Map Compressibility Assessment for LiDAR Registration

Ming-Fang Chang[1], Wei Dong[1], Joshua Mangelson[2], Michael Kaess[1], and Simon Lucey[1,3]

***Abstract*— We aim to assess the performance of LiDAR-to-map registration on compressive maps. Modern autonomous vehicles utilize pre-built HD (High-Definition) maps to perform sensor-to-map registration, which recovers pose estimation failures and reduces drift in a large-scale environment. However, sensor-to-map registration is usually realized by registering the sensor to a dense 3D model, which occupies massive storage space in the HD map and requires much data processing overhead. Although smaller 3D models are preferable, the optimal compressive map format for preservation of the best registration performance remains unclear.**

In this paper, we propose a novel and challenging benchmark to evaluate existing LiDAR-to-map registration methods from three perspectives: map compressibility, robustness, and precision. We compared various map formats, including raw points, hierarchical GMMs, and feature points, and show their performance trade-offs between compressibility and robustness on real-world LiDAR datasets: KITTI Odometry Dataset and Argoverse Tracking Dataset. Our benchmark reveals that state-of-the-art deep feature point based methods outperform traditional methods significantly when the map size budget is high. However, when map size budget is low, deep methods are outperformed by the methods using simpler models in Argoverse Tracking Dataset due to poor spatial coverage. In addition, we observe that the recently published TEASER++ significantly outperforms RANSAC for the feature point methods. Our analysis provides a valuable reference for the community to design budgeted real-world systems and find potential research opportunities. We will release the benchmark for public use.

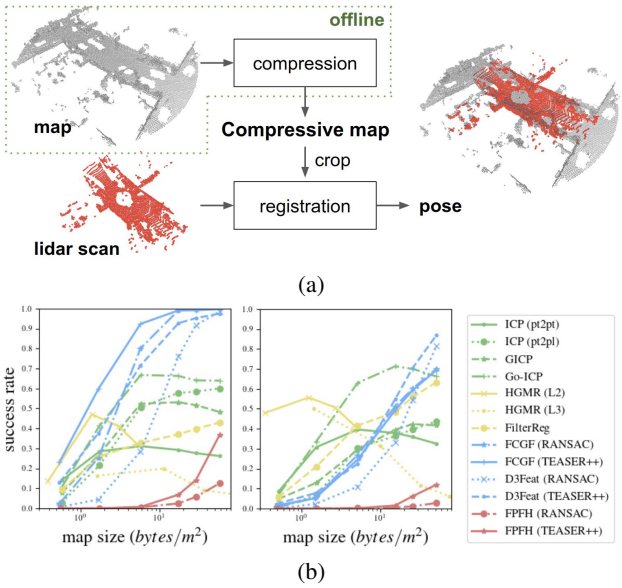


Fig. 1. (a) The system pipeline of the proposed compressive registration. We compress the map offline, and register the online LiDAR input to the compressive map. (b) The success rate trends of the evaluated methods under different map size budgets on KITTI Odometry Dataset (left) and Argoverse Tracking Dataset (right).

I. INTRODUCTION

Maps are essential for modern autonomous driving systems. A map with rich prior knowledge provides valuable, offline-refined information that is not observable by online sensors, and thus improves the system performance. Modern maps, such as the HD maps used by autonomous vehicles, mostly contain high-quality dense 3D models and semantic labels. However, these dense 3D models require vast storage space and cause extra online data processing overhead.

The dense 3D model is mainly used to achieve accurate sensor-to-map registration, which is a crucial task for the autonomous vehicles to re-localize against the map when pose estimation fails, and also to reduce pose drifting errors in large-scale environments. Recently, Martinez *et al.* proposed a benchmark for retrieval-based localization methods because the dense HD maps are too expensive to collect and build at scale [2]. However, without the prior knowledge from the map, such retrieval-based methods require much training data and generalize poorly in unseen environments.

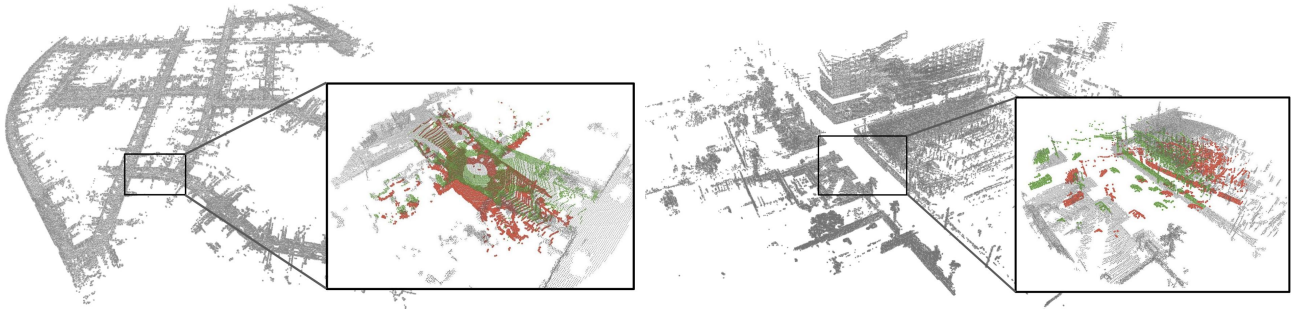
In practice, the dense 3D models are unnecessary for the essential tasks in autonomous driving other than relocalization. For example, motion planning, motion forecasting, object tracking, and obstacle avoidance only require the sensor input and the semantic map labels with rough 3D information, such as lane directions and the bounding boxes of the traffic lights. Since other information in the HD map is much lighter in size, eliminating the need of dense 3D models in the sensor-to-map registration process would reduce the total HD map size significantly.

Although eliminating the need of the dense 3D model is desirable, it deserves more research attention. Most existing point cloud registration studies focused only on the accuracy and speed of registering two scans with similar data distributions, while the data distributions of a sensor scan and a map are very different. Relevant benchmarks evaluate the point cloud compression performance by reconstruction accuracy, not by sensor-to-map registration accuracy [3]. In fact, loading a perfectly reconstructed dense 3D model is unnecessary if accurate sensor-to-map registration can be achieved with a lighter map. Although some works have evaluated sensor-to-map registration against map compression ratio [4], [5]

[1]Ming-Fang Chang, Wei Dong, Michael Kaess, Simon Lucey are with Carnegie Mellon University {mingfanc, weidong, kaess, slucey}@andrew.cmu.edu

[2] Joshua Mangelson is with Brigham Young University joshua_mangelson@byu.edu

[3] Simon Lucey is also part of the Australian Insitute of Machine Learning (AIML) at the The University of Adelaide



(a) (b)

Fig. 2. The maps and input LiDAR scans from (a) KITTI Odometry Dataset and (b) Argoverse Tracking Dataset. The input noisy poses are shown in red, and the ground truth poses are shown in green. We removed other vehicles using PV-RCNN [1] for (a) and the provided driveable region map for (b).

for the proposed specific data formats, there is no universal standard available for a fair quantitative comparison among different compressive map formats.

In this paper, we focus on a popular setting – registering a 3D LiDAR *scan* to a 3D *map*, which is the most common configuration for the modern autonomous vehicles to perform sensor-to-map registration. The raw map in this case is a high-quality, dense, and large-scale point cloud built offline. We propose that a sensor-to-map registration algorithm should operate directly on a certain compressive map format, instead of the raw point cloud, to eliminate the need of storing and processing the original large-scale point cloud. We refer to this pipeline as *compressive registration* in the following. The proposed compressive registration pipeline, as shown in Figure 1, has several advantages over the methods using raw point cloud maps: 1) The map feature can be precomputed offline since it does not require any online input. 2) The online map data decompression, if needed, takes less time since it does not need to recover a dense 3D map. 3) It takes much less storage space and data transmission time. As a result, we are interested in the sensor-to-map registration methods that directly operate on compressive formats.

We propose the first benchmark for compressive LiDAR-to-map registration. Given initial inaccurate LiDAR pose estimations, we evaluated the LiDAR-to-map registration performance on various compressive maps, including raw points, hierarchical GMMs, and feature points, under different map size budgets. Our benchmark is challenging due to the different data distribution of the LiDAR scans and the maps. We design universal map size based metrics for quantitative comparison. Our results illustrate the different trade-off trends between map size and robustness of the recent deep-learning based methods and classical methods. We show that the deep-learning based methods performed the best under high map size budgets but might perform worse than the classical methods using simpler models under low map size budgets, depending on the local map structure. As an additional contribution, we analysed the robust registration methods, RANSAC and recent TEASER++ [6] together with various 3D features and show that TEASER++ in general outperforms RANSAC. To summarize, our contributions are:

- We propose the first compressive LiDAR-to-map registration benchmark. Our benchmark evaluates the map compressibility, robustness, and precision, and can be applied to various map formats.
- We evaluated both recent deep learning based and classical point cloud registration methods, including raw point based, GMM based, and feature point based methods. Our quantitative results reveal the trade-offs made by different methods and provide a valuable reference for future research.
- We will release the benchmark for the community to evaluate more methods conveniently in the future.

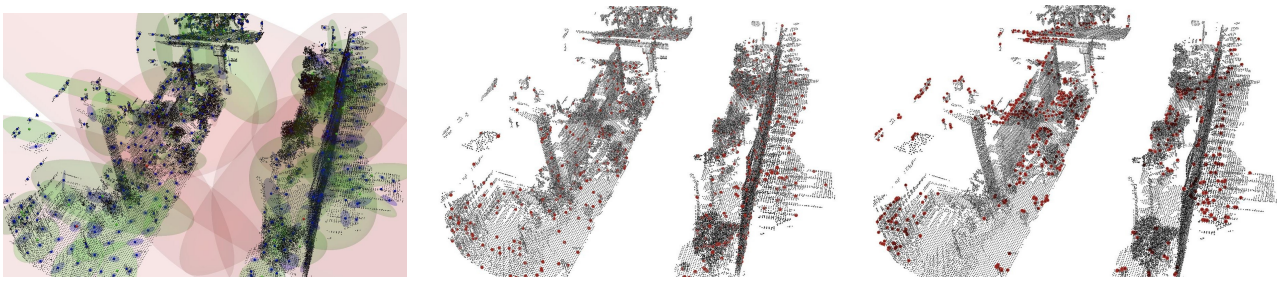
II. Related Work

In this section, we categorize and discuss existing point cloud registration methods by the corresponding compressive map formats. Due to space limitation, we refer to [3] for additional compression tools that focus on reconstruction accuracy – they can be applied on top of the following compressive maps, such as Octree [7] and bzip2 [8].

A. Raw Point Clouds

We list raw point cloud based methods in this section. Iterative Closest Point (ICP) [10] registers two point clouds by iteratively finding the closest point pairs and computing the transformation matrix based on the found pairs. Since its debut in the early 90s, researchers have proposed a tremendous amount of ICP variants. ICP and its variants are still arguably the most widely adopted point cloud registration method in practical systems nowadays, despite its well-known drawback of being easily trapped in a local minimum.

The efficiency and accuracy of ICP variants mainly depend on the method of point correspondence search between source and target point clouds, and the quality of initialization. Greenspan and Yurick [11] proposed speeding up the correspondence search using a k-D tree. Generalized ICP (G-ICP) provides a probabilistic formulation that unifies point-to-point and point-to-plane ICP [12]. Modern off-the-shelf ICP tools such as PCL (Point Cloud Library) [13] and Open3D [14] are still vulnerable to local minima and require good initialization. Yang *et al.* [15] proposed Go-ICP that performs a global search to avoid the local minima at the cost of slow speed. As for reducing the size of the raw point cloud



(a) (b) (c)

Fig. 3. Visualization of different compressive map formats (a) hierarchical GMM tree from HGMR [9]. The red, green, and blue colored ellipsoids represent a three-level GMM tree (b) Randomly downsampled points (red) (c) score-based downsampling used with D3Feat [4].

maps, Yin *et al*. [5] proposed to use the hit frequency as an indicator to prune LiDAR maps. Dubé *et al*. [16] proposed SegMap, which compresses semantic map segments with a 3D auto-encoder network and reconstructs raw point clouds for registration.

B. Feature Points

Compressing an input point cloud into representative key points with descriptors can potentially reduce the map size and improve the robustness of correspondence search. Feature point correspondences can be extracted by comparing the feature descriptors and the registration can be solved globally in a closed form using the Procrustes algorithm [17]. Rusu *et al*. [18] proposed Fast Point Feature Histograms (FPFH) as the descriptor for finding robust point correspondences. The noisy initial correspondences found by the descriptor matching can be filtered by robust methods such as RANSAC and the recently proposed TEASER++ [6].

Deep networks can be used to detect feature points and extract descriptors from raw point clouds. Wang *et al*. used attention-based modules and the information from the other point cloud to learn the feature descriptors and the correspondences [19]. Choy *et al*. [20] proposed Fully Convolutional Geometric Features (FCGF), which uses a 3D sparse fully-convolutional network to extract per-point descriptors, and a follow-up work [17] uses a 6-D sparse fully-convolutional network to predict point correspondences. Bai *et al*. proposed D3Feat [4] that uses KPConv [21] to extract dense point features, and trains point features by distance-learning losses for robust matching and importance scores. Points with low importance scores are pruned to compress the map.

C. Shape Models

The local point cloud structure can be represented by compressive shape models, and registration can be performed without recovering the raw points. GMM-based methods use Gaussian models to approximate the local shape of point clouds and perform GMM-to-GMM or point-to-GMM registration using the EM algorithm [22]. Normal Distributions Transform (NDT) based methods perform efficient registration between NDT models [23]. Eckart *et al*. [9] propose a hierarchical, anisotropic GMM tree for coarse-to-fine registration. Gao and Tedrake [24] proposed FilterReg that accelerates the EM algorithm by formulating the E-Step as a 3D filtering problem. Yuan *et al*. proposed DeepGMR [25] that replaces the E-Step using an end-to-end trainable network. Without keeping the local structure, PointNetLK [26] compresses a whole point cloud into a single feature embedding using PointNet and performs direct feature registration with the feature embeddings.

Beyond the scope of this paper, the intensity information is proven to be useful in LiDAR-based localization [27]. It would be interesting to explore the role of intensity in LiDAR map compression as a future work.

III. Overview

We propose a universal benchmark for compressive sensor-to-map registration for various compressive map formats. The proposed compressive registration pipeline is illustrated in Figure 1. In the pipeline, we first perform offline map feature computation and compression, crop the local map using a noisy initial pose, and then register an input LiDAR scan to the cropped compressive map. The LiDAR scan is converted into the corresponding format used in the evaluated registration methods, such as feature points or GMMs. Let $\mathbf{P}$ be the source point cloud and the input LiDAR scan, $\mathbf{Q}$ be the target point cloud and the cropped map, and $\mathbf{T} \in \mathrm{SE}(3)$ be the transformation matrix that comprises the rotation matrix and the translation. The problem of point cloud registration can be defined as:

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} \ \mathcal{L}\Big(f(\mathbf{T}, \mathbf{P}), \mathbf{Q}\Big), \tag{1}$$

where $f(.)$ denotes the point cloud transformation function, and $\mathcal{L}$ denotes the cost function used in the point cloud registration method. The cost function $\mathcal{L}$ varies among different methods. For example, point-to-point ICP uses Euclidean distances between selected point pairs and point-to-plane ICP uses squared distance from a point to a paired local plane patch. For methods that operate on other formats instead of raw point clouds, denoting $\phi(.)$ as the general feature extraction function, Eq. (1) becomes:

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} \ \mathcal{L}\Big(f(\mathbf{T}, \phi_p(\mathbf{P})), \phi_q(\mathbf{Q})\Big). \tag{2}$$

Notice that $\phi_p(.)$ and $\phi_q(.)$ are not necessarily the same. For example, one can register a raw point cloud to a GMM model.

We assume a noisy initial pose is available – in practice, an autonomous vehicle receives the GPS signal and performs pose estimation on-the-go. In reality, the map is stored in the world coordinate frame. Let the transformation from the local LiDAR frame to the world frame be $\mathbf{T}_l^w$ and ideally $f((\mathbf{T}_l^w)^{-1}, \mathbf{Q})$ would be aligned with $\mathbf{P}$. And the map feature extraction $\phi_q(\mathbf{Q})$ should happen in the world coordinate frame since the initial pose is not available in the offline map preprocessing step. Letting $\mathbf{T}_{ini}$ be an initial noisy estimation of $\mathbf{T}_l^w$, Eq. (2) can be rewritten as:

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} \mathcal{L}\Big(f(\mathbf{T}, \phi_p(\mathbf{P})), f(\mathbf{T}_{ini}^{-1}, \phi_q(\mathbf{Q}))\Big). \quad (3)$$

IV. Method Categories

We categorize registration methods by map data types and techniques used for compression. A list of related methods is shown in Table I, whose attributes are explained as follows:

- **Map type**: the actual data format used for registration, such as raw points, GMMs, and feature points.
- **Data dimension**: the dimension of the used data format. For example, point-to-point ICP uses only xyz coordinates so the dimension is 3. Point-to-plane ICP and GICP use the additional 3D normals thus the dimension is $3 + 3 = 6$.
- **Global**: the method does not require a good initial pose.
- **Scalable**: the method is feasible for building a large-scale compressive map.
- **Deep**: the method is deep learning based.

Some methods are not considered to be scalable for practical reasons: Go-ICP [15] and CPD [22] are much slower than other methods when running with our LiDAR point clouds. The feature dimensions of DCP [19] and LORAX [29] are very high and lead to huge map size if we compute and store the features in the map. The sparse 6-D convolutional network in DGR [17] is not applicable to very sparse inputs when map size budget is slow. The PointNet backbones used by DeepGMR [25] and PointNetLK [26] are only suitable for small object-scale point clouds.

V. Benchmark for Compressive Registration

A major difference between our benchmark and other existing evaluations is the asymmetry between the source (the LiDAR scan) and the target (the map). A LiDAR scan is sparser, noisier and contains moving objects (*e.g.*other vehicles), while a map is denser, pre-built, and refined by denoising and moving object removal. Please see Figure 2 for the visualizations of our LiDAR scans and the maps.

To cover the initial error range in the real environment, we applied uniformly distributed noise within $[-10, 10]m$ to the xyz dimensions of translation and $[-10, 10]^\circ$ to the roll, pitch, and yaw rotation angles. This error range covers most of the possible GPS errors of a modern autonomous vehicle, according to [2]. To evaluate the pipeline in Figure 1 with large-scale maps, we first preprocess the dataset into pairs of local maps and lidar scans. For each pair, We cropped a local map region within a $40m$ range around the initial pose, and then compress the local the map region to perform registration. The order of map cropping and compression does not affect the compression result for the compression methods used in this work. For the feature extraction methods such as FPFH [18], FCGF [20], and D3Feat [4], we precomputed feature extraction in the world coordinate frame, since the initial pose was not available when performing offline map compression.

A. Data Preparation

For this work, we focus on the autonomous driving scenario and prepared data from two real-world autonomous driving datasets: the KITTI Odometry Dataset [30] and the Argoverse Tracking Dataset [31] (KITTI and Argoverse for short). We aggregated the LiDAR scans to build a dense point cloud map. For KITTI, the provided ground truth poses are noisy, so we used the the poses estimated by SLAM [32]. We used the provided ground truth poses for the Argoverse. Considering that vehicles are the most common moving objects in the autonomous driving scenario, we removed vehicles form the maps. For KITTI, the vehicles were first detected by PV-RCNN [1] and then removed from the input LiDAR scans before building the map. For Argoverse, the vehicles were removed by pruning the driveable regions in the LiDAR scans before building the maps.

As for source clouds, we used the LiDAR scans from sequence 00 of KITTI and the test set from Argoverse. The KITTI sequences 03, 05, 07, 09 and the Argoverse training set were used to train the deep learning based methods. We applied a simple threshold-based ground removal to the input scans of Argoverse to match the map point distribution, as visualized in Figure 2. The KITTI data contains 2271 scans and a map area of 4,678,598 m^2. The Argoverse data contains 1545 scans and a map area of 3,590,315 m^2. The map area is computed by the area of occupied regions at the m^2 resolution. A LiDAR scan of both datasets contain 64 bins and the FoV of a Argoverse LiDAR scan is wider (50°) than a KITTI LiDAR scan (26.9°).

B. Evaluation Metrics

We define evaluation metrics with the robustness and precision at different map sizes. The robustness is measured by *success rate* (also referred to as *recall*), and the precision is measured by the translation and rotation errors among successful samples. Because the total map area varies, we quantify map size with density $bytes/m^2$. Since the actual speed evaluation largely depends on the implementations and varies across different platforms, we refer the readers to the original papers for detailed speed comparisons.

Let $\mathbf{R} \in \mathrm{SO}(3)$ be a rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ be a translation vector from $\mathbf{T}$. We measure the precision by:

- **Translation Error (TE)**: the median of the L2 distance between the translation vectors of the successful pairs:

$$\mathrm{TE} = ||\mathbf{t} - \mathbf{t}_{gt}||_2^2. \quad (4)$$

TABLE I

A LIST OF RELATED REGISTRATION METHODS AND THE CORRESPONDING CATEGORIES.

Map type	Method Name	Data Dim.	Deep	Global	Scalable
raw points	ICP (pt2pt)[1] [10]	3			✓
	ICP (pt2pl)[2] [28]	6			✓
	GICP [12]	6			✓
	Go-ICP [15]	3		✓	
GMMs	CPD [22]	3			
	NDT [23]	9			✓
	HGMR[9]	10			✓
	FilterReg [24]	3[3]			✓
	DeepGMR [25]	5[4]	✓	✓	
feature points	FPFH [18]	36		✓	✓
	DCP [19]	515	✓	✓	
	FCGF [20]	35	✓	✓	✓
	D3Feat [4]	35	✓	✓	✓
	DGR [17]	35	✓	✓	
global embedding	PointNetLK [26]	1024	✓		
hybrid	LORAX [29]	1035[5]	✓	✓	

- **Rotation Error (RE)**: the median of the rotation angle between the rotation matrices of the successful pairs:

$$\text{RE} = \arccos \frac{\text{tr}(\mathbf{R}\mathbf{R}_{gt}^T) - 1}{2}. \tag{5}$$

Here the subscript gt denotes the ground truth.

We measure the robustness by:

- **Success Rate (SR)**: the ratio of the pairs with both translation and rotation error lower than the assigned successful threshold. We choose the successful thresholds to be 2m for TE and 5° for RE as [4], [33].

We attach numbers to the metrics to represent the value under a map size budget. For example, SR10 refers to the success rate measured give map size budget 10 $bytes/m^2$.

VI. EVALUATION

We evaluated the state-of-the-art point cloud registration methods using the proposed benchmark. Overall our benchmark successfully spotted interesting trade-offs between the map size and the success rate in real-world environments. Note that our benchmark is very challenging due to the map size constraints and the different data distribution between input LiDAR scans and the map. The detailed results are shown in Table III and IV. The success rate curves using different success thresholds are shown in Figure 6.

Since data dimensions of different methods vary as shown in Table I, we used the raw point format as the standard for feature point based methods. Let the number of raw points be N_r, x be the name of a method, F_x be the data dimension of method x. We computed the number of feature points N_x of method x by

$$N_x = \frac{3N_r}{F_x}. \tag{6}$$

For example, corresponding to raw point based methods with 5000 raw points, a feature point based method with feature descriptor dimension 32 has a total dimension $32 + 3 = 35$, so the number of feature points corresponding to the 5000 raw points is approximately $\frac{3\times5000}{35} \approx 429$. For HGMR, constrained by the predefined tree structure, we evaluated with tree levels 2 and 3, and spanning node numbers $n \in [4, 6, 12, 16]$. This generated the different map size range of HGMR (L2) and HGMR (L3) against other methods in Figure 1. See Table II for an intuitive data size comparison. Here we computed the raw data size despite the possibility of using additional compression tools, which can be applied on top of all the methods.

A. Raw Points

Among the methods using raw points for registration, we evaluated the classical point-to-point ICP, point-to-plane ICP, GICP, and the global method, Go-ICP. We compressed the map by randomly downsampling the raw points. Note

TABLE II

DATA SIZE COMPARISON AND THE CORRESPONDING AVERAGE MAP SIZES. THE METHODS WITH HIGHER DATA DIMENSION CAN AFFORD LESS N_x.

	unit	dim.	N_x		
Raw point	# points	3	100	1000	5000
Feature point	# point	35	9	86	429
GMMs	# weighted Gauss.	10	30	300	1500
KITTI	$bytes/m^2$	-	0.58	5.82	29.12
Argoverse	$bytes/m^2$	-	0.52	5.16	25.8

[1]ICP (pt2pt)represents point-to-point ICP

[2]ICP (pt2pl)represents point-to-plane ICP, which requires normal input.

[3]We evaluated the FilterReg version with fixed covariance and equal weights [24], so the data dimension is the same as raw points.

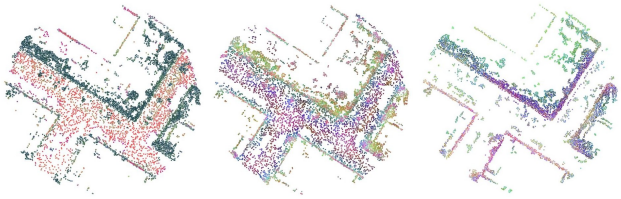
[4]DeepGMR uses weighted isotropic GMM formulation [25]

[5]This is the dimension of the super points used in LORAX. An additional ICP refinement with the dense raw point cloud is required by LORAX besides the super points [29].

that we computed the normals for GICP and point-to-plane on the raw dense map before downsampling to maintain precision. Overall, the success rate dropped as map size budget decreased. Go-ICP, as a global method, outperforms others by avoiding the local minima, but took at least seconds for a registration [15] thus is less practical. The global optimum found by Go-ICP was also not guaranteed to be the ground truth, especially for the Argoverse, since the point distributions were different in the source and the target point clouds.

B. GMMs

We evaluated the recent HGMR [9] and FilterReg [24] as the representatives of the GMM-based methods. We found that HGMR with a level-2 tree outperformed all the other evaluated methods significantly in the Argoverse when the map size is small. Given a fixed map size budget, the Gaussian model by HGMR is much smaller than the descriptors used by feature point based methods and thus allows more components as shown in Table II. Therefore, it had a better spatial coverage than the feature point methods that used sparse feature point map. We also observed that increasing the tree size for HGMR did not lead to better registration results, as also shown in [9] for LiDAR datasets.



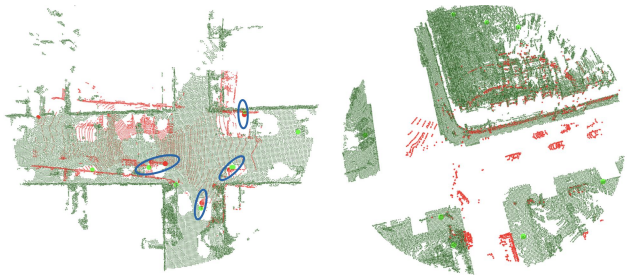
(a) FPFH (b) FCGF (c) D3Feat

Fig. 4. Cropped maps of feature point based methods. The features were projected to three-dimensional space and visualized by the RGB colors. This visualization was downsampled to contain 10000 points so that the difference between random downsampling (FPFH and FCGF) and score-based downsampling (D3Feat) can be observed.

C. Feature Points

We considered feature point based methods together with robust registration algorithms, RANSAC and TEASER++ [6]. We first searched for the correspondence candidates by the descriptor matching and then filtered out noisy correspondences with RANSAC or TEASER++. In our experiments, the deep features, D3Feat [4] and FCGF [20], when used with either RANSAC or TEASER++, significantly outperformed the hand-crafted FPFH [18] in both datasets. For D3Feat, we downsampled the maps to fit the map size budgets by selecting points with higher learned scores. For FCGF and FPFH, we randomly downsampled the maps. A visualization of the downsampled maps for FPFH, FCGF, and D3Feat is shown in Figure 4.

We observed that overall TEASER++ outperformed RANSAC. In addition, the success rates of FCGF and D3Feat in Argoverse were much worse than KITTI with smaller maps. As visualized in Figure 5, the local cropped maps in KITTI overlap better with the LiDAR scans than in Argoverse, because the latter contains large regions that are invisible to the LiDAR scan. A severely downsampled feature point based map in Argoverse is more unlikely to overlap with the LiDAR scan and leads to lower success rate than the GMM-based method which covers more environment under the same map size budget.



(a) KITTI Odometry Dataset (b) Argoverse Tracking Dataset

Fig. 5. An extremely downsampled case to show the differences between two datasets. The cropped maps are in dark green, the LiDAR scans are in red, and the downsampled maps are the light-green dots. (a) Many LiDAR scans overlap well with the cropped map and the FCGF (TEASER++) method can successfully find good correspondences (blue) in this case. The LiDAR scan is shown with the estimated pose here. (b) The LiDAR scan only overlaps by relatively smaller regions, and the downsampled cropped map does not overlap with the LiDAR scan well. This is common in Argoverse. The LiDAR scan is shown with the ground truth pose.

VII. Discussion

In our experiments, we used a cropped map as the target and a LiDAR scan as the source. For the asymmetric methods, switching the source and the target might impact the results. Among the asymmetric methods we evaluated, we used the normals computed from the map for GICP and point-to-plane ICP because the map is much denser and less noisy. For HGMR, we build a GMM tree on the map because the compression ratio of GMM trees is more significant than downsampling the raw points. We also swapped the source and target for FilterReg and found it performing worse than the current configuration.

For the point based methods, we randomly downsampled the points if a score is not provided by the registration method. It is also possible to apply other downsampling techniques together with the evaluated methods to improve the performance. For example, Yin *et al.* [5] used point repeatibility to prune the raw map for ICP.

Our results on feature point based methods suggest that, both the feature extraction and the correspondence filtering methods are crucial for the final performance. TEASER++ [6] in general outperforms the classical RANSAC, but works best only when used together with the deep descriptors under our map size budget.

We also notice that the deep learning based methods failed when map size is small mostly because the target map is feature point based and too sparse after intensive downsampling. A deep shape model based method might potentially increase the spatial coverage of the downsampled map and improve the performance. Further reducing the feature dimension without sacraficing the global feature matching accuracy is also worth more research.

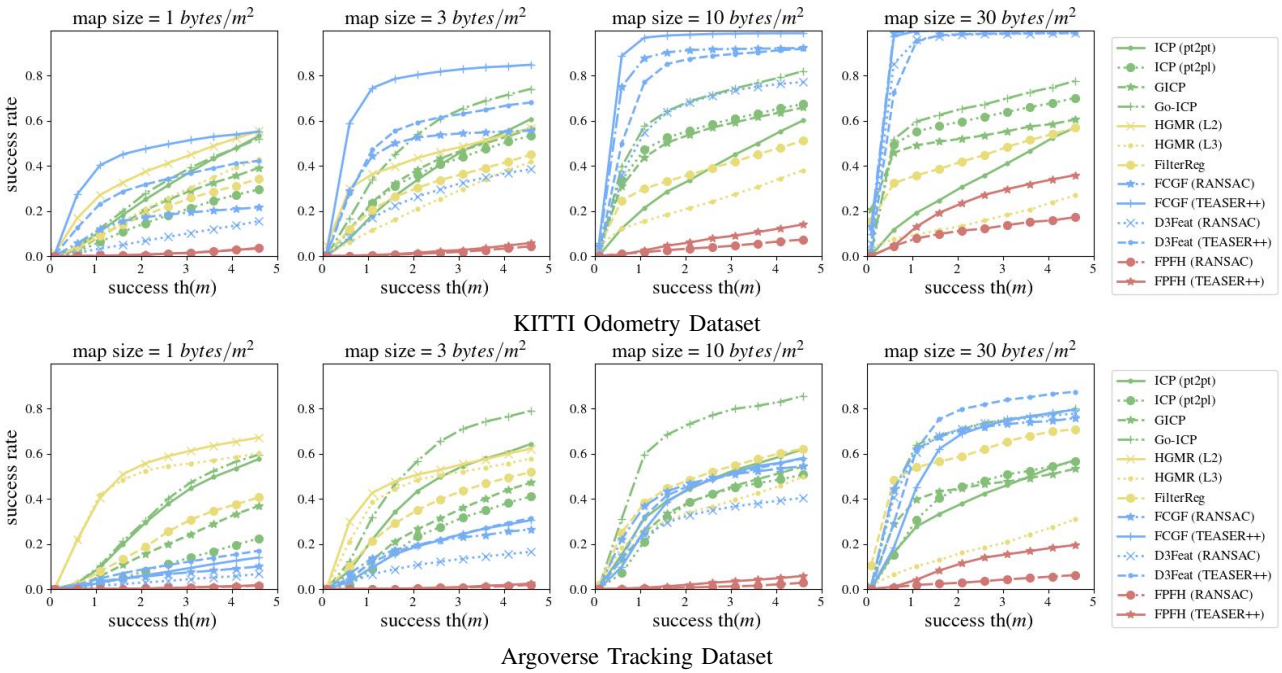


Fig. 6. The success rate curves. We observed that FCGF (TEASER++) and HGMR (L2) outperformed all other methods in the case of map size = 1 $byte/m^2$ in KITTI and Argoverse.

TABLE III

RESULTS ON THE KITTI ODOMETRY DATASET. OVERALL FCGF (TEASER++) AND D3FEAT (TEASER++) OUTPERFORMED ALL OTHER METHODS IN ROBUSTNESS UNDER ALL MAP SIZE BUDGETS.

Metric name	**SR01**	**TE01**	**RE01**	**SR03**	**TE03**	**RE03**	**SR10**	**TE10**	**RE10**	**SR30**	**TE30**	**RE30**
map size($bytes/m^2$)		1			3			10			30	
ICP (pt2pt)	0.19	1.11	2.28	0.31	0.82	1.38	0.30	0.74	1.22	0.27	0.76	1.12
ICP (pt2pl)	0.10	1.23	2.62	0.45	0.67	1.39	0.53	0.48	1.03	0.60	0.21	0.56
GICP	0.15	1.17	2.61	0.47	0.71	1.17	0.53	0.50	0.77	0.49	**0.09**	0.20
Go-ICP	0.22	1.23	2.66	0.62	0.68	1.34	0.67	0.51	1.02	0.64	0.27	0.60
HGMR (L2)	0.34	0.82	**1.37**	0.33	**0.22**	**0.34**	-	-	-	-	-	-
HGMR (L3)	0.16	1.01	1.84	0.18	0.73	1.28	0.19	0.39	0.59	0.09	0.28	0.55
FilterReg	0.15	1.15	1.92	0.31	0.50	0.93	0.34	0.32	**0.60**	0.42	0.11	**0.07**
FCGF (RANSAC)	0.11	0.81	2.89	0.70	0.45	1.84	0.87	0.34	1.38	**1.00**	0.19	0.74
D3Feat (RANSAC)	0.02	0.80	3.09	0.24	0.59	2.76	0.46	0.49	2.34	0.97	0.23	1.00
FPFH (RANSAC)	0.00	-	-	0.00	-	-	0.01	0.59	3.44	0.11	0.46	2.63
FCGF (TEASER++)	**0.38**	**0.48**	1.94	**0.86**	0.33	1.54	**0.95**	**0.28**	1.32	**1.00**	0.20	0.92
D3Feat (TEASER++)	0.23	0.62	1.98	0.65	0.56	1.79	0.79	0.51	1.63	0.97	0.40	1.09
FPFH (TEASER++)	0.00	-	-	0.00	-	-	0.01	1.18	3.38	0.30	0.93	2.54

VIII. CONCLUSION

In this benchmark, we consider single-frame LiDAR-to-map registration, which is an important module in a complex autonomous driving system. In real world conditions, autonomous vehicle systems include complicated interaction between multiple module and sensor modalities. Performance evaluation in this scenario remains an open research question. Our potential future works include the evaluation of multi-modal registration algorithms such as registering an image to the LiDAR map.

ACKNOWLEDGEMENTS

This work was supported by the CMU Argo AI Center for Autonomous Vehicle Research. We also thank our labmates for the valuable suggestions to improve this paper.

REFERENCES

[1] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

TABLE IV

RESULTS ON THE ARGOVERSE TRACKING DATASET. HGMR (L2) AND HGMR (L3) OUTPERFORMED ALL THE OTHER METHODS SIGNIFICANTLY WHEN MAP SIZE IS SMALL.

Metric name	**SR01**	**TE01**	**RE01**	**SR03**	**TE03**	**RE03**	**SR10**	**TE10**	**RE10**	**SR30**	**TE30**	**RE30**
map size($bytes/m^2$)		1			3			10			30	
ICP (pt2pt)	0.19	1.26	3.57	0.39	0.77	2.17	0.39	0.73	1.96	0.33	0.63	1.36
ICP (pt2pl)	0.03	1.20	3.29	0.29	1.18	3.10	0.33	1.03	2.62	0.43	0.72	1.85
GICP	0.09	1.10	3.40	0.29	0.93	2.36	0.34	0.78	2.07	0.42	0.43	1.07
Go-ICP	0.19	1.30	3.39	**0.62**	0.68	1.34	**0.67**	0.51	**1.02**	0.64	0.27	0.60
HGMR (L2)	**0.54**	0.74	1.70	0.40	**0.42**	**0.89**	-	-	-	-	-	-
HGMR (L3)	0.51	**0.67**	**1.64**	0.42	0.58	1.38	0.32	**0.47**	1.05	0.09	0.73	1.41
FilterReg	0.13	1.24	3.14	0.41	0.62	1.73	0.45	0.51	1.49	0.63	**0.15**	**0.33**
FCGF (RANSAC)	0.03	0.71	2.28	0.26	0.66	2.31	0.38	0.59	2.13	0.69	0.43	1.62
D3Feat (RANSAC)	0.01	0.79	1.76	0.11	0.59	2.05	0.21	0.55	1.99	0.80	0.38	1.45
FPFH (RANSAC)	0.00	-	-	0.00	-	-	0.00	-	-	0.03	0.57	3.10
FCGF (TEASER++)	0.03	0.93	2.97	0.24	0.96	2.66	0.36	0.93	2.51	0.69	0.82	2.08
D3Feat (TEASER++)	0.05	0.82	2.44	0.22	0.75	2.50	0.38	0.75	2.32	**0.86**	0.72	1.82
FPFH (TEASER++)	0.00	-	-	0.00	-	-	0.01	1.26	3.47	0.12	1.14	2.78

[2] J. Martinez, S. Doubov, J. Fan, I. A. Bârsan, S. Wang, G. Máttyus, and R. Urtasun, "Pit30m: A benchmark for global localization in the age of self-driving cars," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[3] C. Cao, M. Preda, and T. Zaharia, "3D point cloud compression: A survey," in *Proc. of ACM International Conf. on 3D Web Technology (Web3D)*, 2019.

[4] X. Bai, Z. Luo, L. Zhou, H. Fu, L. Quan, and C. L. Tai, "D3Feat: Joint learning of dense detection and description of 3D local features," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[5] H. Yin, Y. Wang, L. Tang, X. Ding, S. Huang, and R. Xiong, "3D lidar map compression for efficient localization on resource constrained vehicles," *IEEE Intelligent Transportation Systems Conference (ITSC)*, vol. 22, no. 2, pp. 837 – 852, 2021.

[6] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and certifiable point cloud registration," *IEEE Trans. on Robotics (TRO*, pp. 1–20, 2020.

[7] R. Schnabel and R. Klein, "Octree-based point-cloud compression," *Eurographics Symposium on Point-Based Graphics*, 2006.

[8] "bzip2," in *http://www.bzip.org/*.

[9] B. Eckart, K. Kim, and J. Kautz, "HGMR: Hierarchical gaussian mixtures for adaptive 3D registration," in *Proc. Eur. Conf. on Computer Vision (ECCV)*, 2018.

[10] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 14, no. 2, pp. 239–256, 1992.

[11] M. Greenspan and M. Yurick, "Approximate k-d tree search for efficient ICP," in *Proc. International Conf. 3-D Digital Imaging and Modeling (3DIM)*, 2003.

[12] A. V. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Proc. Robotics: Science and Systems (RSS)*, 2009.

[13] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[14] Q. Y. Zhou, J. Park, and V. Koltun, *Open3D: A modern library for 3D data processing*. arXiv:1801.09847, 2018.

[15] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-ICP: A globally optimal solution to 3D icp point-set registration," *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 38, no. 11, pp. 2241–2254, 2016.

[16] R. Dubé, A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart, and C. Cadena, "SegMap: Segment-based mapping and localization using data-driven descriptors," in *Proc. Robotics: Science and Systems (RSS)*, 2018.

[17] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[18] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3d registration," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2009.

[19] Y. Wang and J. Solomon, "Deep Closest Point: Learning representations for point cloud registration," in *Proc. Intl. Conf. on Computer Vision (ICCV)*, 2019.

[20] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *Proc. Intl. Conf. on Computer Vision (ICCV)*, 2019.

[21] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. Intl. Conf. on Computer Vision (ICCV)*, 2019.

[22] A. Myronenko and X. Song, "Point set registration: Coherent point drifts," *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 32, no. 12, pp. 2262–2275, 2010.

[23] P. Biber, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2003.

[24] W. Gao and R. Tedrake, "FilterReg: Robust and efficient probabilistic point-set registration using gaussian filter and twist parameterization," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[25] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "DeepGMR: Learning latent gaussian mixture models for registration," in *Proc. Eur. Conf. on Computer Vision (ECCV)*, 2020.

[26] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "PointnetLK: Robust & efficient point cloud registration using PointNet," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[27] H. Wang, C. Wang, and L. Xie, "Intensity-slam: Intensity assisted localization and mapping for large scale environment," in *IEEE Robotics and Automation Letters (RA-L)*, 2021.

[28] C. Yang and G. Medioni, "Object modelling by registration of multiple range images," *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 1992.

[29] G. Elbaz, T. Avraham, and A. Fischer, "3D point cloud registration for localization using a deep neural network auto-encoder," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[30] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[31] M. F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, "Argoverse: 3D tracking and forecasting with rich maps," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[32] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard, "Autonomous robot navigation in highly populated pedestrian zones," *Journal of Field Robotics*, vol. 32, no. 4, 2015.

[33] Z. J. Yew and G. H. Lee, "3DFeat-Net: Weakly supervised local 3D features for point cloud registration," in *Proc. Eur. Conf. on Computer Vision (ECCV)*, 2018.