2015 IEEE International Conference on Robotics and Automation (ICRA)
Washington State Convention Center
Seattle, Washington, May 26-30, 2015

Fast LIDAR Localization using Multiresolution Gaussian Mixture Maps

Ryan W. Wolcott and Ryan M. Eustice

Abstract— **This paper reports on a fast multiresolution scan matcher for vehicle localization in urban environments for self-driving cars. State-of-the-art approaches to vehicle localization rely on observing road surface reflectivity with a three-dimensional (3D) light detection and ranging (LIDAR) scanner to achieve centimeter-level accuracy. However, these approaches can often fail when faced with adverse weather conditions that obscure the view of the road paint (e.g., puddles and snowdrifts) or poor road surface texture. We propose a new scan matching algorithm that leverages Gaussian mixture maps to exploit the structure in the environment; these maps are a collection of Gaussian mixtures over the z-height distribution. We achieve real-time performance by developing a novel branch-and-bound, multiresolution approach that makes use of rasterized lookup tables of these Gaussian mixtures. Results are shown on two datasets that are 3.0 km: a standard trajectory and another under adverse weather conditions.**

I. 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. While several groups have attempted to transition to vision-only solutions for autonomous vehicles because of cost and visual appearance [1]–[4], manufacturers continue to lower the price and increase the aesthetic appeal of 3D LIDAR scanners [5]. Further, production automated vehicles will need to consider multi-modality methods that will yield a more robust solution.

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 [6], [7] 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, however, can fail when there is not sufficient observable road paint or in harsh weather conditions that result in partially occluded roadways. In this paper, we seek a fast, globally optimal scan matcher that allows us to quickly localize a vehicle by exploiting the 3D structure of the scene as opposed to ground-plane reflectivities.

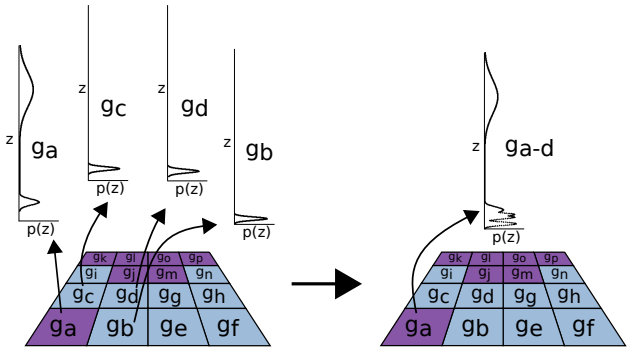


Fig. 1: Overview of our proposed LIDAR localization scheme. We propose to use Gaussian mixture maps: a 2D grid over xy where each cell in the grid holds a one-dimensional Gaussian mixture model that accurately models the distribution over that cell's z-height. We then perform registration in these maps by formulating a branch-and-bound search over multiresolution, rasterized versions of the Gaussian mixture maps where coarser resolutions capture an upper-bound over the finer resolutions. This methodology finds the guaranteed optimal registration over a user-specified search space.

We propose to leverage a Gaussian mixture map, which is a 2D grid structure where each grid cell maintains a Gaussian mixture model characterizing the distribution over z-height (i.e., vertical structure) in that cell. Furthermore, 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. 1 for a sample of these maps. The key contributions of our paper are:

- Data reduction of large point clouds to a compact mixture of Gaussians.
- Online rasterization of these parametric maps that enables fast inference.
- Branch-and-bound registration formulation that allows real-time, guaranteed-optimal registration, using generic upper-bound rasterizations.

II. 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 et al. [6] initially proposed using a 3D LIDAR scanner to observe the ground-plane reflec-

*This work was supported by a grant from Ford Motor Company via the Ford-UM Alliance under award N015392; R. Wolcott was supported by The SMART Scholarship for Service Program by the Department of Defense.

R. Wolcott is with the Computer Science and Engineering Division, University of Michigan, Ann Arbor, MI 48109, USA rwolcott@umich.edu.

R. Eustice is with the Department of Naval Architecture & Marine Engineering, University of Michigan, Ann Arbor, MI 48109, USA eustice@umich.edu.

tivities, 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 [8] employed a similar approach, by using a two-dimensional (2D) LIDAR scanner to build 3D swathes as the vehicle traversed the environment. In previous work, we demonstrated that ground-plane reflectivities can also be used to localize a monocular camera in a 3D LIDAR reflectivity map [4].

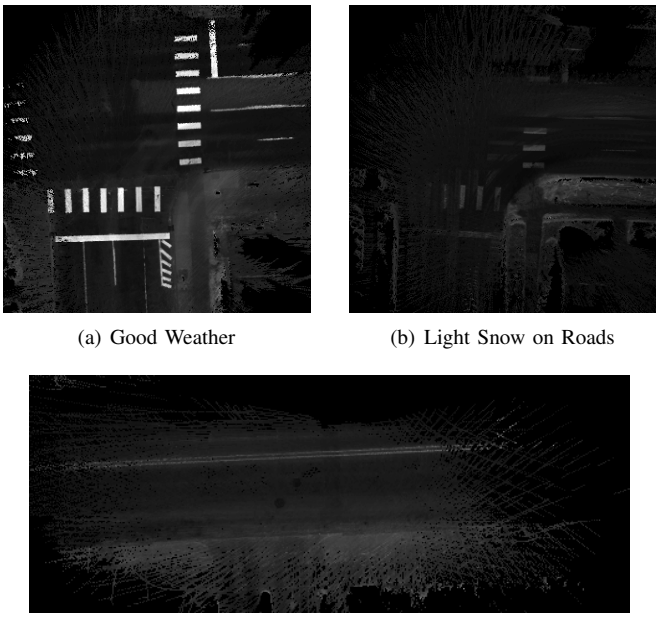
Despite attempts by Levinson et al. in [7] to model slight changes in appearance of these ground plane maps, all of these methods can fail when harsh weather is present in the environment—for example, rain puddles and snowdrifts can build up and occlude the view of the informative ground signal, see Fig. 2. Additionally, long two-lane roads with a double lane-marker between them can allow longitudinal uncertainty to grow unbounded due to lack of texture in the longitudinal direction. Thus, to increase robustness to these types of scenarios, we are interested in 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) [9], generalized iterative closest point (GICP) [10], normal distributions transform (NDT) [11], or a similar variant 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 [12] and Aghamohammadi et al. [13] presented different features that can be extracted and matched from raw point cloud points. Pandey et al. [14] 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. [15], Kümmerle et al. [16], and Maier et al. [17] 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 [18], NDT [19] [20], and occupied voxel lists [21]. These searches use heuristics to greedily guide the coarse-to-fine steps that yield good results in practice, but still cannot guarantee global optimality.



(a) Good Weather

(b) Light Snow on Roads

(c) Poor Texture in Road

Fig. 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.

We employ techniques presented by Olson [22] 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 [22] to handle full-3D point clouds by creating efficient Gaussian mixture maps for fast inference.

III. Gaussian Mixture Maps

The key challenge to enabling fast localization is developing 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 over the z-height distribution. This 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, as seen in Fig. 1.

This representation is quite similar to NDT maps [11] in the sense that both representations 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. Our representation is also similar to multi-level surface (MLS)

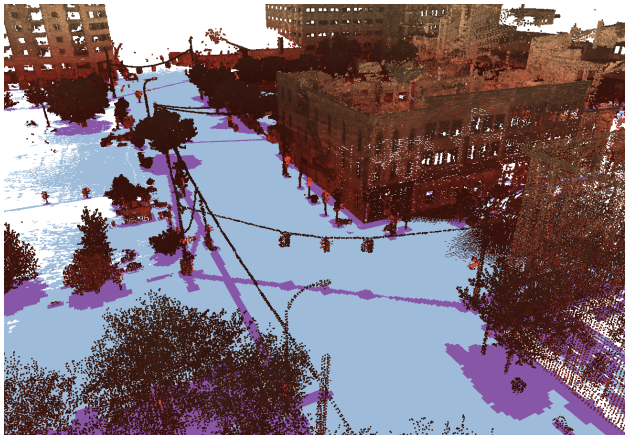


Fig. 3: Here we demonstrate the efficacy of our point cloud reduction to a Gaussian mixture map. Colored from brown-yellow is a rendering of the point cloud that we accumulate to build our Gaussian mixture maps. On the ground plane we rendered the Gaussian mixture map, where color encodes the number of Gaussians in each cell's Gaussian mixture (*white*, *blue*, and *purple* corresponds to 0, 1, and 2, respectively). Our GMM is able to compactly parameterize both ground-plane and superstructure to a pair of Gaussians in each cell.

maps [23], which cluster the point cloud into *horizontal* and *vertical* structure components. 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.

We build these maps offline as a full SLAM problem. In doing so, we integrate odometry, GPS, and scan-matching constraints to build a self-consistent pose-graph. This pipeline is identical to our previous work [4], with the exception that we no longer apply artificial height priors—this allows our graph to be built in full-3D in order to construct a more accurate representation.

With the optimized pose-graph, we then reproject all of our 3D LIDAR scan points to create a point cloud. Then at a fixed grid resolution (we used 20 cm throughout), we look at each "column" of z-points. In order to capture the variance of our LIDAR scanner and reduce discretization errors, we blur these z-points with neighboring "columns" with a Gaussian kernel. We then fit a weighted Gaussian mixture model to these points using EM. The entire offline map building process can be completed in approximately 1 hour on our datasets.

For each cell, we also re-run the EM for a varying number of Gaussians, choosing the resulting Gaussian mixture with the greatest likelihood without overfitting. In our experiments, we empirically determined that limiting the number of Gaussians to two is sufficient for capturing the ground surface, building facades, and overhangs; though this threshold can be application dependent on expected environment. This resulting grid of Gaussian mixtures makes up our Gaussian mixture map, $\mathcal{G}$.

The efficacy of our Gaussian mixture map in modeling the distribution of the world can be seen in Fig. 3. In this figure, we show our input point cloud that is constructed from our pose-graph optimization and the reduction of this point cloud to a 2D collection of Gaussian mixtures (our actual Gaussian mixture map).

Algorithm 1 Full Registration

Input: GMM $\mathcal{G}$, Point Cloud $\mathcal{C}$, guess (x, y, z, r, p, h), search range X, Y, H
Output: Optimal registration = (x, y, z, r, p, h)*
1: $(\hat{x}, \hat{y}, z, r, p, \hat{h}) = \texttt{SEARCH}(x, y, z, r, p, h)$
2: $(x, y, z, r, p, h)^* = \texttt{HILL-CLIMB}\ (\hat{x}, \hat{y}, z, r, p, \hat{h})$

Inference in our Gaussian mixture map is easily done. With our Gaussian mixture map, $\mathcal{G}$, and an online point cloud scan, $\mathcal{C}$, we can probabilistically evaluate the likelihood of a given transformation, T, that brings the two into alignment. We evaluate this likelihood by applying the transformation to each point in the point cloud, $\mathcal{C}' = T\mathcal{C}$. Then, for each point in this transformed cloud, $p_i = (x_i, y_i, z_i)$, we index into our Gaussian mixture map to obtain that cell's Gaussian mixture, $g_i \leftarrow \mathcal{G}(x_i, y_i)$. From there, we treat each point as an independent Gaussian observation, allowing us to compute the total log-likelihood by summing each point's log-likelihood,

$$\texttt{LL} = \sum_i log\left(\sum_j \frac{w_{ij}}{\sqrt{2\pi {\sigma_{ij}}^2}} \exp\left(-\frac{(z_i - \mu_{ij})^2}{{\sigma_{ij}}^2}\right)\right), \tag{1}$$

where w_{ij}, μ_{ij}, and σ_{ij} are the weight, mean, and standard deviation, respectively, of the j^{th} component of g_i.

A. Registration Formulation

Given a point cloud, we seek to find the optimal transformation that maximizes Eq. 1.

We make the observation that a typical wheeled-robotic platform will be fairly 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, we can tailor our search strategy according to this by exhaustively searching over a range of x, y, and $heading$ transformations. As in [22], we can efficiently compute these by applying the $heading$ rotation to all points *first*, then evaluating xy translations.

With our solution within the near vicinity of the optimum, we then perform a simple, constrained 6-DOF hill-climbing to lock into the global optimum over our search space: (x, y, z, r, p, h)*. 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

Algorithm 2 Exhaustive Search

Input: GMM $\mathcal{G}$, Point Cloud $\mathcal{C}$, guess (x, y, z, r, p, h), search range X, Y, H
Output: Best registration = $(\hat{x}, \hat{y}, \hat{h})$

$best = -\infty$
for h_i in H **do**
  apply rotation h_i to $\mathcal{C}$
  for x_i, y_i in XY **do**
    $likelihood = LL(x_i, y_i)$ ▷ Eq. 1
    if $likelihood > best$ **then**
      $best = likelihood$
      $(\hat{x}, \hat{y}, \hat{h}) = (x_i, y_i, h_i)$
    end if
  end for
end for

into our Gaussian mixture map can be found in Algorithm 1 and Algorithm 2.

IV. Multiresolution Branch-and-Bound

In this section, we replace the extremely expensive exhaustive search with an efficient multiresolution branch-and-bound search.

A. 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 can upper-bound 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.

For evaluating a single transformation (i.e., (x_i, y_i)), you 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 $(x_i, y_i) \rightarrow (x_{i+N}, y_{i+N})$, each point is evaluated against a total of $(N+1)^2$ Gaussian mixtures. However, each cell in our map is quite spatially similar, meaning that inference into (x_i, y_i) yields a similar log-likelihood as (x_{i+1}, y_i), so the exhaustive search will often spend unnecessary time in low-likelihood regions.

We formulate a branch-and-bound search that exhaustively searches over our coarsest resolution providing upper-bounds over a range of transformations. These coarse search results are then added to a priority queue, ranked by these 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. 4) to better compute tight upper-bounds. This optimization makes the trade off for better bounds as opposed to a smaller memory footprint.

Algorithm 3 Multiresolution Search

Input: Multires-GMM $\mathcal{G}$, Point Cloud $\mathcal{C}$, guess (x, y, z, r, p, h), search range X, Y, H
Output: Best registration = $(\hat{x}, \hat{y}, \hat{h})$

// initialize priority queue with search over coarsest resolution
Initialize $PriorityQueue$ ▷ priority = log-likelihood
$coarsest = N$
$\mathcal{RC} = empty$ ▷ rotated point clouds
for h_i in $h + H$ **do**
  // store rotated clouds — allows to do transformations once
  $T = f(0, 0, z, r, p, h_i)$ ▷ $[x, y]$ applied later
  $\mathcal{RC}[h_i] = T * \mathcal{C}$
  for x_i in $x + X/2^{coarsest}$ **do**
    for y_i in $y + Y/2^{coarsest}$ **do**
      $cur.res = coarsest$
      $cur.[x, y, h] = [x_i, y_i, h_i]$
      $cur.LL = LL(\mathcal{G}[coarsest], \mathcal{RC}[h_i]), x_i, y_i)$
      $PriorityQueue.add(cur)$
    end for
  end for
end for
// iterate priority queue, branching into finer resolutions
while $prev = PriorityQueue.pop()$ **do**
  if $prev.res == 0$ **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 $\left[prev.x, prev.x + 2^{prev.res-1}\right]$ **do**
    for y_i in $\left[prev.y, prev.y + 2^{prev.res-1}\right]$ **do**
      $cur.res = prev.res - 1$
      $cur.[x, y, h] = [x_i, y_i, h_i]$
      $cur.LL = LL(\mathcal{G}[cur.res], \mathcal{RC}[prev.h]), x_i, y_i)$
      $PriorityQueue.add(cur)$
    end for
  end for
end while

Because our maps are the same resolution throughout each multiresolution layer, this results in us taking larger *strides* through the coarser resolutions, where $stride = 2^{res}$. Branching factor and number of multiresolution maps is completely user-defined. In our experiments, we opted for a branching factor of 2 ((x_i, y_i) : $[(x_i, y_i), (x_i, y_{i+2^{\text{res}}}), (x_{i+2^{\text{res}}}, y_i), (x_{i+2^{\text{res}}}, y_{i+2^{\text{res}}})]$) to limit the amount of unnecessary work.

Refer to Algorithm 3 and Fig. 4 for a more detailed overview.

B. Rasterized Gaussian Mixture Maps

Here, we define our bounding function for our multiresolution search.

Finding good, 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. At the finest resolution, we replace our parametric Gaussian mixture map with a rasterized version by evaluating the log-

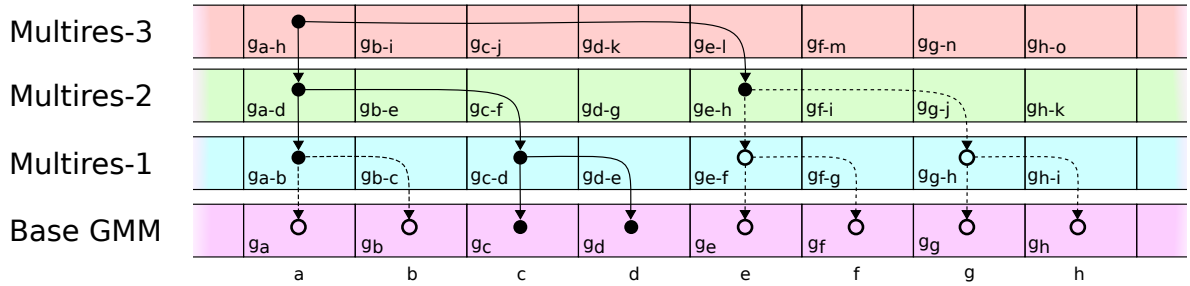


Fig. 4: A one-dimensional example of our multiresolution search formulation, where we demonstrate how a *single* point cloud point would traverse through the multiresolution tree. Given some knowledge that the best transformation aligns the point somewhere within a-h, we begin the search at the coarsest 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 $g_{a\text{-}h}$ refers to the fact that inference in that cell is an upper-bound over the distributions $g_a - g_h$, 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.

likelihood at a fixed discretization, generating a rasterization for each grid cell. Upper bounds can then be *exactly* computed by taking the `max` across each discretization in the rasterized lookup table. See Fig. 5 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. This allows us to store $1\ \mathrm{km}$ of urban maps using less than $30\ \mathrm{MB}$ of disk space, where our maps have no more than *two* Gaussians per $20\ \mathrm{cm}$ grid cell. We are then able to efficiently compute rasterized 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 be used with many map types including standard NDTs, MLS maps, occupancy voxels, etc. After converting one of these arbitrary maps to a rasterized multiresolution map, the remainder of our proposed pipeline can be used for fast registration of a point cloud. The pipeline can also be adapted to the probabilistic intensity maps of Levinson et al. [7].

V. Results

We evaluated our algorithm through data collected on our autonomous platform, a TORC ByWire XGV. This automated vehicle is equipped with four Velodyne HDL-32E 3D LIDAR scanners and an Applanix POS-LV 420 inertial navigation system (INS), as can be seen in Fig. 6. All experiments were run on a laptop equipped with a Core i7-3820QM central processing unit (CPU).

Experiments are presented on two primary datasets:

- *Downtown*: $3.0\ \mathrm{km}$ trajectory through downtown Ann Arbor, Michigan in which multiple roads are traversed from both directions and the dataset contains several dynamic obstacles.
- *Downtown Snowy*: Same trajectory as *Downtown* on a snowy day with snow actively falling and covering the ground, as depicted in Fig. 7.

Fig. 6: Test platform used for experimental results. This platform is a TORC ByWire XGV equipped with 4 Velodyne HDL-32E LIDAR scanners and an Applanix POS-LV 420 INS.

We also performed an additional pass through downtown Ann Arbor to construct our Gaussian mixture map. All three of these datasets were aligned using our offline SLAM procedure, providing us with sufficiently accurate ground-truth for our experiments (ground-truth accuracy an order of magnitude greater than our localization errors).

A. Multiresolution Registration Results

For a set of scans in our datasets, we evaluated our multiresolution registration by randomly sampling within $10\ \mathrm{m}$ of the ground-truth pose. By performing a search around these randomly sampled points, we expect to see that our algorithm is able to return the scan to the ground-truth estimate; quantifying our registration error by L^2 distance.

We present these results in two ways. First, we compiled the results into a histogram, as shown in the top row of Fig. 8. Here we see that our proposed solution is able to return to within $0.5\ \mathrm{m}$ 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 our results are similar across both datasets. Further, despite the significant amount of falling snow, as shown in Fig. 7, our method is still robust.

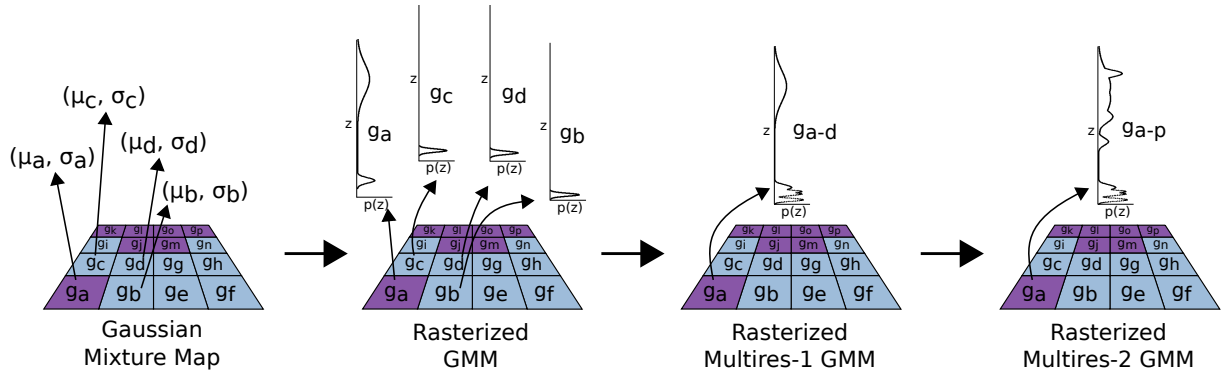


Fig. 5: Demonstration of the rasterization performed on the original Gaussian mixture map to facilitate exact upper-bounds. We begin with a parametric 2D map that encodes a Gaussian mixture over z-height in each cell, which we then rasterize for each cell (note we display the likelihood, not log-likelihood 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.

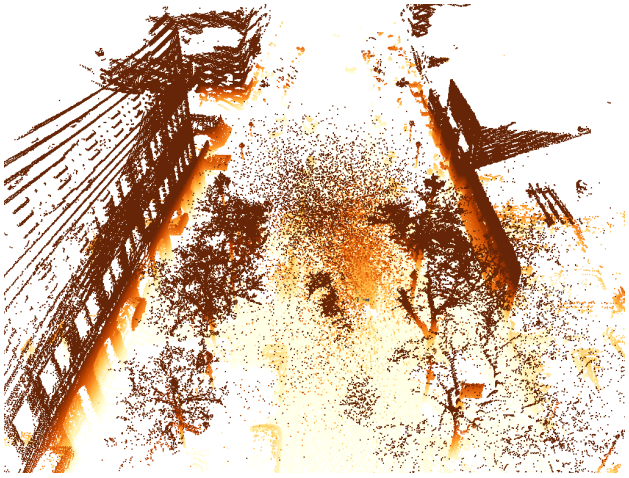


Fig. 7: Point cloud rendering of typical snowfall observed during the *Downtown Snowy* dataset.

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. 8. 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 perfect transformation.

A sample search through our multiresolution search space can be seen in Fig. 9. The shown example explores a $25\ \mathrm{m} \times 25\ \mathrm{m}$ area in approximately 2 seconds, while only needing to evaluate 1% of the transformations necessary in the exhaustive search.

B. Filtered Results

We integrated our registration algorithm into an extended Kalman filter (EKF) localization framework, which is an extension from [4]. The only measurements used were those from our IMU for odometry and our multiresolution scan matches initialized around our 3-σ posterior belief. We use fixed measurement uncertainties when incorporating multiresolution scan matches into our filter; however, one could fit a conservative covariance using the explored search space as in [22].

We compare our localization performance against our own implementation of the state-of-the-art reflectivity-based localization proposed by Levinson et al. in [6], [7]. Our reflectivity-based localization 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.

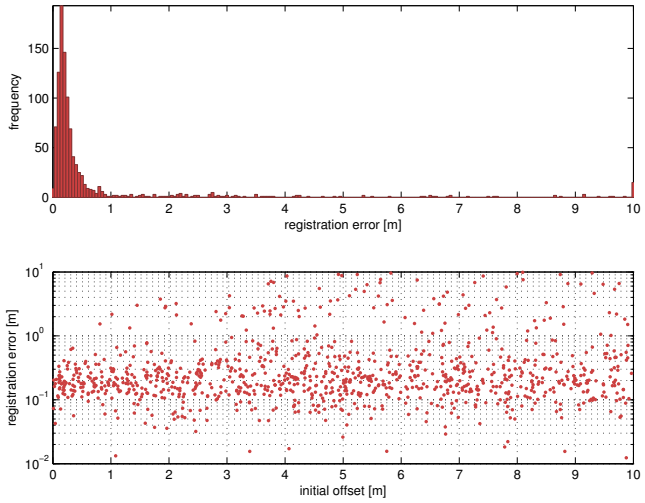
Due to the significant amount of snow on the ground during the *Downtown Snowy* dataset, we also had to incorporate GPS measurements into our reflectivity-based localization system (results denoted with an asterisk). Without these additional measurements, the filter would constantly diverge as orthographic reflectivity matches were rarely successful.

We display the lateral and longitudinal error over time for both the *Downtown* and *Downtown Snowy* datasets in Fig. 10. As can be seen, our proposed solution is able to achieve similar performance on the *Downtown* dataset as the reflectivity-based solution. Moreover, we are able to stay well localized in the *Downtown Snowy* dataset, whereas the reflectivity-based solution consistently diverges, only being able to incorporate occasional measurements into its EKF.

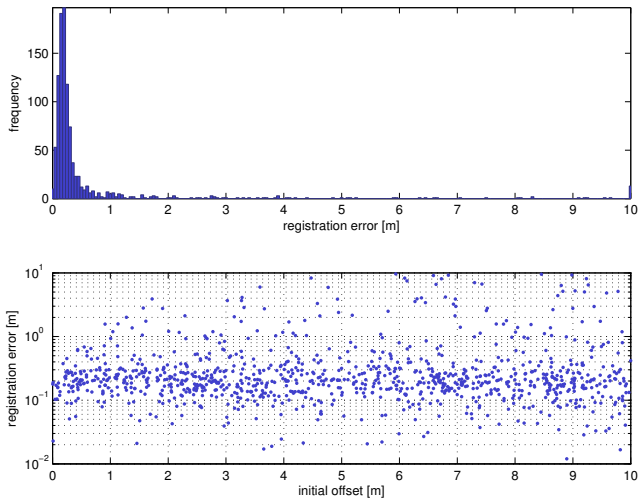
Further, our results are tabulated in Table I. Here we show that we are able to achieve longitudinal and lateral root mean square (RMS) errors of $15.5\ \mathrm{cm}$ and $10.3\ \mathrm{cm}$, respectively, on the *Downtown* dataset. Additionally, we obtain longitudinal and lateral RMS errors of $18.0\ \mathrm{cm}$ and $9.4\ \mathrm{cm}$, respectively, on the *Downtown Snowy* dataset. Our method is able to provide scan matches at approximately 3-4 Hz.

	Downtown **RMS Error**		***Downtown Snowy*** **RMS Error**	
Method	Longitudinal	Lateral	Longitudinal	Lateral
Reflectivity	12.4 cm	8.0 cm	73.4 cm*	62.9 cm*
Proposed	15.5 cm	10.3 cm	18.0 cm	9.4 cm

TABLE I: Comparison of RMS errors for reflectivity-based localization and our proposed 3D structure-based localization.



(a) *Downtown*—Registration Results.



(b) *Downtown Snowy*—Registration Results.

Fig. 8: This figure shows the registration error of our proposed scan-matching algorithm. We generated random initial offsets for our scan matcher around a ground-truth estimate, evaluating how accurately the scan-matcher can return to this ground-truth. 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.

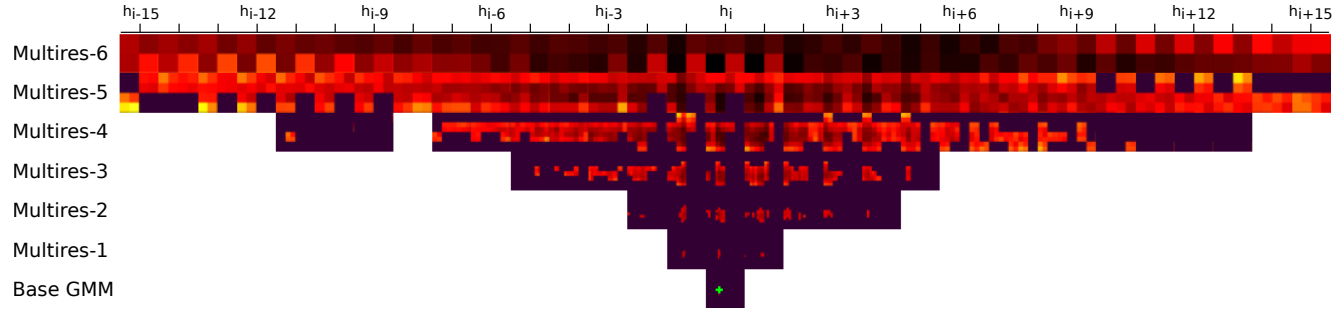


Fig. 9: Sample multiresolution search space traversal. Top-bottom represents coarse-to-fine searching, left-right represents different slices through our $heading$ search, and each pixel depicts an xy translation searched. Log-likehoods are colored increasingly yellow-black, purple and non-existent cells 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.

C. Obstacle Detection

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 not. 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. Thus, 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 all points that exceed this threshold as obstacles. Our formulation allows us to do this classification on a frame-by-frame basis and extend our sensing range of obstacles. A visualization of point cloud classification can be seen in Fig. 11.

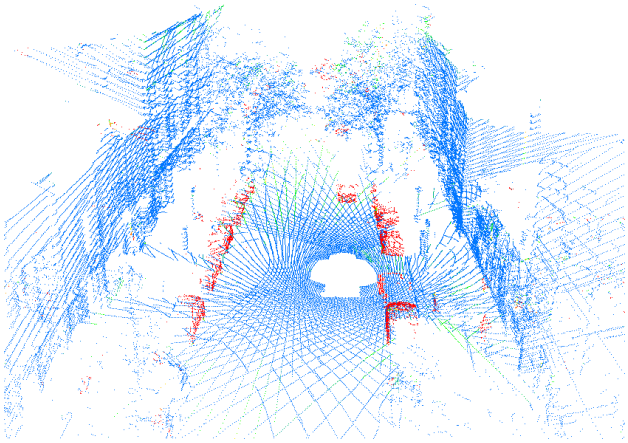
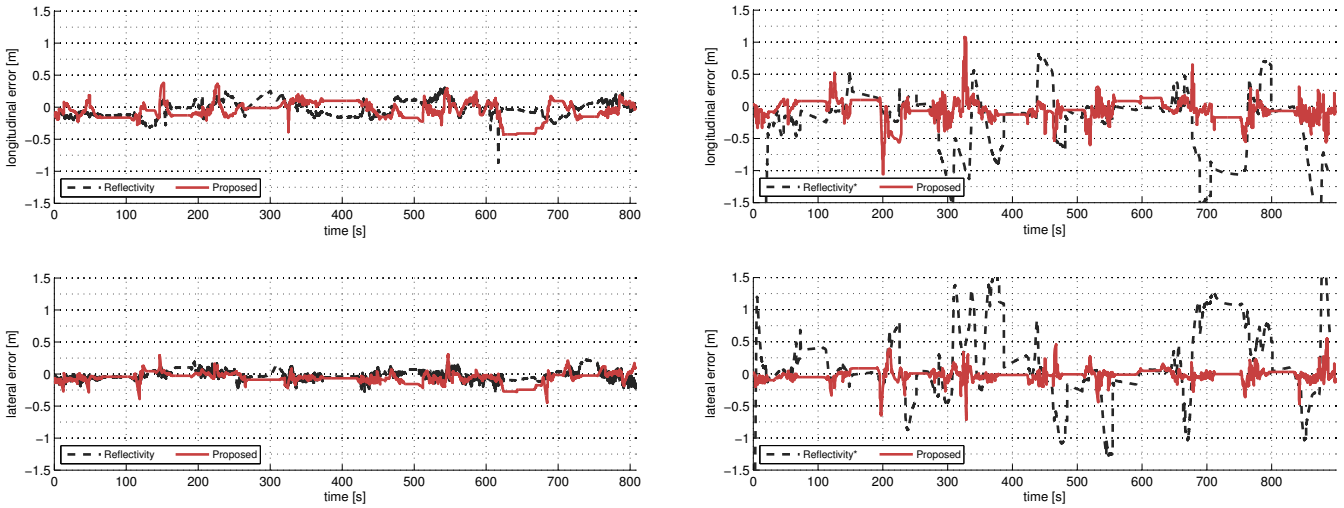


Fig. 11: Sample point cloud colored by Mahalanobis distance from the underlying map's Gaussian mixture. Note the obstacles in red and agreeing prior map in blue. Our method allows us to expand our obstacle sensing horizon, as we can not sense the ground-plane beyond 40 m.

VI. Conclusion

In this paper we demonstrated a new Gaussian mixture map that can be used for rapid registration from an observed



(a) *Downtown*—Filtered Results.

(b) *Downtown Snowy*—Filtered Results.

Fig. 10: Here we present our localization accuracy in terms of longitudinal and lateral error relative to SLAM-optimized ground-truth over time. Our proposed solution is able to overcome lack of ground plane reflectivites by exploiting the structure in the *Downtown Snowy* dataset.

point cloud. Through the use of multiresolution rasterized maps that can be computed online, we can efficiently 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 an autonomous platform can remain well localized in a prior map using these measurements alone. 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.

REFERENCES

[1] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[2] M. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights." in *Proc. IEEE Int. Conf. Robot. and Automation*, Saint Paul, MN, USA, May 2012, pp. 1643–1649.

[3] A. Stewart and P. Newman, "LAPS — localisation using appearance of prior structure: 6-DOF monocular camera localisation using prior pointclouds," in *Proc. IEEE Int. Conf. Robot. and Automation*, Saint Paul, MN, USA, May 2012, pp. 2625–2632.

[4] R. W. Wolcott and R. M. Eustice, "Visual localization within LIDAR maps for automated urban driving," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Chicago, IL, USA, Sept. 2014, pp. 176–183.

[5] A. Davies. This palm-sized laser could make self-driving cars way cheaper. *Wired*, 25 September 2014. [Online]. Available: http://www.wired.com/2014/09/velodyne-lidar-self-driving-cars/

[6] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments." in *Proc. Robot.: Sci. & Syst. Conf.*, Atlanta, GA, June 2007.

[7] J. Levinson and S. Thrun, "Robust vehicle localization in urban environments using probabilistic maps," in *Proc. IEEE Int. Conf. Robot. and Automation*, Anchorage, AK, May 2010, pp. 4372–4378.

[8] I. Baldwin and P. Newman, "Road vehicle localization with 2d push-broom lidar and 3d priors," in *Proc. IEEE Int. Conf. Robot. and Automation*, Saint Paul, MN, USA, May 2012, pp. 2611–2617.

[9] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992.

[10] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Proc. Robot.: Sci. & Syst. Conf.*, Seattle, WA, June 2009.

[11] M. Magnusson, "The three-dimensional normal-distributions transform—an efficient representation for registration, surface analysis, and loop detection," Ph.D. dissertation, Örebro University, Dec. 2009, Örebro Studies in Technology 36.

[12] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments," Ph.D. dissertation, Computer Science department, Technische Universitaet Muenchen, Germany, Oct. 2009.

[13] 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 *Proc. European Conf. on Mobile Robots*, Freiburg, Germany, Sept. 2007.

[14] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Visually bootstrapped generalized ICP," in *Proc. IEEE Int. Conf. Robot. and Automation*, Shanghai, China, May 2011, pp. 2660–2667.

[15] 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 *Proc. IEEE Int. Conf. Robot. and Automation*, Karlsruhe, Germany, May 2013, pp. 1554–1559.

[16] R. Kümmerle, R. Triebel, P. Pfaff, and W. Burgard, "Monte carlo localization in outdoor terrains using multilevel surface maps," *Journal of Field Robotics (JFR)*, vol. 25, pp. 346–359, June - July 2008.

[17] D. Maier, A. Hornung, and M. Bennewitz, "Real-time navigation in 3D environments based on depth camera data," in *Proc. of the IEEE/RAS Int. Conf. Humanoid Robots*, Osaka, Japan, Nov. 2012, pp. 692–697.

[18] S. Granger and X. Pennec, "Multi-scale EM-ICP: A fast and robust approach for surface registration," in *Proc. European Conf. Comput. Vis.*, Copenhagen, Denmark, May 2002, pp. 418–432.

[19] C. Ulaş and H. Temelta, "3d multi-layered normal distribution transform for fast and long range scan matching," *J. Intell. and Robotic Syst.*, vol. 71, no. 1, pp. 85–108, 2013.

[20] N. Ripperda and C. Brenner, "Marker-free registration of terrestrial laser scans using the normal distribution transform," vol. 4, Mestre-Venice, Italy, August 2005.

[21] J. Ryde and H. Hu, "3d mapping with multi-resolution occupied voxel lists." *Auton. Robots*, vol. 28, no. 2, pp. 169–185, 2010.

[22] E. Olson, "Real-time correlative scan matching," in *Proc. IEEE Int. Conf. Robot. and Automation*, Kobe, Japan, June 2009, pp. 4387–4393.

[23] R. Triebel, P. Pfaff, and W. Burgard, "Multi-level surface maps for outdoor terrain mapping and loop closing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Beijing, China, Oct. 2006, pp. 2276–2282.