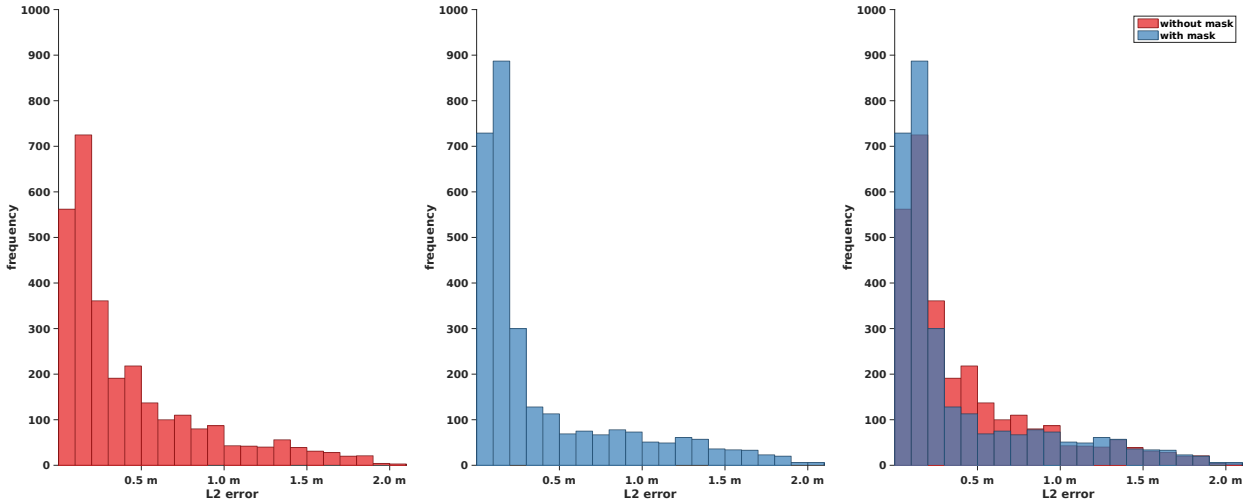
from a randomly initialized offsets (within a 3 m window around the ground-truth). From the same randomly initialized offsets, we attempted a registration without and with the use of obstacle masks.

A histogram of these errors can be seen in Fig. 4.8, where we see that the obstacle masks decrease out lateral error. This is frequent in our dataset as there is frequently perceptual aliasing in our logs over crosswalks—when obstacles partially occlude these crosswalks there is much lateral ambiguity present. A common improvement that we see is demonstrated in Fig. 4.9. Overall, we see a modest improvement in median absolute deviation (MAD) from 12.4 cm to 11.6 cm longitudinally, and 14.3 cm to 9.1 cm laterally.

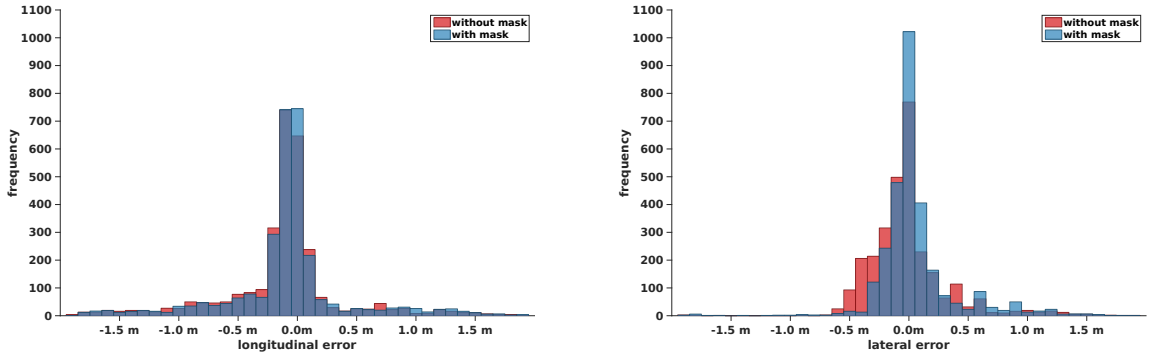
In addition to these benefits, we also see that the cost function is much more peaked when obstacle masks are used. Even in situations where registrations are successful without obstacle masks, we notice that the cost surface is significantly improved, see Fig. 4.10 for an example. In future work, we hope that this distinct improvement can help with gradient-based localization methods that can improve our image registration efficiency.

4.5 Conclusion

In this chapter, we showed that a monochrome, monocular camera can be used to partition an image into disjoint sets of obstacles and the ground plane. We utilized a textured prior map to derive appearance models and optical flow likelihoods that could be integrated into an MRF. The resulting formulation can be solved at a framerate of 5–8 Hz. We also demonstrated that the addition of sparse LIDAR returns can improve the entire pipeline. Furthermore, we integrated this into our visual localization pipeline and demonstrated improved robustness when obstacle partitions are considered during registration. In the future, we hope to use the extracted optical flow vectors to segment objects lying above the image partition, which can further be used to improve the recursive potential with a motion model.

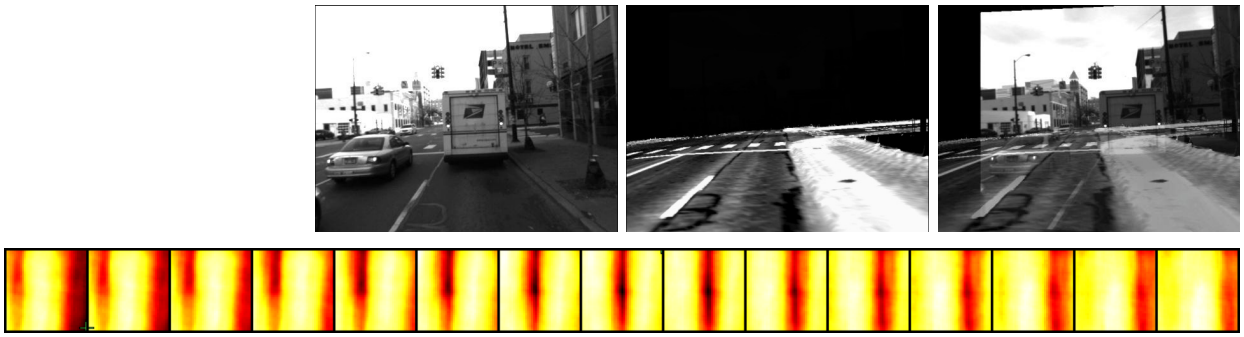


(a) L_2 Errors Without Mask (b) L_2 Errors With Mask (c) L_2 Errors Comparison

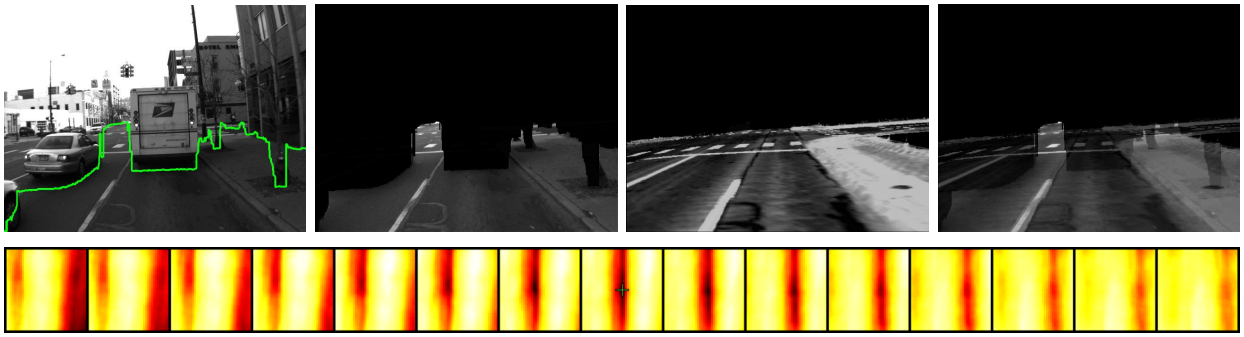


(d) Longitudinal Errors Comparison (e) Lateral Errors Comparison

Figure 4.8: Example of the improved registration errors when using our proposed obstacle partitioning to mask out obstacles during registration. In the first row, we show a histogram of L_2 (a) without our obstacle masks, (b) with our obstacle masks, and (c) the two overlaid to highlight differences. Furthermore, we split (c) into (d) longitudinal and (e) lateral registration errors, where we see that much improvement comes in our lateral registration—this is particularly an improvement at intersections with vehicles occluding our field of view.



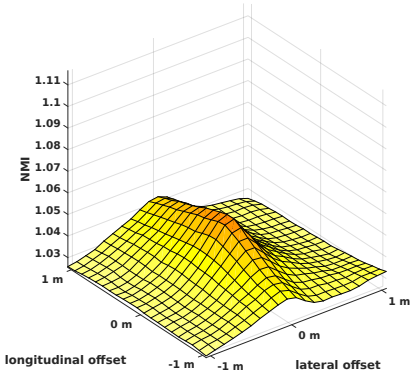
(a) Without obstacle masks



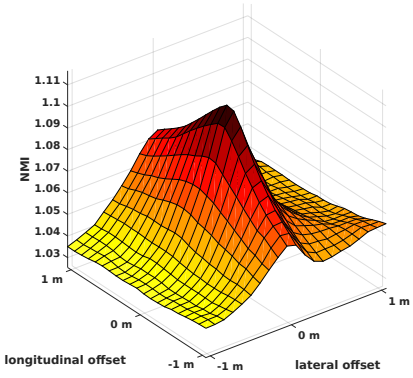
(b) With obstacle masks

Figure 4.9: From left-to-right in (a), we show three images: the source image, the best predicted synthetic LIDAR image, and an alpha-blending of the source and synthetic image. In (b), we added the source image with the obstacle partition drawn in *green*. The normalized mutual information cost map is also shown, where each tile in the cost map represents a different *heading* slice of the 3D cost surface, each pixel then represents an xy translation, and the maximum found is marked with a green '+'. Note, the cost surface should be maximized at the center pixel of the center tile. As demonstrated in Fig. 3.11(a), the NMI cost surface can be distracted by obstacles in the field of view, as is similarly shown in (a). However, using only pixels below the obstacle partition in our NMI evaluation, the cost surface is cleaned up, allowing the true registration to be the maximum.

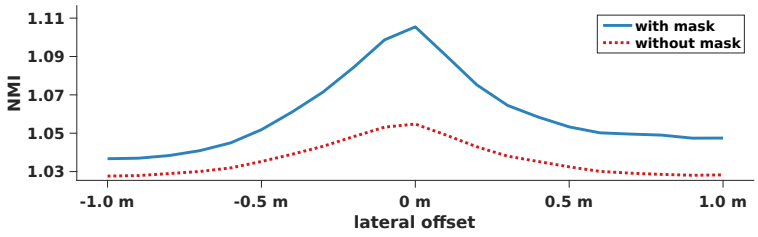
(a) Partitioned Image



(b) Cost surface without masks



(c) Cost surface with masks



(d) Cost surface slice at *long.*=0 m

Figure 4.10: Sample of improvement of the NMI cost surface when obstacle masks are used in the visual localization framework. In (a), we show a sample partitioned image, which we then explored the cost surface around. We evaluated the NMI at varying longitudinal and lateral offsets (b) without and (c) with the masks. Further, we extracted the lateral slice at 0 m longitudinal offset, displayed in (d). In these figures, we see that the cost function is more peaked and pronounced when obstacle masks are used—this increases the signal-to-noise-ratio by eliminating parts of the image (i.e., obstacles) that do not have a statistical dependence on the prior map.

CHAPTER 5

Conclusion

Self-driving cars often rely on precise localization within a prior map for autonomous navigation. These maps are annotated with rules of the road including precise lane markings, stop sign locations, speed limits, etc. Therefore, accurate localization within these maps provide the autonomous agent with a wealth of knowledge to influence its decision. The common approaches to localization often use a three-dimensional (3D) light detection and ranging (LIDAR) for registration against a LIDAR reflectivity ground-plane map or use cameras and extracted image features for registration against a dictionary of localized image features during a previous mapping run.

In this thesis, we looked at both of these sensing modalities and detailed methods for improving on the state-of-the-art. The fusion of our contributions yield a multi-modal localization system that is robust through poor weather including heavy snowfall, road construction overhauls, and even sensor failures given our redundant approach. Furthermore, our approach throughout this thesis focuses on utilization of raw sensor data as a mechanism for added robustness, thus avoiding common pitfalls of failed feature extraction.

5.1 Contributions

The specific contributions of this thesis include:

Gaussian Mixture Maps

We proposed Gaussian mixture maps (Chapter 2) as a method for condensing the full state of the world into a compact distribution of Gaussians characterizing the structure (i.e., 3D points) and appearance (i.e., reflectivity) of the environment. Using this, we proposed a multiresultion, branch-and-bound method using rasterized versions of these distributions to perform accurate and efficient localization into these maps. We demonstrated that our method that jointly reasons over structure and appearance allows our vehicle to remain localized under complex scenarios including heavy snowfall and road construction.

Visual Localization within LIDAR Maps

In Chapter 3, we introduced a visual localization pipeline that generates synthetic views of our 3D LIDAR map and performs whole image matching against these views. We demonstrated on a series of datasets that our method is able to achieve a similar order of magnitude error rate as LIDAR-based methods using a sensor that is several orders of magnitude cheaper.

Probabilistic Obstacle Partitioning

In Chapter 4, we presented a probabilistic approach to obstacle partitioning that defines a partition between ground-plane and obstacles in an image frame. We then used this method to improve the quality of our visual localization by improving registrations when the camera imagery is dominated by obstacles.

5.2 Future Work

There are many areas for future improvement or alternative use cases for our work. The remainder of this section discusses these potential areas of interest, split into a discussion over each technical chapter presented and a general discussion for the future of localization as a whole.

Gaussian Mixture Maps

Our proposed Gaussian mixture map formulation can provide a wealth of information for an autonomous car aside from that leveraged for localization. Specifically, as we briefly discussed in Chapter 2, the Gaussian mixture map provides a mechanism to probabilistically evaluate obstacle likelihood for a given point cloud point. Future work can consider a joint evaluation over localization and obstacle detection that can yield a framework for partitioning and tracking points temporally. Quite like the simultaneous localization and mapping (SLAM) problem, the reasoning of obstacle points from stationary points used for localization is a heavily interconnected problem.

Furthermore, our approach always assumed that the registration was between a point cloud and a well sampled prior map. Future work can consider our approach for scan to scan registration, quite similar to ordinary scan matching algorithms (Segal, Haehnel, and Thrun, 2009). Moreover, the application of our approach to other domains that are similarly well structured (such as indoor quadrotor localization) or use a slightly different sensing modality (such as sweeping, planar LIDAR scanners) provide interesting avenues for continued research.

Visual Localization within LIDAR Maps

While briefly mentioned in Chapter 3, future work with visual localization in LIDAR maps can consider alternative, compact map representations that allow for efficient localization in 3D maps. Our presented approach relies on meshed ground-plane maps as it improved efficiency of our system. The failure modes faced do seem to be correctable using a higher fidelity map (such as the issue faced when our platform is only constrained laterally). Furthermore, the addition of more cameras to our platform from various viewpoints can also be seen as areas for future work. Finally, we are interested in extending our whole image matching beyond localization within prior maps, instead looking to use the measurements in a visual odometry pipeline.

Probabilistic Obstacle Partitioning

Our proposed obstacle partitioning framework can be viewed as a system that can benefit from more diverse inputs. Introducing various deep learning and regression techniques can improve measurement likelihoods used in our probabilistic partitioning and can likely improve the quality of the system.

Furthermore, we see our approach that fuses LIDAR information into the segmentation problem as an initial step toward a fully joint segmentation process. Future work should further consider how each modality can be used simultaneously, as opposed to the more common approach of segmenting them independently and fusing their outputs at the tracking level.

General Areas for Future Work

Finally, a major area of future work is understanding the gaps and limitations of our proposed localization system. Throughout our work, we strive for a system that always maintains centimeter-level localization accuracy. However, transitioning this onto all cars covering all roads, there are bound to be environments or events that have not been accounted for. In certain circumstances, having precise localization could be impossible (e.g., a country road with no 3D structure or lane markings for position estimation), thus understanding how our work could be used in conjunction with a more rule-based agent (e.g., "drive down the right lane") is a large area for future work to improve robustness.

APPENDICES

APPENDIX A

Odometry Model

In this chapter, we detail the vehicle odometry model that is used throughout this thesis. This odometry model incorporates measurements from an Applanix POS-LV 420 inertial navigation system (INS) with external wheel encoder and dual global positioning system (GPS) antennas for estimating incremental vehicle motion.

The vehicle state that we are concerned with estimating, ${}^{l}\boldsymbol{\mu}_k$, is defined by the 6-vector,

$$ {}^{l}\boldsymbol{\mu}_k = \left[{}^{l}\mathbf{t}_k^\top, {}^{l}\boldsymbol{\Theta}_k^\top \right]^\top = \left[{}^{l}x_k, {}^{l}y_k, {}^{l}z_k, {}^{l}r_k, {}^{l}p_k, {}^{l}h_k \right]^\top , \tag{A.1} $$

which is the vehicle pose relative to the local navigation frame, l, at time k. Here, ${}^{l}\mathbf{t}_k$ is a translation 3-vector expressed relative to frame l and ${}^{l}\boldsymbol{\Theta}_k$ is the corresponding 3-vector of Euler angles with r representing roll about the x axis, p representing pitch about the y axis, and h representing heading about the z axis. When the vehicle is powered on, this local frame is initialized at the origin, ${}^{l}\boldsymbol{\mu}_0 = [0, 0, 0, 0, 0, 0]^\top$.

The Applanix INS provides state observations at 100 Hz of the form:

$$ {}^{l}\mathbf{u}_k = \left[{}^{l}\mathbf{v}_k^\top, {}^{l}\mathbf{w}_k^\top \right]^\top , \tag{A.2} $$

where ${}^{l}\mathbf{v}_k$ is the measured platform velocities with respect to the local frame and ${}^{l}\mathbf{w}_k$ is the measured Euler angle orientations. We then define a discrete update process for estimating vehicle odometry in terms of the previous odometry estimate,

$$ {}^{l}\boldsymbol{\mu}_k = f\left({}^{l}\boldsymbol{\mu}_{k-1}, {}^{l}\mathbf{u}_k \right) . \tag{A.3} $$

The function $f(\,\cdot\,)$ integrates the velocity over time while directly using the doubly integrated

orientation performed internal to the Applanix. Specifically, $f(\cdot)$ is computed as:

$$^{l}\mathbf{t}_k = {}^{l}\mathbf{t}_{k-1} + \Delta t \cdot {}^{l}\hat{\mathbf{v}}_k, \tag{A.4}$$

$$^{l}\boldsymbol{\Theta}_k = {}^{l}\hat{\mathbf{w}}_k. \tag{A.5}$$

A.1 Uncertainty Estimation

As frequently required in robotics applications, it is also necessary to estimate odometry uncertainty for the robot. This is especially important when incorporating odometry estimates in a filtering application, as done in Chapter 2 and Chapter 3. Moreover, characterizing vehicle odometry allows us to probabilistically evaluate perceptual data as demonstrated in Chapter 4.

We define our uncertainty estimation using a parametric model for incremental uncertainty propagation. This is defined using the incremental "delta odometry",

$$\Delta\boldsymbol{\mu}_{k-1,k} = \ominus\boldsymbol{\mu}_{k-1} \oplus \boldsymbol{\mu}_k, \tag{A.6}$$

which is the tail-to-tail composition that expresses the current odometry estimate at time k relative to the belief at time $k-1$ (Smith, Self, and Cheeseman, 1990). This allows us to propagate uncertainty by performing first-order covariance propagation of $\boldsymbol{\mu}_k = \boldsymbol{\mu}_{k-1} \oplus \Delta\boldsymbol{\mu}_{k-1,k}$, arriving at:

$$\Sigma_k = \begin{bmatrix} \mathrm{J}_{\oplus 1} & \mathrm{J}_{\oplus 2} \end{bmatrix} \begin{bmatrix} \Sigma_{k-1} & 0 \\ 0 & \Delta\Sigma_{k-1,k} \end{bmatrix} \begin{bmatrix} \mathrm{J}_{\oplus 1} & \mathrm{J}_{\oplus 2} \end{bmatrix}^{\top}, \tag{A.7}$$

where $\mathrm{J}_{\oplus 1}$ is the partial derivative of $\boldsymbol{\mu}_k$ with respect to $\boldsymbol{\mu}_{k-1}$, $\mathrm{J}_{\oplus 2}$ is the partial derivative of $\boldsymbol{\mu}_k$ with respect to $\Delta\boldsymbol{\mu}_{k-1,k}$, and $\Delta\Sigma_{k-1,k}$ is the incremental uncertainty accrued from $k-1$ to k. This can be familiarly seen in the Kalman prediction step as,

$$\Sigma_k = \mathrm{J}_{\oplus 1}\Sigma_{k-1}\mathrm{J}_{\oplus 1}^{\top} + \mathrm{J}_{\oplus 2}\Delta\Sigma_{k-1,k}\mathrm{J}_{\oplus 2}^{\top}, \tag{A.8}$$

where $\mathrm{J}_{\oplus 2}$ transforms the incremental noise, $\Delta\Sigma_{k-1,k}$, into the local frame (i.e., $\mathrm{Q}_k = \mathrm{J}_{\oplus 2}\Delta\Sigma_{k-1,k}\mathrm{J}_{\oplus 2}^{\top}$).

In many circumstances, $\Delta\Sigma_{k-1,k}$ is set to a fixed covariance. However, we instead fit a parametric model as the true uncertainty is a function of the control action taken from $k-1$ to k (e.g., higher speeds should incur greater magnitude uncertainty in the direction of travel). Thus, we use an approach similar to that proposed by Hu and Kantor (2015), where we use a

parametric model that is a function of a set of extracted features from the "delta odometry",

$$\Delta\Sigma_{k-1,k} = \mathrm{G}\left(\Delta\boldsymbol{\mu}_{k-1,k}\right) = \sum_{i=1}^{n} g_i\left(\Delta\boldsymbol{\mu}_{k-1,k}\right)^2 \cdot \Sigma_{\theta_i}, \tag{A.9}$$

where $g_i(\,\cdot\,)$ are a set of n functions that extract features from the "delta odometry" and apply a weighting of the learned parameterized covariance Σ_{θ_i}. Throughout this thesis we use two features, though the method could be applied to an arbitrary number of features. The first feature used is the L_2 distance of the odometry step, which captures errors from the wheel encoder. The second feature is the "delta heading" over the odometry step to account for errors that are correlated with turning. These are defined as:

$$g_1\left(\Delta\boldsymbol{\mu}_{k-1,k}\right) = \|\Delta\mathbf{t}_{k-1,k}\|, \tag{A.10}$$

$$g_2\left(\Delta\boldsymbol{\mu}_{k-1,k}\right) = \|\Delta h_{k-1,k}\|. \tag{A.11}$$

The corresponding covariances to be learned, Σ_{θ_1} and Σ_{θ_2}, are decomposed into their Cholesky factorization to avoid rank deficient covariances:

$$\Sigma_{\theta_1} = \mathrm{L}_{\theta_1}\mathrm{L}_{\theta_1}^{\top}, \tag{A.12}$$

$$\Sigma_{\theta_2} = \mathrm{L}_{\theta_2}\mathrm{L}_{\theta_2}^{\top}. \tag{A.13}$$

Thus, θ_1 and θ_2 are each a 21-element parameter vector (42 total model parameters):

$$\mathrm{L}_{\theta_1} = \begin{bmatrix} \theta_{1,1} & 0 & 0 & 0 & 0 & 0 \\ \theta_{1,7} & \theta_{1,2} & 0 & 0 & 0 & 0 \\ \theta_{1,12} & \theta_{1,8} & \theta_{1,3} & 0 & 0 & 0 \\ \theta_{1,16} & \theta_{1,13} & \theta_{1,9} & \theta_{1,4} & 0 & 0 \\ \theta_{1,19} & \theta_{1,17} & \theta_{1,14} & \theta_{1,10} & \theta_{1,5} & 0 \\ \theta_{1,21} & \theta_{1,20} & \theta_{1,18} & \theta_{1,15} & \theta_{1,11} & \theta_{1,6} \end{bmatrix}, \mathrm{L}_{\theta_2} = \begin{bmatrix} \theta_{2,1} & 0 & 0 & 0 & 0 & 0 \\ \theta_{2,7} & \theta_{2,2} & 0 & 0 & 0 & 0 \\ \theta_{2,12} & \theta_{2,8} & \theta_{2,3} & 0 & 0 & 0 \\ \theta_{2,16} & \theta_{2,13} & \theta_{2,9} & \theta_{2,4} & 0 & 0 \\ \theta_{2,19} & \theta_{2,17} & \theta_{2,14} & \theta_{2,10} & \theta_{2,5} & 0 \\ \theta_{2,21} & \theta_{2,20} & \theta_{2,18} & \theta_{2,15} & \theta_{2,11} & \theta_{2,6} \end{bmatrix}.$$

These parameters are learned from the maximum likelihood estimate (MLE) of ground-truth training data. Given a set of m "delta odometry" samples, $\{\Delta\boldsymbol{\mu}_j\}_{j=1}^{m}$, with corresponding ground-truth, $\{\Delta\hat{\boldsymbol{\mu}}_j\}_{j=1}^{m}$, we are interested in finding the parameters that maximize the log-likelihood,

$$\hat{\theta}_1, \hat{\theta}_2 = \underset{\theta_1,\theta_2}{\operatorname{argmax}} \sum_j -\frac{1}{2}\log \mathrm{G}\left(\Delta\boldsymbol{\mu}_j\right) - \frac{1}{2}\epsilon_j^{\top}\mathrm{G}\left(\Delta\boldsymbol{\mu}_j\right)^{-1}\epsilon_j, \tag{A.14}$$

where $\epsilon_j = \Delta\boldsymbol{\mu}_j - \Delta\hat{\boldsymbol{\mu}}_j$ is the error for the j^{th} sample. This likelihood treats each sample independently and evaluates against the Gaussian density function. Further, we found that constraining elements along the diagonal (i.e., the first 6 parameters of θ_1 and θ_2) to be strictly greater than *zero* led to faster convergence and more stable parameters. The algorithm was implemented using GSL's `multimin` (Gough, 2009) in which numerical gradients were used. Moreover, it is important to generate random samples that well explore the feature space so that the model can generalize over the features.

APPENDIX B

Offline SLAM Pipeline

In this chapter, we detail the offline simultaneous localization and mapping (SLAM) pipeline that we use for constructing maps and generating ground-truth for our experiments.

Prior to the offline mapping stage, our robot has no *a priori* knowledge of the environment, thus, we must use the light detection and ranging (LIDAR) scanners and inertial sensors to build a model of the environment while simultaneously localizing within this environment. We use the state-of-the-art in nonlinear least-squares, pose-graph SLAM to map the three-dimensional (3D) structure in a globally consistent frame.

We construct a pose-graph to solve the full SLAM problem as shown in Fig. B.1, where nodes in the graph are poses (X) and edges are either odometry constraints (U) as outlined in Appendix A, laser scan matching constraints (Z), or GPS prior constraints (G). These constraints are modeled as Gaussian random variables; therefore, we model the joint distribution over poses and constraints as

$$P(X,U,Z,G) = \prod_{i=1}^{M} P(\mathbf{x}_i|\mathbf{x}_{i-1},\mathbf{u}_i) \prod_{k=1}^{K} P(\mathbf{z}_k|\mathbf{x}_{i_k},\mathbf{x}_{j_k}) \prod_{a=1}^{A} P(\mathbf{g}_a|\mathbf{x}_a) \tag{B.1}$$

$$\propto \prod_{i=1}^{M} e^{-\frac{1}{2}\|f_i(\mathbf{x}_{i-1},\mathbf{u}_i)-\mathbf{x}_i\|^2_{\Sigma_i}} \prod_{k=1}^{K} e^{-\frac{1}{2}\|h_k(\mathbf{x}_{i_k},\mathbf{x}_{j_k})-\mathbf{z}_k\|^2_{\Sigma_k}} \prod_{a=1}^{A} e^{-\frac{1}{2}\|h_a(\mathbf{x}_a)-\mathbf{g}_a\|^2_{\Sigma_a}}, \tag{B.2}$$

where there are M poses, K loop closures, and A GPS prior constraints. Thus, to solve the SLAM problem, we seek to find the maximum *a posteriori* (MAP) estimate over the robot

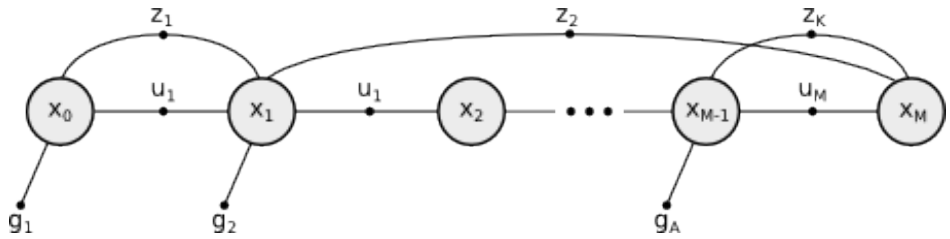


Figure B.1: Factor graph of the pose-graph SLAM problem that we solve in the offline mapping stage. Here, x_i represents states of the robot, u_m represents incremental odometry measurements, z_k represents laser scan-matching constraints, and g_a are GPS prior measurements.

poses by minimizing the negative log of the joint probability:

$$X^* = \arg\max_X P(X, U, Z, G) \tag{B.3}$$

$$= \arg\min_X -\log P(X, U, Z, G) \tag{B.4}$$

$$\begin{aligned} = \arg\min_X & \sum_{i=1}^{M} \|f_i(\mathbf{x}_{i-1}, \mathbf{u}_i) - \mathbf{x}_i\|^2_{\Sigma_i} \\ & + \sum_{k=1}^{K} \|h_k(\mathbf{x}_{i_k}, \mathbf{x}_{j_k}) - \mathbf{z}_k\|^2_{\Sigma_k} \\ & + \sum_{a=1}^{A} \|h_k(\mathbf{x}_a) - \mathbf{g}_a\|^2_{\Sigma_a}, \end{aligned} \tag{B.5}$$

where $f_i(\,\cdot\,)$ is our process model, $h_k(\,\cdot\,)$ is our scan matching measurement model, and $h_a(\,\cdot\,)$ is our GPS measurement model. Each is corrupted by normally distributed noise with covariance Σ_i, Σ_k, and Σ_a, respectively. This summation equates to solving a nonlinear least-squares problem. We use incremental smoothing and mapping (iSAM) (Kaess, Ranganathan, and Dellaert, 2008), which uses incremental QR factorization to solve this nonlinear least-squares problem.

Since map construction is an offline task, we do not construct our pose-graph and make loop closures "online". Instead, we first construct a graph with only odometry and GPS prior constraints. With this skeleton pose-graph in the near vicinity of the global optimum, we use Segal, Haehnel, and Thrun (2009)'s generalized iterative closest point (GICP) to establish 6-degree of freedom (DOF) laser scan-matching constraints between poses; adding both odometry constraints (temporally neighboring poses) and loop closure constraints (spatially neighboring poses) to our pose-graph. Furthermore, we also include reflectivity-based scan-matching constraints that only consider the appearance of the point clouds; this is formulated

as a 3-DOF constraint (i.e., relative x, y, and heading), optimizing a cost function similar to Levinson, Montemerlo, and Thrun (2007).

B.1 Ground-Truth

We further use this SLAM pipeline to generate ground-truth for evaluation. Given a newly acquired vehicle trajectory, we are interested in finding its optimal trajectory, $Y = \{\mathbf{y}_j\}_{j=1}^{N}$. Without loss of generality, a pose-graph over the variables in Y can be constructed as those in X above; however, in this section we are further interested in expressing the location of poses in Y *relative* to the poses in X. Thus, we are interested in stitching Y into the mapping pose-graph over X

This is done by establishing laser scan-matching constraints between poses in X and poses in Y. Unlike the previous section, we cannot directly add a pairwise factor to our factor graph because it would cause the underlying map's pose-graph to change (i.e., estimates of each $\mathbf{x}_i \in X$ needs to be *fixed* as maps are constructed once from this data). Instead, we use pose composition to generate artificial prior factors that can be added to our factor graph.

Given a 6-DOF scan-matching constraint, $\mathbf{z}_k$, that measures the position of $\mathbf{y}_{j_k}$ with respect to $\mathbf{x}_{i_k}$, we calculate this artificial prior,

$$\mathbf{z}'_k = \mathbf{x}_{i_k} \oplus \mathbf{z}_k, \tag{B.6}$$

where $\oplus$ is the head-to-tail composition operation (Smith, Self, and Cheeseman, 1990). We further propagate uncertainty through this transformation to arrive at

$$\Sigma'_k = \begin{bmatrix} \mathrm{J}_{\oplus 1} & \mathrm{J}_{\oplus 2} \end{bmatrix} \begin{bmatrix} \Sigma_{\mathbf{x}_{i_k}} & 0 \\ 0 & \Sigma_k \end{bmatrix} \begin{bmatrix} \mathrm{J}_{\oplus 1} & \mathrm{J}_{\oplus 2} \end{bmatrix}^\top, \tag{B.7}$$

where $\Sigma_{\mathbf{x}_{i_k}}$ is the marginal covariance from the *fixed* pose-graph over X and Σ_k is the scan-matching uncertainty as above. Further, $\mathrm{J}_{\oplus 1}$ is the partial derivative of $\mathbf{z}'_k$ with respect to $\mathbf{x}_{i_k}$ and $\mathrm{J}_{\oplus 2}$ is the partial derivative of $\mathbf{z}'_k$ with respect to $\mathbf{z}_k$. The prior factor with mean $\mathbf{z}'_k$ and covariance Σ'_k are then added to the pose-graph—this is done similar to the GPS prior factors.

BIBLIOGRAPHY

BIBLIOGRAPHY

A. A. Aghamohammadi, H. D. Taghirad, A. H. Tamjidi, and E. Mihankhah. Feature-based laser scan matching for accurate and high speed mobile robot localization. In *Proceedings of the European Conference on Mobile Robots*, Freiburg, Germany, Sept. 2007.

J. M. Álvarez and A. M. Ĺopez. Road detection based on illuminant invariance. *IEEE Transactions on Intelligent Transportation Systems*, 12(1):184–193, Oct. 2011.

J. M. Alvarez, T. Gevers, Y. LeCun, and A. M. Lopez. Road scene segmentation from a single image. In *Proceedings of the European Conference on Computer Vision*, pages 376–389. Springer, Firenze, Italy, Oct. 2012.

A. Angeli, S. Doncieux, J. A. Meyer, and D. Filliat. Visual topological slam and global localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4300–4305, Kobe, Japan, May 2009.

H. Badino, U. Franke, and D. Pfeiffer. The stixel world – a compact medium level representation of the 3d-world. In *Proceedings of the DAGM Symposium on Pattern Recognition*, volume 5748, pages 51–60, Jena, Germany, Sept. 2009.

I. Baldwin and P. Newman. Road vehicle localization with 2d push-broom lidar and 3d priors. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2611–2617, Saint Paul, MN, USA, May 2012.

L. Bao, Q. Yang, and H. Jin. Fast edge-preserving patchmatch for large displacement optical flow. *IEEE Transactions on Image Processing*, 23(12):4996–5006, Dec. 2014.

Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation*. John Wiley & Sons, Inc., New York, 2001.

C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), Aug. 2009.

C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized PatchMatch correspondence algorithm. In *Proceedings of the European Conference on Computer Vision*, pages 29–43, Heraklion, Crete, Greece, Sept. 2010.

J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:43–77, 1994.

F. Becker, F. Lenzen, J. H. Kappes, and C. Schnörr. Variational recursive joint estimation of dense scene structure and camera motion from monocular high speed traffic sequences. *International Journal of Computer Vision*, 105(3):269–297, 2013.

P. Beeson, J. Modayil, and B. Kuipers. Factoring the mapping problem: Mobile robot map-building in the hybrid spatial semantic hierarachy. *International Journal of Robotics Research*, 29(4):428–459, 2009.

P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

P. Biber. The normal distribution transform: A new approach to laser scan matching. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2473–2748, Las Vegas, NV, USA, Oct. 2003.

J.-L. Blanco, J.-A. Fernández-Madrigal, and J. Gonzalez. Toward a unified bayesian approach to hybrid metric–topological slam. *IEEE Transactions on Robotics*, 24(2):259–270, 2008.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, Nov. 2001.

G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly, 2008.

C. Braillon, C. Pradalier, J. L. Crowley, and C. Laugier. Real-time moving obstacle detection using optical flow models. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 466–471, Tokyo, Japan, June 2006.

G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *Proceedings of the European Conference on Computer Vision*, pages 44–57, Marseille, France, Oct. 2008.

T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (3):500–513, 2011.

T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proceedings of the European Conference on Computer Vision*, pages 25–36, Prague, Czech Republic, May 2004.

T. Brox, C. Bregler, and J. Malik. Large displacement optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 41–48, Miami, FL, USA, June 2009.

A. Bruhn, J. Weickert, and C. Schnörr. Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3): 211–231, 2005.

N. Carlevaris-Bianco and R. M. Eustice. Learning temporal co-observability relationships for lifelong robotic mapping. In *IROS Workshop on Lifelong Learning for Mobile Robotics Applications*, Vilamoura, Portugal, Oct. 2012.

B. Charmette, E. Royer, and F. Chausse. Efficient planar features matching for robot localization using GPU. In *IEEE Workshop on Embedded Computer Vision*, pages 16–23, San Francisco, CA, June 2010.

S. M. Chaves, R. W. Wolcott, and R. M. Eustice. NEEC research: Toward GPS-denied landing of unmanned aerial vehicles on ships at sea. *Naval Engineers Journal*, 127(1): 23–35, 2015.

Z. Chen, H. Jin, Z. Lin, S. Cohen, and Y. Wu. Large displacement optical flow from nearest neighbor fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2443–2450, Portland, OR, USA, June 2013.

Z. J. Chong, B. Qin, T. Bandyopadhyay, M. H. Ang Jr., E. Frazzoli, and D. Rus. Synthetic 2d lidar for precise vehicle localization in 3d urban environment. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1554–1559, Karlsruhe, Germany, May 2013.

H. Choset and K. Nagatani. Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2):125–137, 2001.

W. Churchill and P. Newman. Continually improving large scale long term visual navigation of a vehicle in dynamic urban environments. In *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, pages 1371–1376, Anchorage, AK, USA, Sept. 2012.

M. Cummins and P. Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research*, 27(6):647–665, 2008.

J. Cunha, E. Pedrosa, C. Cruz, A. J. Neves, and N. Lau. Using a depth camera for indoor robot localization and navigation. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, Los Angeles, CA, USA, June 2011.

H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. In *Proceedings of Robotics: Science and Systems*, Philadelphia, PA, USA, Aug. 2006.

A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):1052–1067, 2007.

F. Dellaert and M. Kaess. Square root SAM. *International Journal of Robotics Research*, 25 (12):1181–1204, 2006.

F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1322–1328, Detroit, MI, May 1999.

E. Eade and T. Drummond. Scalable monocular SLAM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 469–476, New York, NY, USA, June 2006.

J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1449–1456, Sydney, Australia, Dec. 2013.

J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *Proceedings of the European Conference on Computer Vision*, pages 834–849, Zurich, Switzerland, Sept. 2014.

S. Engelson. *Passive Map Learning and Visual Place Recognition*. PhD thesis, Department of Computer Science, Yale University, 1994.

W. Enkelmann, V. Gengenbach, W. Kruger, S. Rossle, and W. Tolle. Obstacle detection by real-time optical flow evaluation. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 97–102, Paris, France, Oct. 1994.

R. Eustice, O. Pizarro, and H. Singh. Visually augmented navigation in an unstructured environment using a delayed state history. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 25–32, New Orleans, LA, USA, Apr. 2004.

R. Eustice, M. Walter, and J. Leonard. Sparse extended information filters: Insights into sparsification. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3281–3288, Edmonton, Canada, Aug. 2005.

R. M. Eustice. *Large-area visually augmented navigation for autonomous underwater vehicles*. PhD thesis, Massachusetts Institute of Technology / Woods Hole Oceanographic Institution Joint Program, Cambridge, MA, June 2005.

R. M. Eustice, H. Singh, and J. J. Leonard. Exactly sparse delayed-state filters for view-based SLAM. *IEEE Transactions on Robotics*, 22(6):1100–1114, 2006.

M. F. Fallon, H. Johannsson, and J. J. Leonard. Efficient scene simulation for robust Monte Carlo localization using an RGB-D camera. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1663–1670, Saint Paul, MN, USA, May 2012.

G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis*, pages 363–370, Berlin, Heidelberg, June 2003. Springer-Verlag.

P. F. Felzenszwalb and O. Veksler. Tiered scene labeling with dynamic programming. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3097–3104, San Francisco, CA, USA, June 2010.

P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept. 2010.

C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 15–22, Hong Kong, China, May 2014.

D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, volume 1999, pages 343–349, Orlando, FL, USA, July 1999.

D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, pages 391–427, 1999.

J. Fritsch, A. Geiger, and T. Kühnl. A new performance measure and evaluation benchmark for road detection algorithms. In *IEEE Conference on Intelligent Transportation Systems*, pages 1693–1700, The Hague, Netherlands, Oct. 2013.

B. Gough. *GNU Scientific Library Reference Manual — Third Edition*. Network Theory Ltd., 2009. ISBN 0954612078.

S. Granger and X. Pennec. Multi-scale EM-ICP: A fast and robust approach for surface registration. In *Proceedings of the European Conference on Computer Vision*, pages 418–432, Copenhagen, Denmark, May 2002.

J.-S. Gutmann, T. Weigel, and B. Nebel. A fast, accurate and robust method for self-localization in polygonal environments using laser range finders. *Advanced Robotics*, 14(8): 651–667, 2001.

M. Haag and H.-H. Nagel. Combination of edge element and optical flow estimates for 3d-model-based vehicle tracking in traffic image sequences. *International Journal of Computer Vision*, 35(3):295–319, 1999.

R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

D. Held, J. Levinson, and S. Thrun. A probabilistic framework for car detection in images using context and scale. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1628–1634, Saint Paul, MN, USA, May 2012.

B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17: 185–203, 1981.

H. Hu and G. Kantor. Parametric covariance prediction for heteroscedastic noise. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3052–3057, Hamburg, Germany, Sept. 2015.

M. Irani and P. Anandan. About direct methods. In *Vision Algorithms: Theory and Practice*, pages 267–277. Springer, 2000.

K. Irie and M. Tomono. Road recognition from a single image using prior information. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1938–1945, Tokyo, Japan, Nov. 2013.

M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, 31(2):216–235, 2012.

A. Kim and R. M. Eustice. Real-time visual SLAM for autonomous underwater hull inspection using visual saliency. *IEEE Transactions on Robotics*, 29(3):719–733, June 2013.

A. Kitanov, S. Bisevac, and I. Petrovic. Mobile robot self-localization in complex indoor environments using monocular vision and 3D model. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 1–6, Zurich, Switzerland, Sept. 2007.

G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, Nara, Japan, 2007.

K. Konolige and M. Agrawal. FrameSLAM: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, 2008.

K. Konolige and J. Bowman. Towards lifelong visual maps. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1156–1163, St. Louis, MO, USA, Oct. 2009.

W. Krüger, W. Enkelmann, and S. Rössle. Real-time estimation and tracking of optical flow vectors for obstacle detection. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 304–309, Detroit, MI, USA, Sept. 1995.

R. Kümmerle, R. Triebel, P. Pfaff, and W. Burgard. Monte carlo localization in outdoor terrains using multilevel surface maps. *Journal of Field Robotics*, 25:346–359, June - July 2008.

R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3607–3613, Shanghai, China, May 2011.

J. J. Leonard and H. F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1442–1447, Osaka, Japan, 1991a.

J. J. Leonard and H. F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *Proceedings of the IEEE International Conference on Robotics and Automation*, 7 (3):376–382, 1991b.

J. J. Leonard and R. J. Rikoski. Incorporation of delayed decision making into stochastic mapping. In *Experimental Robotics VII, D. Rus and S. Singh, Eds., Lecture Notes in Control and Information Sciences*, pages 533–542. Springer-Verlag, 2000.

D. Levi, N. Garnett, and E. Fetaya. Stixelnet: A deep convolutional network for obstacle detection and road segmentation. In *Proceedings of the British Machine Vision Conference*, pages 109.1–109.12, Swansea, United Kingdom, Sept. 2015.

J. Levinson and S. Thrun. Robust vehicle localization in urban environments using probabilistic maps. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4372–4378, Anchorage, AK, May 2010.

J. Levinson and S. Thrun. Unsupervised calibration for multi-beam lasers. In *Experimental Robotics*, pages 179–193. Springer, 2014.

J. Levinson, M. Montemerlo, and S. Thrun. Map-based precision vehicle localization in urban environments. In *Proceedings of the Robotics: Science & Systems Conference*, Atlanta, GA, June 2007.

C. Liu. *Beyond Pixels: Exploring New Representations and Applications for Motion Analysis*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, May 2009.

M. Liu, S. Lin, S. Ramalingam, and O. Tuzel. Layered interpretation of street view images. In *Proceedings of the Robotics: Science & Systems Conference*, Rome, Italy, July 2015.

M. I. Lourakis and S. C. Orphanoudakis. Visual detection of obstacles assuming a locally planar ground. In *Proceedings of the Asian Conference on Computer Vision*, pages 527–534. Hong Kong, China, Jan. 1998.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, Canada, Apr. 1981.

W. Maddern, G. Pascoe, and P. Newman. Leveraging Experience for Large-Scale LIDAR Localisation in Changing Cities. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1684–1691, Seattle, WA, USA, May 2015.

J. Maddox. Improving driving safety through automation. In *Congressional Robotics Caucus*. NHTSA, July 2012.

F. Maes, A. Collignon, D. Vandermeulen, G. Marchal, and P. Suetens. Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*, 16(2):187–198, 1997.

M. Magnusson. *The Three-Dimensional Normal-Distributions Transform—An Efficient Representation for Registration, Surface Analysis, and Loop Detection*. PhD thesis, Örebro University, Dec. 2009.

D. Maier, A. Hornung, and M. Bennewitz. Real-time navigation in 3D environments based on depth camera data. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robotics*, pages 692–697, Osaka, Japan, Nov. 2012.

C. McManus, W. Churchill, A. Napier, B. Davis, and P. Newman. Distraction suppression for vision-based pose estimation at city scales. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3762–3769, Karlsruhe, Germany, May 2013.

N. Molton, A. J. Davison, and I. Reid. Locally planar patch features for real-time structure from motion. In *Proceedings of the British Machine Vision Conference*, pages 1–10, London, United Kindom, Sept. 2004.

M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 593–598, Edmonton, Canada, July 2002.

M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the International Joint Conference on Artifical Intelligence*, pages 1151–1156, Acapulco, Mexico, Aug. 2003.

A. Napier and P. Newman. Generation and exploitation of local orthographic imagery for road vehicle localisation. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 590–596, Madrid, Spain, June 2012.

R. Negenborn. *Robot localization and Kalman filters*. PhD thesis, Utrecht University, 2003.

R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, Basel, Switzerland, Oct. 2011.

R. A. Newcombe, S. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2320–2327, Barcelona, Spain, Nov. 2011.

E. Olson. Real-time correlative scan matching. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4387–4393, Kobe, Japan, June 2009.

E. Olson. M3RSM: Many-to-many multi-resolution scan matching. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5815–5821, Seattle, WA, USA, June 2015.

E. Olson, J. J. Leonard, and S. Teller. Robust range-only beacon localization. *IEEE Journal of Oceanic Engineering*, 31(4):949–958, Oct. 2006a.

E. B. Olson, J. J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2262–2269, Orlando, FL, USA, May 2006b.

P. Ozog and R. M. Eustice. Toward long-term, automated ship hull inspection with visual SLAM, explicit surface optimization, and generic graph-sparsification. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3832–3839, Hong Kong, China, June 2014.

G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice. Visually bootstrapped generalized ICP. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2660–2667, Shanghai, China, May 2011.

G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice. Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 2053–2059, Toronto, Canada, July 2012.

G. Pascoe, W. Maddern, A. D. Stewart, and P. Newman. FARLAP: Fast Robust Localisation using Appearance Priors. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Seattle, WA, USA, May 2015.

D. Pfeiffer and U. Franke. Towards a global optimal multi-layer stixel representation of dense 3d data. In *Proceedings of the British Machine Vision Conference*, pages 51.1–51.12, Dundee, United Kingdom, Sept. 2011.

M. Pizzoli, C. Forster, and D. Scaramuzza. REMODE: Probabilistic, monocular dense reconstruction in real time. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2609–2616, Hong Kong, China, May 2014.

J. P. Pluim, J. A. Maintz, and M. A. Viergever. Mutual-information-based registration of medical images: A survey. *IEEE Transactions on Medical Imaging*, 22(8):986–1004, 2003.

A. Ranganathan, D. Ilstrup, and T. Wu. Light-weight localization for vehicles using road markings. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 921–927, Tokyo, Japan, Nov. 2013.

N. Ripperda and C. Brenner. Marker-free registration of terrestrial laser scans using the normal distribution transform. In *Proceedings of the ISPRS Working Group V/4 Workshop 3D-ARCH*, Mestre-Venice, Italy, Aug. 2005.

R. Roberts and F. Dellaert. Optical flow templates for superpixel labeling in autonomous robot navigation. In *IROS Workshop on Planning, Perception, and Navigation for Intelligent Vehicles*, Tokyo, Japan, Nov. 2013.

R. Roberts and F. Dellaert. Direct superpixel labeling for mobile robot navigation using learned general optical flow templates. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1032–1037, Chicago, IL, USA, Sept. 2014.

R. Rosales and S. Sclaroff. 3D trajectory recovery for tracking multiple objects and trajectory guided recognition of actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 117–123, Fort Collins, CO, USA, June 1999.

P. J. Rousseeuw and C. Croux. Alternatives to the median absolute deviation. *Journal of the American Statistical association*, 88(424):1273–1283, 1993.

R. B. Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany, Oct. 2009.

J. Ryde and H. Hu. 3d mapping with multi-resolution occupied voxel lists. *Autonomous Robots*, 28(2):169–185, 2010.

A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Proceedings of the Robotics: Science & Systems Conference*, Seattle, WA, June 2009.

T. Senlet and A. Elgammal. A framework for global vehicle localization using stereo images and satellite and road maps. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2034–2041, Barcelona, Spain, Nov. 2011.

G. Silveira, E. Malis, and P. Rives. An efficient direct approach to visual SLAM. *IEEE Transactions on Robotics*, 24(5):969–979, 2008.

E. P. Simoncelli, E. H. Adelson, and D. J. Heeger. Probability distributions of optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 310–315, Maui, HI, USA, June 1991.

R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. *Autonomous Robot Vehicles*, 1:167–193, 1990.

A. Stewart and P. Newman. LAPS — localisation using appearance of prior structure: 6-dof monocular camera localisation using prior pointclouds. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2625–2632, Saint Paul, MN, USA, May 2012.

H. Strasdat, J. M. Montiel, and A. J. Davison. Visual SLAM: Why filter? *Image and Vision Computing*, 30(2):65–77, 2012.

C. Studholme, D. L. Hill, and D. J. Hawkes. An overlap invariant entropy measure of 3d medical image alignment. *Pattern Recognition*, 32(1):71–86, 1999.

P. Sturgess, K. Alahari, L. Ladicky, and P. Torr. Combining appearance and structure from motion features for road scene understanding. In *Proceedings of the British Machine Vision Conference*, pages 62.1–62.11, London, United Kingdom, Sept. 2009.

Z. Taylor, J. I. Nieto, and D. Johnson. Automatic calibration of multi-modal sensor systems using a gradient orientation measure. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1293–1300, Tokyo, Japan, Nov. 2013.

S. Thrun. What we're driving at. *Google Official Blog*, 2010. URL http://googleblog.blogspot.com/2010/10/what-were-driving-at.html.

S. Thrun and M. Montemerlo. The graph SLAM algorithm with applications to large-scale mapping of urban structures. *International Journal of Robotics Research*, 25(5-6):403–429, May 2006.

S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1):99–141, 2001.

S. Thrun, M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto, and E. Nebot. FastSLAM: An efficient solution to the simultaneous localization and mapping problem with unknown data association. *J. Machine Learning Res.*, 4(3):380–407, 2004.

J. Tighe and S. Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. In *Proceedings of the European Conference on Computer Vision*, pages 352–365. Springer, Crete, Greece, Sept. 2010.

R. Triebel, P. Pfaff, and W. Burgard. Multi-level surface maps for outdoor terrain mapping and loop closing. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2276–2282, Beijing, China, Oct. 2006.

C. Ulaş and H. Temelta. 3d multi-layered normal distribution transform for fast and long range scan matching. *Journal of Intelligent and Robotics Systems*, 71(1):85–108, 2013.

I. Ulrich and I. Nourbakhsh. Appearance-based obstacle detection with monocular color vision. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 866–871, Austin, TX, USA, July 2000.

C. Urmson. How a driverless car sees the road. TED Talks, March 2015. URL http://www.ted.com/talks/chris_urmson_how_a_driverless_car_sees_the_road.

A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13(2):260–269, April 1967.

A. Wedel, U. Franke, J. Klappstein, T. Brox, and D. Cremers. Realtime depth estimation and obstacle detection from monocular video. In *DAGM-Symposium*, volume 4174 of *Lecture Notes in Computer Science*, pages 475–484. Springer, 2006.

A. Wedel, T. Schoenemann, T. Brox, and D. Cremers. Warpcut–fast obstacle segmentation in monocular video. In *Pattern Recognition*, pages 264–273. Springer Berlin Heidelberg, 2007.

R. W. Wolcott and R. M. Eustice. Visual localization within LIDAR maps for automated urban driving. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 176–183, Chicago, IL, USA, Sept. 2014.

R. W. Wolcott and R. M. Eustice. Fast LIDAR localization using multiresolution Gaussian mixture maps. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2814–2821, Seattle, WA, USA, May 2015.

R. W. Wolcott and R. M. Eustice. Probabilistic obstacle partitioning of monocular video for autonomous vehicles. In *Proceedings of the British Machine Vision Conference*, York, UK, Sept. 2016. Submitted, under review.

World Health Organization. *WHO global status report on road safety 2013: Supporting a decade of action*. World Health Organization, 2013.

T. Wu and A. Ranganathan. Vehicle localization using road markings. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 1185–1190, Gold Coast, Australia, June 2013.

K. Yamaguchi, T. Kato, and Y. Ninomiya. Moving obstacle detection using monocular vision. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 288–293, Tokyo, Japan, June 2006.

J. Yao, S. Ramalingam, Y. Taguchi, Y. Miki, and R. Urtasun. Estimating drivable collision-free space from monocular video. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 420–427, Waikoloa Beach, HI, USA, Jan. 2015.

Y. Zhang, S. J. Kiselewich, W. A. Bauson, and R. Hammoud. Robust moving object detection at distance in the visible spectrum and beyond using a moving camera. In *Computer Vision and Pattern Recognition Workshop*, page 131, New York, NY, USA, June 2006.