Fast Monte-Carlo Localization on Aerial Vehicles using Approximate Continuous Belief Representations

Aditya Dhawale* Kumar Shaurya Shankar* Nathan Michael

Abstract

Size, weight, and power constrained platforms impose constraints on computational resources that introduce unique challenges in implementing localization algorithms. We present a framework to perform fast localization on such platforms enabled by the compressive capabilities of Gaussian Mixture Model representations of point cloud data. Given raw structural data from a depth sensor and pitch and roll estimates from an on-board attitude reference system, a multi-hypothesis particle filter localizes the vehicle by exploiting the likelihood of the data originating from the mixture model. We demonstrate analysis of this likelihood in the vicinity of the ground truth pose and detail its utilization in a particle filter-based vehicle localization strategy, and later present results of real-time implementations on a desktop system and an off-the-shelf embedded platform that outperform localization results from running a state-of-the-art algorithm on the same environment.

1. Introduction

For an agent lost in a known environment, much of the cost of localization can be offset by precomputing measures of what the sensor is expected to see; localization can then be cast as the much simpler problem of a search through these pre-existing "hallucinated" views. However, exhaustively considering all possible views incurs a prohibitive cost that increases exponentially with both the dimensionality of the state space and the size of the environment. Further, naïve pre-rendering approaches can be susceptible to errors caused by perceptual aliasing due to slight variations in the environment appearance or by regions that are not feature rich, such as blank walls [1].

In this paper we present a framework enabling rapid computation of sensor data likelihood via an environment representation that is both high-fidelity and memory efficient. The framework models a depth camera observation as being sampled from the environment representation with a likelihood measure that varies smoothly about the true camera pose. We thus exploit the dramatically reduced storage complexity of the representation and the local spatial regularity of a fast-to-compute likelihood function to re-cast the pose estimation problem as that of Monte-Carlo localization [25].

Similar in spirit to our choice of representation, [8] map the world with a succinct, albeit restrictive parameterization of using only dominant planes. Our choice of representation however, does not have any such restrictions. Likewise, [2] is similar in its restriction of representation to a structured world consisting of planes and edges and only localizes in 2D. Approaches such as NDT occupancy maps [5, 16, 23], unlike the representation used in this work, learn independent distributions over each discretized cell leading to a lower fidelity representation at the cell boundaries [6, 21]. Further, for fast pointcloud alignment they require precomputation of the incoming data for comparison with a stored model, known as Distribution-to-Distribution (D2D) registration. The more accurate Point-to-Distribution (P2D) registration approaches, however, are not as real-time viable [5, 14], and are less so for our purpose. In [4] the authors represent each point in the reference scan of a 2D sonar scan as an isotropic Gaussian distribution and iteratively compute the 2D transformation that maximizes the likelihood of all points in the target scan. Our approach, on the other hand, represents clusters of points in the reference scan as individual anisotropic Gaussian components and is not restricted to transformations of small magnitude.

Closest in terms of our choice of implementation framework, [9] propose a particle filter based real-time RGB-D pose estimation approach and [3] present a real-time localization approach on a fixed-winged aircraft during aggressive flights with a laser scanner and an IMU. The latter work implicitly exploits much more restricted dynamics of a fixed wing aircraft within its process model and further, both these approaches use the OctoMap [12] representation to provide correction updates within their filter estimates. Note that the memory footprint of an OctoMap is much greater than that of our choice of representation. For instance, memory consumption for an OctoMap of dataset D3 (Sec. 4.1) with 0.1 m resolution is 872 KB while the

*Contributed equally to this work. Authors are with The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 15213, USA. {adityand, kshaurya, nmichael}@cmu.edu

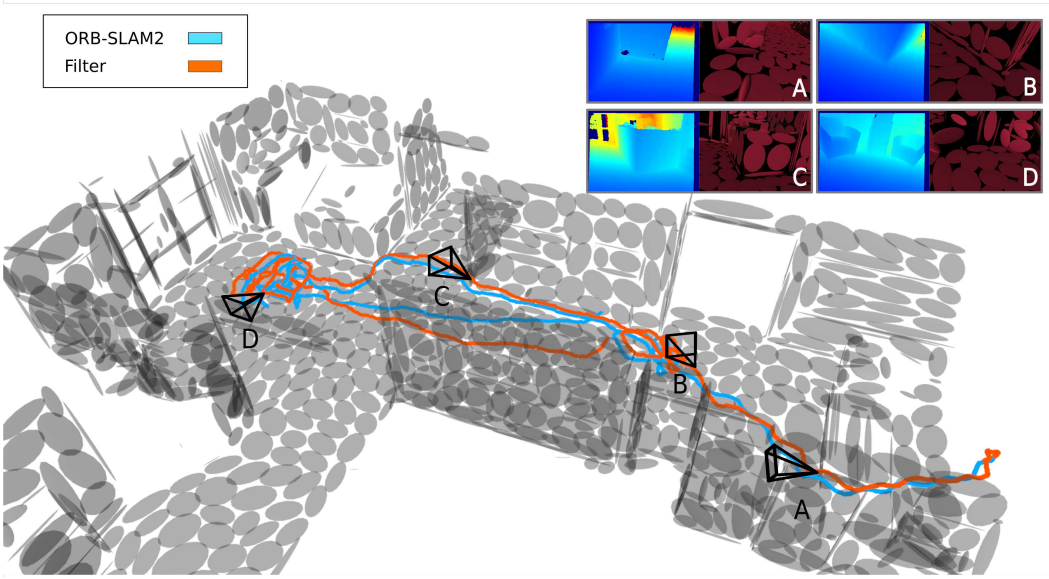

Figure 1. Comparison of the mean particle filter pose (orange) with that of the integrated process model trajectory (cyan) from a representative (office) dataset. The filter estimate is initialized with a uniform distribution away from the true location of the vehicle. As the camera observes more informative data the filter quickly converges to the correct pose. *Top Right*: Four views of the raw point cloud sensor data and the corresponding view of the GMM map from the mean of the particle filter estimates. The GMM components are superimposed on top of the source point cloud with their $3\boldsymbol{\Sigma}$ bounds visualized as gray ellipsoids.

corresponding data usage for our Gaussian Mixture Model (GMM) map representation consisting of 1000 Gaussian components is 40 KB.

Prior works exploiting known map appearance for precise monocular pose estimation [10,17,19,22] employ a textured depth map within an iterative optimization framework to compute the warp that minimizes a photometric cost function between a rendered image and the live image such as the Normalized Information Distance [19], that is robust to illumination change, or a Sum of Squared Differences cost function with an affine illumination model to tackle illumination change [17]. Both algorithms rely on initialization for tracking via a GPS prior or an ORB-based bag-of-words approach, respectively, and expensive ray casted dense textured data for refinement. Note that in contrast to the above mentioned algorithms that use RGB information, we only use depth observations and only project a finite number of mixture components as opposed to dense pre-rendered views of the map.

Our framework solves the problem of 6 Degree-of-Freedom pose estimation for Size, Weight, and Power (SWaP) constrained micro air vehicles operating in a known dense 3D pointcloud environment with an onboard monocular depth camera and Inertial Measurement Unit (IMU). We assume that the vehicle pitch and roll are obtained from an attitude estimation algorithm using the IMU in order to constrain the search space to just heading and position. Our main contributions are:

- A particle filter-based localization strategy based on a high fidelity, memory efficient environment representation enabled by a fast likelihood computation approximation; and
- Experimental evaluation of the approach on a desktop and an off-the-shelf mobile GPU system.

2. Approach

Contemporary direct tracking algorithms require the projection of a large number of dense or semi-dense points into image space to align the current sensor data to a reference model. In contrast, we employ GMMs as a succinct parameterized representation to achieve orders of magnitude computational savings via an analytic projection into image space. The consequent reduction in complexity enables projection in multiple pose hypotheses concurrently in real-time and motivates this work.

This section details the choice of environment representation and how it enables the proposed real-time sensor data likelihood computation.

2.1. Spatial GMMs as an Environment Representation for Tracking

Conventional means of representing maps such as voxel grids discretize space to encode occupancy leading to resolution dependent model fidelity and memory efficiency. An alternate approach is to represent occupancy using a GMM map that attempts to approximate the underlying distribution from which sensor measurements are sampled.

This formulation is capable of representing the environment model with as high a fidelity as required that scales gracefully with model complexity when used in a hierarchical fashion [21]. Additionally, this representation provides a probabilistic uncertainty estimate of the occupancy at any sampled location. Fitting these models to data in real time is possible due to recent advances that enable efficient operation [7]. We utilize the contribution presented in [7, 21] to inform the number of Gaussian components required to pre-compute GMM maps of the environment point cloud at various fidelity levels. For the purpose of this paper, however, we limit the discussion to using only GMM maps at a certain fidelity level chosen according to Sec. 4.2, but note that the approach can be readily extended to a hierarchical formulation.

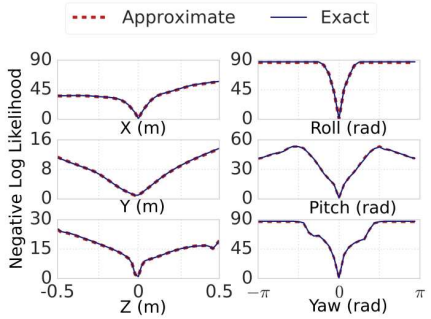

Figure 2. Negative log-likelihood plots of sensor data acquired from camera poses offset from a randomly chosen true pose in dataset D1 by incremental linear and rotational displacements. Utilizing only the relevant components using the approximation discussed in Sec. 3 leads to almost identical likelihoods as when utilizing all the Gaussian components present in the model.

A spatial GMM represents the probability of matter existing at a specific position $\mathbf{P}^w$ given the model component parameters $\mathbf{\Theta} = \{\boldsymbol{\mu}_i, \mathbf{\Sigma}_i, \lambda_i\}_{i=1\ldots M}$ such that

$$p(\mathbf{P}^w; \mathbf{\Theta}) = \sum_i^M \lambda_i \mathcal{N}(\boldsymbol{\mu}_i, \mathbf{\Sigma}_i) \tag{1}$$

where λ_i is the mixture weight, $\boldsymbol{\mu}_i$ the mean 3D position, and $\mathbf{\Sigma}_i$ the covariance of the i^{th} component of the GMM respectively, with $\sum_i^M \lambda_i = 1$ and $\lambda_i > 0$.

2.2. Projection of a GMM Component into Image Space

In order to determine relevant mixture components of a given spatial GMM map for evaluating sensor data likelihood we first analytically project the mixture components into image space.

For a point $\mathbf{P}^w$ in the world frame, the transformed position $\mathbf{P}^c$ in a camera frame $\mathbf{T}_w^c$ is denoted as

$$\mathbf{P}^c = \mathbf{R}_w^c \mathbf{P}^w + \mathbf{t}_w^c$$

where $\mathbf{R}_w^c$ and $\mathbf{t}_w^c$ are the corresponding rotation matrix and translation vectors, respectively. Since this is a linear operation on $\mathbf{P}^w$, using Eq. 1 the transformed distribution of points in the camera frame for the i^{th} component is

$$p(\mathbf{P}^c; \mathbf{\Theta}_i) = \mathcal{N}(\mathbf{T}_w^c \boldsymbol{\mu}_i, \mathbf{R}_w^c \mathbf{\Sigma}_i \mathbf{R}_w^{c\,\mathrm{T}})$$

Consider a sample $\mathbf{x}_s \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ and a monotonic continuous nonlinear function $\mathbf{y} = f(\mathbf{x})$ (where $\mathbf{y}$ and $\mathbf{x}_s$ are in the same space). The first order Taylor series expansion about a point $\mathbf{x}_s$ leads to

$$\mathbf{y} \sim \mathcal{N}\left(E[f(\mathbf{x})], \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_s} \mathbf{\Sigma} \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_s}^{\mathrm{T}}\right)$$

Under the standard pinhole projection model, a point $\mathbf{P}$ in the camera frame is projected to the image space using the operation $\pi : \mathcal{R}^3 \rightarrow \mathcal{R}^2$ defined as

$$\boldsymbol{\pi}(\mathbf{P}) = \begin{bmatrix} c_x + f\frac{\mathbf{P}_x}{\mathbf{P}_z} \\ c_y + f\frac{\mathbf{P}_y}{\mathbf{P}_z} \end{bmatrix}$$

where f is the focal length of the camera, and c_x and c_y are the principal point offsets. The derivative of the projection operation with respect to the 3D point $\mathbf{P}$ is

$$\frac{\partial \boldsymbol{\pi}}{\partial \mathbf{P}} = \begin{bmatrix} \frac{f}{\mathbf{P}_z} & 0 & -f\frac{\mathbf{P}_x}{\mathbf{P}_z^2} \\ 0 & \frac{f}{\mathbf{P}_z} & -f\frac{\mathbf{P}_y}{\mathbf{P}_z^2} \end{bmatrix}$$

Thus the projection of a 3D normal distribution component into image space is the 2D normal distribution

$$p(u, v; \mathbf{\Theta}_i) = \mathcal{N}\left(\boldsymbol{\pi}(\mathbf{T}_w^c \boldsymbol{\mu}_i), \left.\frac{\partial \boldsymbol{\pi}}{\partial \mathbf{P}}\right|_{\mathbf{T}_w^c \boldsymbol{\mu}_i} \mathbf{R}_w^c \mathbf{\Sigma}_i \mathbf{R}_w^{c\,\mathrm{T}} \left.\frac{\partial \boldsymbol{\pi}}{\partial \mathbf{P}}\right|_{\mathbf{T}_w^c \boldsymbol{\mu}_i}^{\mathrm{T}}\right) \tag{2}$$

where u, v are pixel coordinates in the image.

2.3. Estimating the Likelihood of a Camera Pose Hypothesis

As shown in the previous subsection, each Gaussian component can be projected into image space as a 2D Gaussian distribution. We utilize this property to determine relevant components for computing the likelihood of sensor data (Sec. 3). Given a scan $\mathbf{Z}_t$ of depth pixels $\{z_1, z_2, \ldots, z_k\}$ from a sensor scan and a set of 3D GMM parameters $\mathbf{\Theta}$, the log likelihood of the scan being sampled from the GMM is defined as

$$l(\mathbf{Z}_t | \mathbf{\Theta}, \mathbf{T}_w^c) = \sum_i^K \ln \sum_j^M \mathbb{1}_j \lambda_j \mathcal{N}(\boldsymbol{\pi}^{-1}(z_i); \mathbf{T}_w^c \boldsymbol{\mu}_j, \mathbf{R}_w^c \mathbf{\Sigma}_j \mathbf{R}_w^{c\,\mathrm{T}}) \tag{3}$$

where $\mathbb{1}_i$ is a binary indicator function that signifies if the i^{th} component is used to compute the log likelihood, $\boldsymbol{\pi}^{-1}$ is the inverse projection from depth image pixel to 3D points, and K is the number of pixels in the sensor scan. This likelihood should peak at the true sensor pose and decay smoothly in the local neighbourhood, which is indeed observed as shown in Fig. 2.

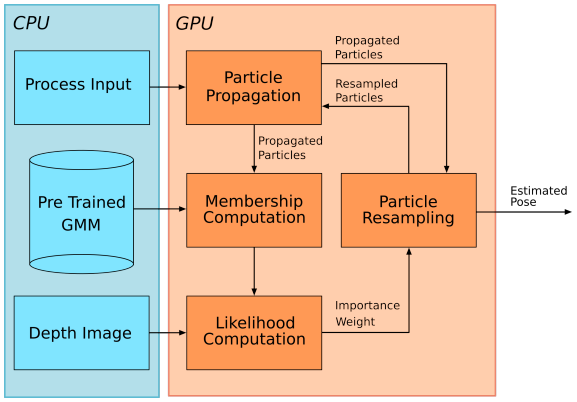

Figure 3. System overview. The algorithm operates on depth image streams and a source of odometry given a precomputed GMM map of the environment. For each particle, the GMM components are projected into image space using its current pose hypothesis. Relevant components are sub-selected and are then used to compute the likelihood of the depth image. The likelihood values for all the particles are used to resample a new set of particles that are then forward propagated using the process model.

2.4. Tracking Multiple Hypotheses

The discussion above only considers the nature of the likelihood in the vicinity of the true location; in practice it is not reasonable to assume that a single viewpoint suffices to localize the system as perceptual aliasing may arise due to a paucity of data that precludes state observability. Hence, we require a technique that permits tracking of multiple hypotheses and ensures appropriate weighting of equally likely viewpoints given the current sensor observations.

A standard approach to tracking multiple hypotheses is a Monte Carlo filter (or particle filter). Particle filters operate by continuously sampling candidate particle poses and measure the likelihood of the current sensor data having originated at the sampled pose. Based on the relative scores of the samples the particles are resampled and propagated based on a process model (often a noisy source of odometry). Convergence is generally achieved as soon as the sequence of observations made over time render alternate hypotheses inadmissible. Note that due to their inherent structure particle filters are extremely parallelizable and we exploit this in our implementation.

2.4.1 State Propagation

We assume the presence of some odometry to drive the first order Markov process model and inject Gaussian noise into it. Note that we assume that we know the pitch and roll that can be obtained from the attitude and heading reference system onboard a robotic system to a high level of accuracy.

2.4.2 Importance Weight

The importance weight of a particle in the filter represents a score of how well the sensor scan matches the GMM map at its location. Since the negative log likelihood of the current scan $\mathbf{Z}_t$ being drawn from the GMM map is a minimum at the true location, as shown in Fig. 2, in practice we use the inverse of the negative log likelihood. Thus, given the current state estimate $\mathbf{T}_w^{c^{(i)}}$ of a particle i out of N particles at time step t, the corresponding normalized importance weight is

$$w_t^{(i)} = \frac{l\left(\mathbf{Z}_t|\boldsymbol{\Theta}, \mathbf{T}_w^{c^{(i)}}\right)^{-1}}{\sum_j^N l\left(\mathbf{Z}_t|\boldsymbol{\Theta}, \mathbf{T}_w^{c^{(j)}}\right)^{-1}} \tag{4}$$

2.4.3 Sampling Strategy

A particle filter should ideally converge to the correct hypothesis after running for a finite amount of iterations with a reduction in the filter variance signifying the confidence of the filter. At the same time, an early reduction in the filter variance may cause the filter to diverge to an incorrect hypothesis and never recover due to low variance. In order to avoid such situations, we implement the stratified sampling strategy [13] in combination with low variance sampling [25]. The particles are divided into random groups of equal weights and in each group we employ low variance sampling. This approach has low particle variance [25] and works well when the particle filter is tracking multiple hypotheses at once.

2.4.4 Handling Particle Deprivation

One of the most common failure modalities of a particle filter is that of particle deprivation [26]. Even with a large number of particles, the stochasticity intrinsic to a particle filter might cause it to diverge from the correct state. We employ a modified version of Augmented MCL strategy as described in [25] where instead of adding new particles we reinitialize N_{modify} number of particles randomly selected from the original set using the parameters α_{slow} and α_{fast}. This is done since we cannot increase the number of particles once the filter is initialized because of implementation limitations. For our process model we use diagonal covariances for translation, and the final choice of parameters in all our experiments is shown in Table 1.

3. Fast Localization

In order to perform fast localization using the above approach it is essential to compute the likelihood of the data given a proposed pose as quickly as possible. Eq. 3 suggests that computing the likelihood of a scan having been sampled from the GMM map is the summation of the contribution of all the components within the GMM. However, the key insight here is that not all the components have a significant contribution to the likelihood.

The point clouds that we use in our experiments have roughly uniform coverage of points across the scene. As a consequence, all Gaussian components fit to these point-clouds end up having roughly equivalent mixture weight probabilities. This fact, in addition to the diminishing probability mass of the Gaussian distribution, permits the approximation of using only the projected components within spatial proximity of a certain pixel location for computing the likelihood of the corresponding 3D point being sampled from the map. As an added optimization step we perform this membership computation over subdivided patches of the image. These optimizations have negligible effect on the computed likelihood value of the sensor data, as demonstrated in Fig. 2.

We follow the following steps (graphically illustrated in Fig. 4) to obtain the relevant components for computing the likelihood of a depth image:

- Divide the image into 32×32 pixel patches;
- Compute the 2D projection of each Gaussian component on to the image plane of the depth sensor;
- Inflate the $3\boldsymbol{\Sigma}$-bound ellipse of the projected 2D Gaussian of each component by half the diagonal of the patch along its major and minor axis to generate ellipses $\mathcal{E}_i$; and
- For each patch, check if the center of the image patch c_p lies within or on each of the $\mathcal{E}_i$ and update the indicator variable $\mathbb{1}_{i,p}$ accordingly.

$$\mathbb{1}_{i,p} = \begin{cases} 1, \text{ if } c_p \in \mathcal{E}_i \\ 0, \text{ otherwise} \end{cases} \tag{5}$$

Given a set of updated indicator variables $\mathbb{1}_{i,p}$ for all the Gaussian components $\boldsymbol{\Theta}$ and a depth image, $\mathbf{Z}_t$, the likelihood of the image can be computed as the sum of the likelihoods of all the image patches computed according to Eq. 3.

4. Results

4.1. Experiment Design

This section presents performance analysis of our filtering approach on a variety of datasets. First, we conduct a sensitivity analysis to determine the number of particles and the number of components we use in our implementation. Second, we analyze metric accuracy of the proposed filter on publicly available datasets and show that our filter output is consistent with ground truth. Third, we compare the localization performance of our approach with a state-of-the-art RGB-D tracking algorithm (ORB-SLAM2 [15]) on the same sequences and demonstrate superior performance for localization. Fourth, we demonstrate the ability of our approach to incorporate both different odometry algorithms and ground truth map acquisition methodologies. Finally, we analyze runtime performance of our filter and show that its runtime is competitive both on a desktop class system and on an embedded platform, thus enabling SWaP constrained operation.

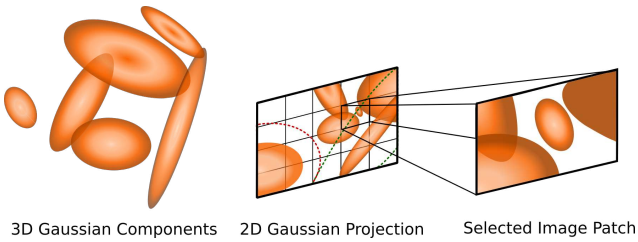

Figure 4. Membership computation process. 3D Gaussian components from the GMM representation of the world are projected to the image plane. The image is subdivided into multiple patches, where for a selected patch the relevant Gaussian members are determined for computing the likelihood. In order to determine the latter, we employ the heuristic described in Sec. 3. For instance the inflated bounds of the bottom left projected component (red) do not contain the center of the selected patch; in contrast those of the bottom right (green) do, and the component is thus selected for computing the likelihood of data within that particular patch.

We evaluate our approach on

- D1: The (a) lounge and (b) copyroom datasets [27];
- D2: The voxblox dataset [18];
- D3: A representative dataset collected in-situ; and
- D4: The TUM Freiburg3 dataset [24] for demonstrating the ability to generalize.

In all cases we utilize a fixed number of components (Sec. 4.2) to first fit a GMM to the pointcloud using the scikit-learn[1] toolkit.

We employ two processing systems for evaluation: (1) A desktop with an Intel i7 CPU and an NVIDIA GTX 960 Ti GPU, and (2) An embedded NVIDIA TX2 platform.

4.2. Sensitivity Analysis

Particle filters can achieve increased performance with large number of particles at the cost of increased computational complexity. Conversely too few particles can lead to divergence from the true location due to an inability to represent the true underlying distribution. In order to find the appropriate number of particles that ensure precision while still being computationally feasible we compare the

[1] http://scikit-learn.org/stable/modules/mixture.html

Table 1. Filter hyperparameters

	Process Noise σ		α_{slow}	α_{fast}
	Translation (m)	Yaw (rad)		
Desktop	0.02	0.01	0.01	0.001
TX2	0.025	0.1	0.05	0.005

filter performance with various number of particles against a ground truth filter with $N = 16200$. Assuming the underlying distribution represented by the particle set to be a unimodal Gaussian (a valid assumption after convergence), we compute the variance of the KL-Divergence [11] of multiple runs of the filter output with that of the ground truth filter to determine the empirically optimal parameters to be used in our implementation. A low value of the KL-Divergence variance indicates similar performance to the ground truth filter.

The fidelity of a GMM map to the underlying distribution monotonically increases with the number of components. However, the marginal benefit (in a KL-Divergence sense) of increasing the model complexity diminishes rapidly after adding an adequate number of components [21]. In order to determine the appropriate model complexity to represent the original map concisely while enabling accurate filter performance, we perform similar experiments with the optimal number of particles obtained from the previous study, this time with varying number of Gaussian components.
We compute the optimal parameters to be $N = 1068$ and $M = 1000$ based on D3, the dataset with the largest volumetric span. This specific parameter choice is further motivated by implementation constraints.

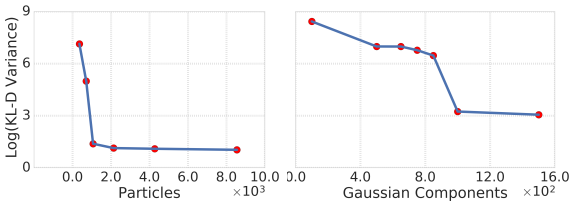

Figure 5. *Left*: Log of variance of KL-Divergence between the ground truth filter ($N = 16200$) and filters with reduced particle counts. The knee point implies similar performance to the ground truth filter at particles counts $N > 1000$. *Right*: A similar comparison given a ground truth map with many components ($M = 2500$) and those with a reduced number motivates the choice of $M >= 1000$. Evaluated on D3.

4.3. Metric Accuracy Analysis

In this subsection we discuss the localization accuracy of our approach. As mentioned in Sec. 3 since we do not add new particles when the filter observes particle deprivation and instead randomly reinitialize the particles from the original set, the Root Mean Squared Error (RMSE) of the filter estimate increases when the filter observes particle deprivation. This is highlighted in the plots as vertical shaded regions. For all our evaluations we run the filter 10 times on each dataset and report the average of the mean filter estimate. We do not quantify the sensitivity of the likelihood values to the AHRS pitch and roll estimates as they are accurate enough to not cause any significant difference.

4.3.1 Evaluation with Ground Truth Datasets (D1, D2)

The objective of using these datasets is to demonstrate the ability of the filter to converge to the ground truth given perfect odometry. We generated a GMM map of the environments using the reconstructed point cloud and used the delta transforms between two consecutive reported sensor poses with added noise as our process model. In all these experiments, we initialized the particles from a uniform distribution over a 4 m cube and π radians yaw orientation around the known initial location. D1(a) and D1(b) contain nominal motion of the sensor, while D2 consists of very aggressive motion in all degrees of freedom.

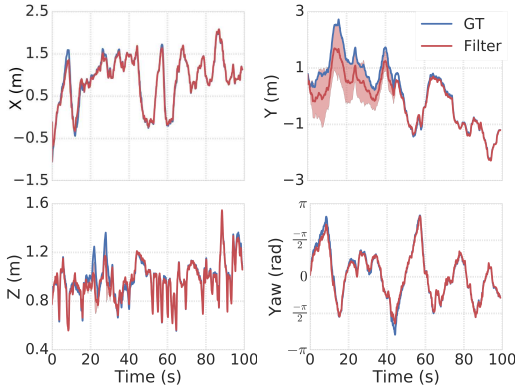

Figure 6. Mean trajectory (red) of the particle filter estimate of 10 trials on the D1(a) dataset compared to the process model trajectory (blue). The shaded region around the mean trajectory shows the variance of the filter estimate over multiple runs. The filter estimates have high variance in the beginning of the trajectories, but soon converge to the correct location and track the ground truth trajectory (blue).

The filter estimate converged to an incorrect hypothesis for some runs in the initial iterations due to the highly symmetric nature of the environments about the X axis, as can be seen in Fig. 6. The RMSE of the filter poses for these datasets is presented in Fig. 7.

4.3.2 Evaluation with Representative Dataset (D3)

The objective of using this dataset is to demonstrate results on a real-world application of the filter. We no longer use ground truth odometry. Additionally, since we don't have a baseline algorithm to directly compare against, we compare

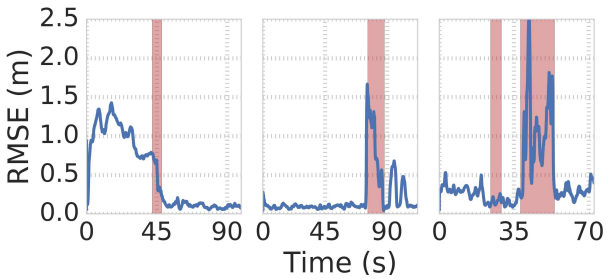

Figure 7. RMSE of 10 trials of the particle filter on the D1(a), D1(b), and D2 datasets respectively. The region in red indicates the time at which the particle filter observes particle deprivation and a consequential RMSE rise.

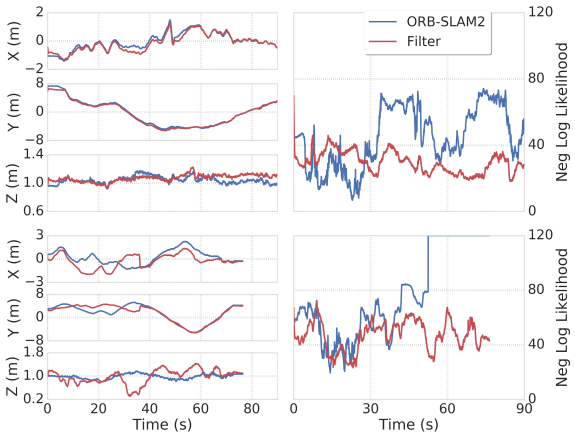

Figure 8. Comparison between the position and corresponding likelihood estimates for two runs from ORB-SLAM2 and our filter, respectively. *Top*: A nominal path with feature rich data, and *Bottom*: A path moving through regions of low observability. Contrast the continually increasing divergence (capped in the graph) of the ORB-SLAM2 estimate after moving through the feature poor region with the lower snapped negative likelihood values for the same locations for our filter. The corresponding poses and overlaid depth scan at approximately 55s is shown in Fig. 9. Due to a minimal overlap of the depth scan with the map for the ORB-SLAM2 frame, the likelihood value is very low.

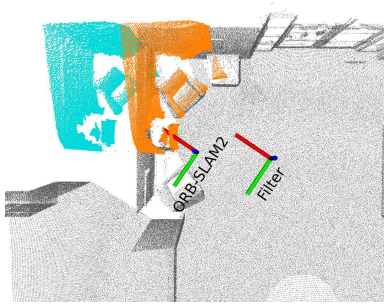

Figure 9. Comparison of registration of current sensor measurement at ground truth point cloud (gray) at ORB-SLAM2 pose estimation (cyan) and at the estimated filter pose (orange). The sensor measurement aligns with the ground truth point cloud in the filter estimate frame while the accumulated drift in the ORB-SLAM2 frame due to transition through a less feature rich region leads to poor alignment.

the localization performance against ORB-SLAM2 which builds its own succinct map representation. Note that ORB-SLAM2 also utilizes the RGB image data in the dataset whereas we only use the depth. Finally, we also briefly contrast the performance of the filter on the same dataset.

We generate a ground truth pointcloud using a FARO Focus 3D Laser scanner [2] and use an ASUS Xtion RGB-D camera for acquiring sensor data. An IMU strapped to the camera determines the roll and pitch of the sensor. For odometry, we only use the frame to frame relative transform as opposed to the global pose output from ORB-SLAM2 as input to the process model. Note that the global ORB-SLAM2 position we compare to in Fig. 8 and Fig. 9 is using loop closure to mitigate the drift in its frame to frame estimates.

As ground truth is not available for this dataset, we report the negative log likelihood values at the mean particle filter location and the reported ORB-SLAM2 poses. We show results of two runs in this environment in Fig. 8: The first through a nominal path with feature rich data (as shown in detail earlier in Fig. 1) where the estimated positions of the sensor for the two approaches are very similar (but with worse likelihood values for ORB-SLAM2). The second run demonstrates the advantage of using particle filters over maximum likelihood estimators in that the former can converge to the correct result even after moving through a region of low observability. We observe that the sensor measurements register at the converged filter location better after snapping back than those for the ORB-SLAM2 estimate, as can be qualitatively seen in Fig. 9.

The particles in these experiments are initialized from a uniform distribution over a $4\text{m} \times 8\text{m} \times 3\text{m}$ for position and π radians in yaw.

[2] https://www.faro.com/products/construction-bim-cim/faro-focus/

4.3.3 Evaluation with TUM Dataset (D4)

To demonstrate the ability of our filter to generalize to both different odometry algorithms and datasets we compare the performance with three different odometry inputs as process models: The Generalized-ICP algorithm [20], ORB-SLAM2 frame-to-frame relative transform, and ground truth odometry. The point cloud map of the environment was created by stitching several sensor scans together using their corresponding ground truth poses. In spite of the stitched point cloud not being as well registered as that from

a FARO scanner due to sensor and ground truth pose noise, the performance of the filter is similar (Table 2).

Table 2. Performance on D4 (RMSE in cm)

Our Approach			ORB-SLAM2
Process Input	mean	var (cm^2)	
ORB-SLAM2 Velocity	7.67	0.21	
Ground Truth Velocity	7.56	0.28	4.55
G-ICP Velocity	9.07	0.21	

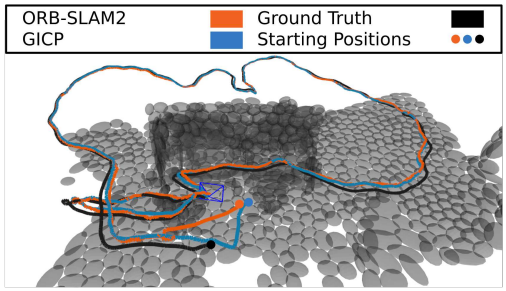

Figure 10. Comparison of our particle filter approach using ORB-SLAM2 frame-to-frame odometry (orange) and Generalized-ICP (cyan) as process models with ground truth pose (black) on TUM's Freiburg 3 Desk Dataset. The GMM representation of the world is created by stitching sensor scans using the ground truth pose estimates. The higher global error of our approach than that of ORB-SLAM2 can be attributed to the noisy reconstruction of the environment point cloud from the accumulated scans.

4.4. Runtime Performance Analysis

As seen in Fig. 11, the likelihood evaluation is the most computationally expensive operation. Execution time for this step varies with the number of Gaussian components used to compute likelihood for each image patch and therefore is dependent on the fidelity of the model.

The filter runs at an average rate of 80 Hz and 9.5 Hz on the Desktop and embedded class systems, respectively. This is comparable to the ORB-SLAM2 rates of 47 Hz and 20 Hz on the respective platforms. Initial convergence on the TX2 is slower due to the implicitly larger odometry steps. However, post convergence the metric performance is not significantly affected. As an illustrative example, the impact of the slower runtime performance on the TX2 for D1(a) is demonstrated in Fig. 12.

5. Summary and Future Work

We present a framework to perform real-time localization of depth sensors given a prior continuous spatial belief representation. Key to being able to do this is the ability to project a succinct representation into the image frame of the sensor to evaluate the likelihood of the data having originated from the given map for a given pose. By utilizing a

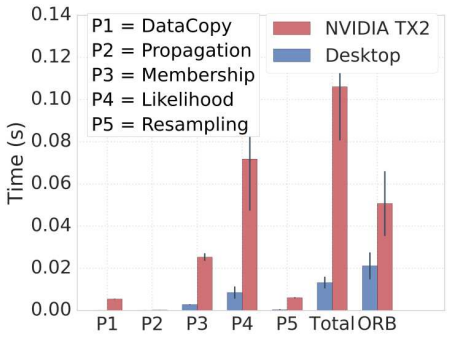

Figure 11. Execution time comparison for subcomponents of the algorithm for the D1(a) dataset on an Intel i7 desktop with an NVIDIA GPU and an embedded NVIDIA TX2 platform. Performance scales linearly with the number of CUDA cores. As a point of comparison ORB-SLAM2 runtime on the same dataset is faster on the embedded platform than on the desktop.

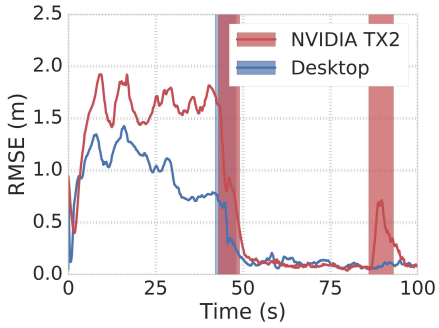

Figure 12. Comparison of the filter performance on the desktop with the NVIDIA TX2 on the D1(a) dataset. As the filter operates at a slower frame rate on the TX2 it initially exhibits a larger error but once the sensor observes a uniquely identifiable location, both trial sets converge to the ground truth location.

fast likelihood computation approximation we can then perform robust particle filter localization in real-time even on an embedded GPU platform.

Despite the apparent suitability of the likelihood function to an optimization based approach for more fine grained pose refinement, it is not so straightforward. For Gaussian components that are more flat the gradient profile can vary rapidly in the vicinity of the components leading to poor conditioning that can challenge traditional iterative Gauss-Newton descent strategies. Further, the absence of strong gradient information in spatial data as encoded by the necessarily smooth-by-construction representation hinders the application of such methods.

Future steps will involve extending the formulation to hierarchical map representations to localize over larger scale environments. We also intend to investigate incorporation of this approach as a possible robust relocalization backend for a visual SLAM algorithm running onboard a relevant SWaP constrained aerial platform.

References

[1] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions on Robotics*, 24(5), 2008. 1

[2] J. Biswas and M. Veloso. Depth camera based indoor mobile robot localization and navigation. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2012. 1

[3] A. Bry, A. Bachrach, and N. Roy. State estimation for aggressive flight in GPS-denied environments using onboard sensing. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2012. 1

[4] A. Burguera, Y. González, and G. Oliver. The likelihood field approach to sonar scan matching. In *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, 2008. 1

[5] A. Das, J. Servos, and S. L. Waslander. 3D scan registration using the normal distributions transform with ground segmentation and point cloud clustering. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2013. 1

[6] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz. MLMD: Maximum Likelihood Mixture Decoupling for fast and accurate point cloud registration. In *Proc. of IEEE Intl. Conf. on 3D Vision*, 2015. 1

[7] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz. Accelerated generative models for 3D point cloud data. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 2016. 3

[8] M. F. Fallon, H. Johannsson, and J. J. Leonard. Efficient scene simulation for robust Monte Carlo localization using an RGB-D camera. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2012. 1

[9] Z. Fang and S. Scherer. Real-time onboard 6DoF localization of an indoor MAV in degraded visual environments using a RGB-D camera. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2015. 1

[10] T. Gonçalves and A. I. Comport. Real-time Direct Tracking of Color Images in the Presence of Illumination Variation. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2011. 2

[11] J. R. Hershey and P. A. Olsen. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *Proc. of the IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*, volume 4, 2007. 6

[12] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3), 2013. 1

[13] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1), 1996. 4

[14] M. Magnusson, N. Vaskevicius, T. Stoyanov, K. Pathak, and A. Birk. Beyond points: Evaluating recent 3D scan-matching algorithms. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2015. 1

[15] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5), 2017. 5

[16] S. Oishi, Y. Jeong, R. Kurazume, Y. Iwashita, and T. Hasegawa. ND voxel localization using large-scale 3D environmental map and RGB-D camera. In *Proc. of IEEE Intl. Conf. on Robotics and Biomimetics*, 2013. 1

[17] K. Ok, W. N. Greene, and N. Roy. Simultaneous Tracking and Rendering: Real-time Monocular Localization for MAVs. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2016. 2

[18] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-Board MAV Planning. In *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. IEEE, 2017. 5

[19] G. Pascoe, W. Maddern, and P. Newman. Robust Direct Visual Localisation using Normalised Information Distance. In *Proc. of British Machine Vision Conf.*, 2015. 2

[20] A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Proc. of Robotics: Science and Systems*, volume 2, 2009. 7

[21] S. Srivastava and N. Michael. Approximate Continuous Belief Distributions for Precise Autonomous Inspection. In *Proc. of IEEE Intl. Symposium on Safety, Security, and Rescue Robotics*, 2016. 1, 3, 6

[22] A. D. Stewart and P. Newman. LAPS-Localisation using Appearance of Prior Structure: 6-DoF Monocular Camera Localisation using Prior Pointclouds. In *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2012. 2

[23] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal. Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. *The Intl. Journal of Robotics Research*, 31(12), 2012. 1

[24] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. In *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems,*. 5

[25] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT press, 2005. 1, 4

[26] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. A. Wan. The unscented particle filter. In *Advances in neural information processing systems*, 2001. 4

[27] Q.-Y. Zhou and V. Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (TOG)*, 32(4). 5