LiDAR-Based Localization on Highways Using Raw Data and Pole-Like Object Features

Sheng-Cheng Lee[1,2], Victor Lu[1], Chieh-Chih Wang[1,2], and Wen-Chieh Lin[2]

[1]Industrial Technology Research Institute, Taiwan
[2]National Yang Ming Chiao Tung University, Taiwan

Abstract

Poles on highways provide important cues for how a scan should be localized onto a map. However existing point cloud scan matching algorithms do not fully leverage such cues, leading to suboptimal matching accuracy in highway environments. To improve the ability to match in such scenarios, we include pole-like objects for lateral information and add this information to the current matching algorithm. First, we classify the points from the LiDAR sensor using the Random Forests classifier to find the points that represent poles. Each detected pole point will then generate a residual by the distance to the nearest pole in map. The pole residuals are later optimized along with the point-to-distribution residuals proposed in the normal distributions transform (NDT) using a nonlinear least squares optimization to get the localization result. Compared to the baseline (NDT), our proposed method obtains a 34% improvement in accuracy on highway scenes in the localization problem. In addition, our experiment shows that the convergence area is significantly enlarged, increasing the usability of the self-driving car localization algorithm on highway scenarios.

1. Introduction

Reliable and accurate pose estimation is a basic but critical problem for a self-driving vehicle. A reliable localization gives the vehicle its state with respect to map coordinates, helping it navigate with road information and plan the paths to the destinations. A more accurate localization result enables the self-driving car to follow the lanes of the road, make control decisions, detect obstacles, and perform other tasks.

LiDAR-based localization often needs to match a given scan onto a point cloud map. We can use one of several general-purpose point cloud matching methods including iterative closest points (ICP) [1, 4, 10, 13] and normal distribution transform (NDT) [2, 9, 14, 15], which are especially effective in urban environments where the abundance of geometric features in buildings help guide and anchor the scan into the correct map location. However, highway environments are usually more monotonous, containing many repeating objects and few prominent features. While there still are many road side features, such as poles, that provide

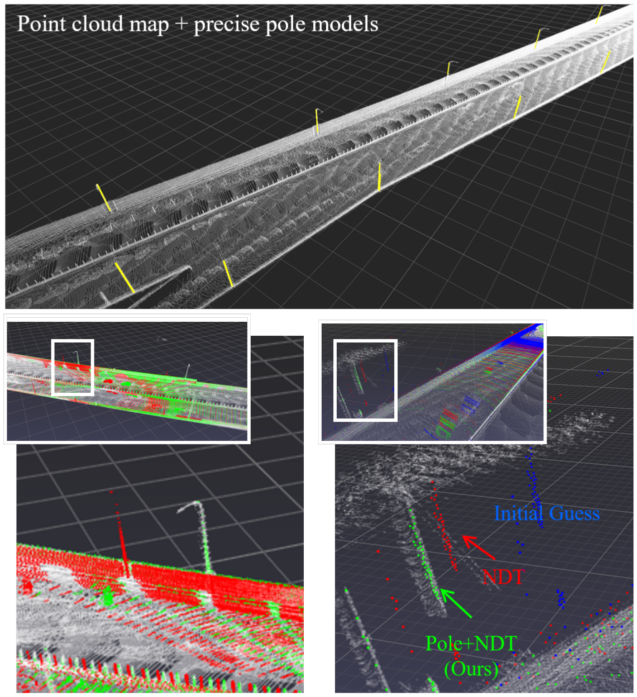


Figure 1. Point cloud map (raw data map) augmented with precise 3D pole models (top). By explicitly aligning pole points to pole models, our method (green) successfully localizes scans in the challenging case of highway with few distinctive features, achieving a noticeably better alignment than the existing general-purpose 3D scan matching method of NDT [9] (red).

important cues on how a given scan should localize, these features are often too small or too few in number of points for general-purpose methods to work well on.

Several papers [5, 6, 11, 12, 17] try to use poles for localization. However, these methods can only work in environments that contain poles. If there are no poles or if they did not detect poles, these methods have nothing to match against or nothing to use to match.

In this paper, we proposed a method to take poles as important features but, at the same time, still retains the effect of other scan details. The matching cost is composed of the point-to-distribution cost as in NDT [2, 9] and a new point-to-pole cost. This approach not only helps us achieve a better localization result, but also works in areas with no poles present.

2. Related Work

There are many variants of ICP, including Besl and McKay's original point-to-point formulation [1] and point-to-plane ICP [4]. Despite the matching accuracy of these ICP variants, its computational cost is relatively high, and is not the best choice for localization in self-driving vehicles.

NDT [9] uses a collection of normal distributions to represent a point cloud map. In doing so, storage cost is greatly reduced. Furthermore, NDT is known to be relatively fast. However, finer geometric details tend to get lost as a single normal distribution tries to fit to a cell with potentially many different objects. It is possible to adopt weights for different objects. Hong et al. [7] proposed a probabilistic weighted version of normal distributions transform (PW-NDT). In this work, cells are estimated as weighted normal distribution functions considering the weight of each point from the target point cloud. Point-to-distribution costs are also weighted according to the probabilities of the source points. In our experiments, the accuracy of scan matching is increased. However, while we could enhance pole weight in all distributions and pole points, some features may be considered as outliers if the initial pose is too far from ground truth, making the pole points outside of the corresponding voxel. In this paper, we solve this problem by individually associating pole features and raw data.

There exist many other studies that attempt to use poles as matching features. Most of them differ in the aspect of detecting poles. For example, the detector proposed in [17] first voxelizes a 3-D scan and counts the points presented in each voxel. According to their assumption, the poles are located in contiguous vertical stacks. Sefati et al. [12] include stereo cameras to assist in detection. The laser points are first filtered to remove the ground points. The remaining points are clustered by height and occupancy and are fit to cylinder models. Schaefer et al. [11] extract pole landmarks with an occupancy map and a pole feature extractor according to the definition of a pole as a vertical stack of occupied voxels and laterally surrounded by free voxels. Instead of using raw points, Dong et al. [5] exploit a detection method on range images. They first project 3D points to a range image according to their azimuths and elevation angles. Then, they put all pixels into clusters by depths. The poles are then detected from each cluster with some heuristics. One is that a pole cluster usually has a large aspect ratio, where its height is larger than its width. Another heuristic is that a pole usually has a significant depth difference between the pixels in its neighborhoods.

3. Proposed Method

3.1. Background

In this work, we solve the 3D registration problem, seeking a rigid transform aligning a scan against a map. We specify a rigid transform $T_\Theta : \mathbb{R}^3 \to \mathbb{R}^3$ by parameters

$$\Theta = [x, y, z, \alpha, \beta, \gamma] \tag{1}$$

where x, y, z form a translation vector, and α, β, γ represent rotation Euler angles. Given a point $\vec{p} \in \mathbb{R}^3$ we compute $T_\Theta(\vec{p})$ as follows.

$$T_\Theta(\vec{p}) = R\vec{p} + \vec{t} \tag{2}$$

$$R = R_z(\gamma)R_y(\beta)R_x(\alpha) \tag{3}$$

$$\vec{t} = [x, y, z]^T \tag{4}$$

3.2. Joint Optimization

The main contribution of this work is a cost that can be optimized jointly with the original cost of P2D-NDT. The purpose of this design is to continue to use NDT's representation of scene geometry, while explicitly enforcing alignment of pole points to pole models, where available. This way, we maintain acceptable performance even when there are no poles present in the environment.

We choose to represent each pole using a truncated cone model, considering how poles are in reality often tapered and rarely perfectly vertical. The model for the k-th pole consists of a base center point $\vec{c}_k$, a base radius r_k, a unit direction vector $\hat{d}_k$ along the pole axis and a taper rate s_k measuring the change in radius for each meter in height. Compared to using a more simplistic 2D representation of pole positions as in previous methods, our use of a more precise pole model can help bring scans into closer alignment with the map.

We associate a pole point with a pole model by searching for the nearest pole axis within a certain distance threshold. In practice, this was implemented by range querying a k-d tree that stored point samplings of all pole axes.

Given a point $\vec{p}_i$ associated with the k-th pole, and denoting the point's transformed position by $T_\Theta(\vec{p}_i)$, we define the cost $e_{\text{pole},i}$ as the transformed position's orthogonal

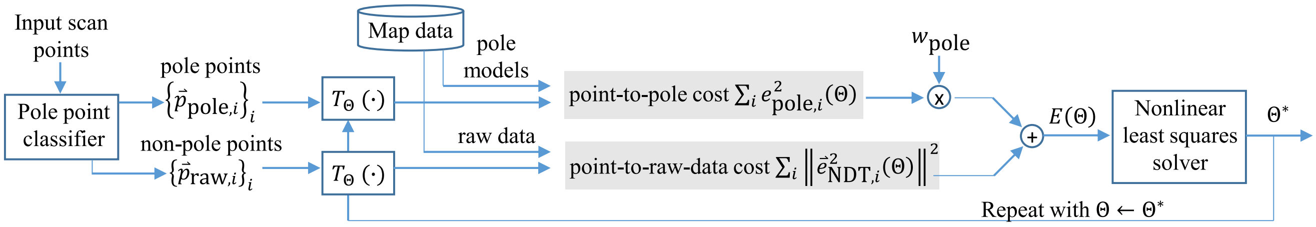

Input scan points
Pole point classifier
pole points
$\{\vec{p}_{\text{pole},i}\}_i$
non-pole points
$\{\vec{p}_{\text{raw},i}\}_i$
Map data
pole models
raw data
$T_\Theta(\cdot)$
$T_\Theta(\cdot)$
point-to-pole cost $\sum_i e^2_{\text{pole},i}(\Theta)$
point-to-raw-data cost $\sum_i \left\|\vec{e}^2_{\text{NDT},i}(\Theta)\right\|^2$
w_{pole}
x
+
$E(\Theta)$
Nonlinear least squares solver
Θ^*
Repeat with $\Theta \leftarrow \Theta^*$

Figure 2. Overview of our method.

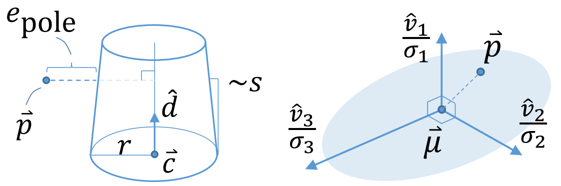

e_{pole}
$\vec{p}$
$\hat{d}$
r
$\vec{c}$
$\sim s$
$\frac{\hat{v}_1}{\sigma_1}$
$\vec{p}$
$\frac{\hat{v}_3}{\sigma_3}$
$\vec{\mu}$
$\frac{\hat{v}_2}{\sigma_2}$

Figure 3. Our method simultaneously aligns scans against precise pole models (left) and the normal distribution "blobs" used by NDT (right). Each point that is associated with a pole incurs a cost e_{pole} (Eq. 5) that measures the point's deviation from the pole's surface, and likewise $\vec{e}_{\text{NDT}}$ for blobs.

distance to the pole axis subtracted by the pole radius at the position of orthogonal projection on the pole axis (Fig 3). Reducing $e_{\text{pole},i}$ to 0, in effect brings $\vec{p}_i$ into alignment with the pole's surface.

$$\begin{aligned} e_{\text{pole},i}(\Theta) = {} & \|\vec{\triangle}_\Theta - (\vec{\triangle}_\Theta \cdot \hat{d}_k)\hat{d}_k\| \\ & - (r_k + s_k \vec{\triangle}_\Theta \cdot \hat{d}_k) \end{aligned} \tag{5}$$

$$\vec{\triangle}_\Theta := T_\Theta(\vec{p}_i) - \vec{c}_k \tag{6}$$

NDT scan matching aligns scan points against raw data points (*i.e.* a point cloud map) by *maximizing* the transformed point's likelihood evaluated on a collection of normal distributions (fitted to the raw data points). In order to reuse this collection of normal distributions in a nonlinear least squares framework, we reformulate NDT scan matching as *minimizing* the transformed point's Mahalanobis distance to the normal distributions. Given a transformed scan point $T_\Theta(\vec{p}_i)$ and its associated normal distribution $\mathcal{N}(\vec{\mu}, C)$ where C has eigenvalue decomposition $C = V\Lambda V^T$, we compute their squared distance $\|\vec{e}_{\text{NDT},i}\|^2$ by

$$\begin{aligned} \|\vec{e}_{\text{NDT},i}(\Theta)\|^2 &= (T_\Theta(\vec{p}_i) - \vec{\mu})^T C^{-1} (T_\Theta(\vec{p}_i) - \vec{\mu}) && (7) \\ &= \|W(T_\Theta(\vec{p}_i) - \vec{\mu})\|^2 && (8) \end{aligned}$$

$$W := \Lambda^{-\frac{1}{2}} V^T \tag{9}$$

Since nonlinear least squares solvers, *e.g.* Ceres, expect pre-squared cost functions, what we actually provide to the solver is just $\vec{e}_{\text{NDT},i}$.

$$\vec{e}_{\text{NDT},i}(\Theta) = W(T_\Theta(\vec{p}_i) - \vec{\mu}) \tag{10}$$

The components of the vector $\vec{e}_{\text{NDT},i}$ measure deviations from $\vec{\mu}$ along C's eigenvector directions $\vec{v}_1, \vec{v}_2, \vec{v}_3$ divided by the distribution's standard deviations $\sigma_1, \sigma_2, \sigma_3$ (Fig. 3). It is easy to see how this assigns the desired costs, for example, in the case of a planar surface, where deviations tangent to the surface are divided by a large standard deviation, and hence do not cost much.

To jointly optimize two sets of costs, we can add a weight parameter w_{pole} to adjust the effect of the residuals of the pole. Here, we fix the weight of the NDT cost to 1 and the weight of pole cost as a variable. The effect of the value of this variable will be discussed in Section 4.1. Finally, we write the combined cost function $E(\Theta)$ as

$$E(\Theta) = w_{\text{pole}} \sum_i e^2_{\text{pole},i}(\Theta) + \sum_i \|\vec{e}_{\text{NDT},i}(\Theta)\|^2 \tag{11}$$

$$\Theta^* = \text{argmin}_\Theta E(\Theta) \tag{12}$$

3.3. Pole Point Classification

To detect pole points in a scan, we assume scan points from a given laser have been ordered by measurement time. We then convert this sequence of scan points into a sequence of line *segments* (Fig. 4), by fitting piecewise lines using a dynamic programming algorithm similar to [8]. We perform per-segment prediction using a 2-stage random forest classifier (Fig. 5), and then assign the segment's pole probability to each of its constituent scan points. Predicting on a per-segment basis (vs. per-point) has efficiency benefits, while the slight loss in per-point accuracy so far seems acceptable (most segments produced by our method are either all pole points or all non-pole points). Working with segments also opens up opportunities for handcrafting a variety of machine learning features, such as those listed in Table 1.

The first stage of our classifier focuses on *intra*-sequence reasoning, computing a set of *base* features (*e.g.* length, angle between current and next segments) meant for characterizing the "shape" of the scan point sequences in the neighborhood of a given segment, The second stage elevates its reasoning to the *inter*-sequence level, computing

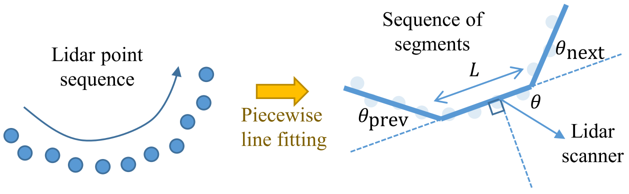


Figure 4. Sequence of scan points are converted into a sequence of line segments, and handcrafted features computed on these segments.

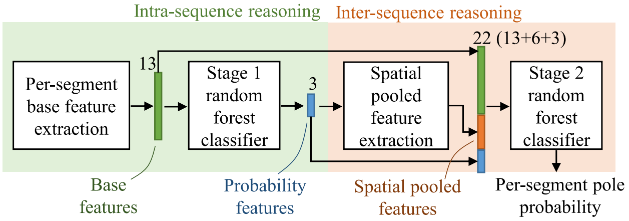

Intra-sequence reasoning
Inter-sequence reasoning
Per-segment base feature extraction
13
Stage 1 random forest classifier
3
Spatial pooled feature extraction
22 (13+6+3)
Stage 2 random forest classifier
Base features
Probability features
Spatial pooled features
Per-segment pole probability

Figure 5. Our 2-stage random forest classifier for predicting per-segment pole probability. The first stage takes the base features with local geometry of a line segment and the second stage take consideration of a pooling area.

pooled features *i.e.* statistics on spatially nearby segments, including segments from other laser sequences. Table 1 contains a detailed listing of the features computed. The final per-segment prediction is computed on the concatenation of base features, probability features, and pooled features.

4. Experiments

Test data. We use sensor data from a prototype self-driving car developed by the Mechanical and Mechatronics Systems Laboratory (MMSL), Industrial Technology Research Institute (ITRI). This includes data from LiDAR, IMU, GNSS and wheel odometer. The data was collected in a drive through parts of Hsinchu, Taiwan, including both urban and highway areas (Fig. 7).

Map data. The point cloud map used here was produced from LiDAR data collected in a separate trip than the test data, again using tools developed MMSL, ITRI. The pole models were obtained by identifying pole points using the classifier in Sec 3.3, fitting poles to the identified points using the method of [8], and then manually removing false positive poles. The top part of Fig. 1 shows in small section of our map data.

Baseline methods. We compare our method with original NDT [9] and PW-NDT [7]. To isolate the effect of adding our pole residuals, we implement NDT and PW-NDT also as a minimization of the Mahalanobis distance in Eq. 7. For PW-NDT, we magnify the residual vector in Eq. 10 specifically for scan points that have been classified as pole points, and we magnify the contribution of map pole points when preparing the map normal distributions, which in effect causes the distribution to better fit to any poles in the distribution's cell. We chose this magnification scalar to be 3.

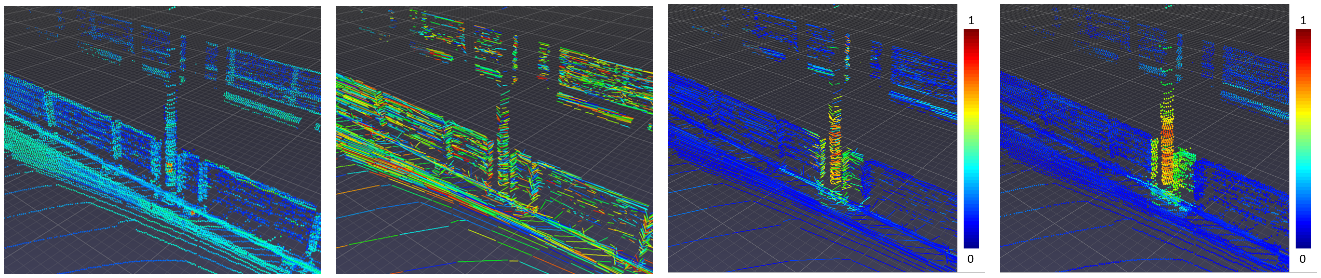
Table 1. Per-segment features used in two-stage random forest classifier.

Feature	Description
Base features	
d	Distance from current segment center to Lidar scanner
θ	Angle between line of sight and current segment
θ_{prev}	Angle between current and prev segment
θ_{next}	Angle between current and next segment
L	Length of current segment
L_{prev}	Total length of smoothly connected prev segments
L_{next}	Total length of smoothly connected next segments
n	Number of points in the current segment
n_{prev}	Total # points in smoothly connected prev segments
n_{next}	Total # points in smoothly connected next segments
ϵ	Fitting error of current segment
ϵ_{prev}	Fitting error of prev segment
ϵ_{next}	Fitting error of next segment
Probability features (first stage predictions)	
ρ	First stage pole probability of current segment
ρ_{prev}	First stage pole probability of prev segment
ρ_{next}	First stage pole probability of next segment
Spatial pooled features *these features are recomputed on multiple pooling domains*	
N_{ring}	Number of rings in pooling domain
N_{points}	Number of points in pooling domain
ρ_{max}	Max first stage pole probability in pooling domain
ρ_{mean}	Mean first stage pole probability in pooling domain
z_{offset}	(segment centroid z) - $(z_{\text{max}} + z_{\text{min}})/2$
z_{range}	$z_{\text{max}} - z_{\text{min}}$

For all methods, we used identical parameters in distribution estimation and optimization: we use a 10cm voxel grid filter to down sample scan points, we use a 2.5m cell size when preparing map normal distributions, we perform up to 20 iterations of the method's outer loop (*i.e.* as in Fig. 2), and we set nonlinear least square solver's maximum number of function evaluations to 100. For the nonlinear least squares solver, we use scipy's implementation of the trust region reflective algorithm [3].

4.1. Weights Under Different Scenes

We first tested our matching method with both urban and highway scenarios. To ensure that the result shows the right effect of pole weights, we choose the scans that actually have poles in the environments. Figure 8 shows different improvements in translation error when using different pole weights. The X-axis is the weight of the pole residuals compared to the NDT residuals chosen from 0 to 10. Therefore,



(a) Scan points (b) Line segments (c) Per-segment prediction (d) Per-point prediction

Figure 6. Pole point classification steps. The confidence score of the Random forests classifier is shown with colors in the third and forth figures, where red represents highest confidence and blue is the lowest confidence. The color coding bar is shown beside the figures.

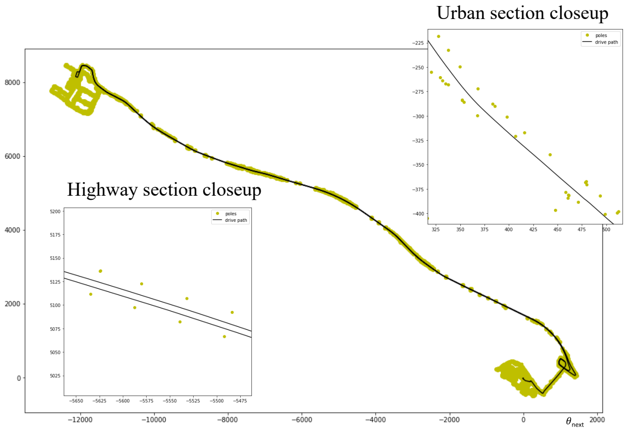


Figure 7. Path of the testing dataset. Black lines are the path of the vehicle and yellow dots are pole centers in maps.

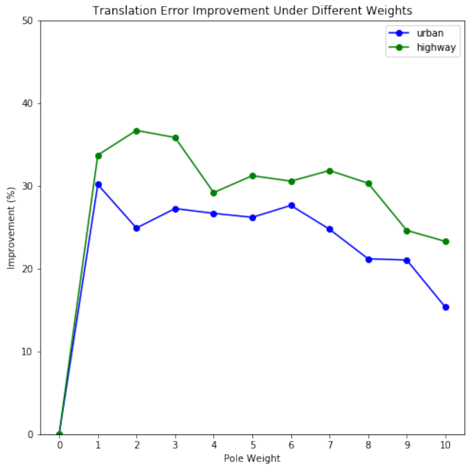


Figure 8. Translation error improvement rates under different pole weights, assuming there is at least one pole in the test environment.

zero-weight equivalents to pure NDT without adding pole constraints. Improvement rates are the percentage of error reduction compared to the zero-weight result. On average, the result shows that we can obtain a better matching accuracy by adding pole residuals to the optimization. Additionally, we need a heavier weight on the highway scene than on the urban scene. To be more specific, we could choose 2 as the weight of pole residuals in highway environments, while 1 is a better choice for urban environments. Meanwhile, the improvement weight is always higher in the highway area if the same weight is used. That is, we can obtain more accuracy of the localization on the highway than in the urban scene by increasing the characteristic of the poles.

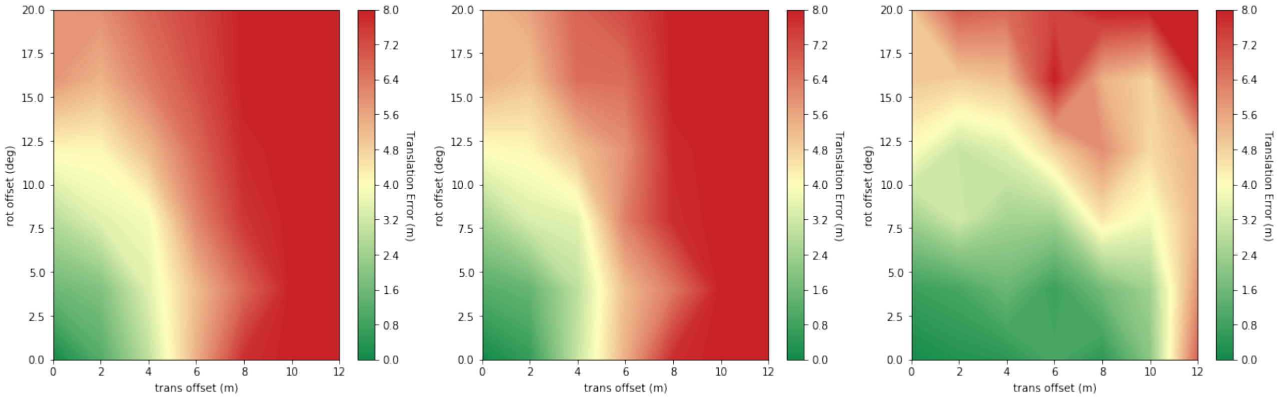
4.2. Convergence Area

In this experiment, a set of point clouds collected from our vehicle with ground-truth poses is used. We selected 32 scans and checked their ground-truth poses to make sure that we had correct answers. We applied translations and rotations with different ranges to the ground-truth poses of the scans as initial guesses. The translation lengths range from 0 to 12 meters with randomly chosen directions. Rotation angles range from 0 to 20 degrees in random directions. We compare three different methods: NDT [9], PW-NDT [7], and our method.

Figure 9 shows the final translation error of the aligned scans. The result shows that our proposed method has a larger converging area compared to NDT and PW-NDT. Also, we can see that the improvement in the translation offset is greater than in the rotation offset. That is, our method is more robust for an initial guess with error than the other two methods, especially in translation. This could be the effect of the separated association of the pole features and raw data.

4.3. Relocalization With Pole Map

We tested our method on drive scan sequences to obtain re-localization results. In this experiment, we used GNSS measurements as initial guesses. In this experiment, two different urban and highway environments are included. The re-localization error is calculated with the ground truth obtained by an Extended Kalman filter (EKF) [16] fusing



(a) NDT (b) PW-NDT (c) Ours

Figure 9. Converge plot of different translation and rotation.

the wheel odometry, IMU, GPS, and the matching result provided by MMSL, ITRI. Table 2 and Table 3 show the matching error of NDT, PW-NDT and our method.

In Table 2, we can see that although PW-NDT gets some improvement, our method gets a more significant improvement on the localization of the highway. The translation error is 35% less than the original NDT. As for the rotation error, there isn't much improvement, as it is already small enough. On average, our method gives a slightly larger error in rotation, but the variance is smaller, which means the rotation error range is more stable.

Meanwhile, in Table 3, the improvement is not obvious in both PW-NDT and our method. We believe it is because there are already enough features for the original NDT to match the two scans. Although the original NDT can match urban scans well without the help of pole features, we can still see that the variance in both translation and rotation error is much lower when using our matching method.

4.4. Simultaneous Localization And Mapping (SLAM)

In this experiment, we try to estimate vehicle poses and build the pole map at the same time. Since our pole modeling method requires stacking several scans to make a more precise pole estimation, there are no poles at the beginning. Therefore, the module acts as the original normal distributions transform matching until enough pole models are detected. Once the poles are presented in the environment, we can utilize pole residuals and keep updating the pole model map. At the end of the run, we get a pose sequence and a pole model map built by accumulating the circle centers detected from each scan and fitting them with our pole fitting method.

The relative pose error of the test result is shown in Table 4. In this table, the errors are calculated by comparing the matched poses with the relative pose of the current scan and the previous scan in ground-truth poses. We can see that on the highway areas, our method has a better performance in both translation and rotation error. On the other hand, we get similar results in the urban areas. This could be the same reason as in Section 4.3 that there are more salient features in the urban areas. Also, the imperfect pole model detection will in crease errors under all scenarios.

5. Conclusion and Future Work

In this paper, we proposed a scan matching method to combine raw data and pole features into an optimization function. The method not only enlarges the effect of pole constraints under monotonic scenes, but can also retain the original point-to-distribution NDT characteristics if poles are not presented in the environments. This way, our approach works both on the urban and on the highway, where the latter scenarios are usually defined to be difficult for a generalized scan matching method. While other localization methods which only consider pole features cannot work in an environment with no poles nearby, our method can still get reasonable results. In addition, we developed a pole detection method that first splits scan points into line segments and predicts the probabilities of pole on a segment basis. The detection result is used in our matching method along with the map pole models obtained by our pole model fitting method.

There are several improvements in this work. Our experiment shows that the pole information is important for a self-driving vehicle to perform the localization task. Our proposed method achieves a wider convergence range, making this scan matching more robust to errors on initial guess. Furthermore, the method improves the accuracy of the localization by 36% on highway compared to the original method. Both in highway scenarios and urban scenarios, our method shows its stability on the matching error with much less variance compared to the original NDT and PW-

Table 2. Result of the localization in the highway region

Methods	Translation Error				Rotation Error			
	Mean (m)	Var. (m)	RMSE (m)	Max(m)	Mean (deg)	Var. (deg)	RMSE (deg)	Max (deg)
NDT	0.224	0.501	0.742	10.054	**0.234**	0.001334	0.362	5.008
PW-NDT	0.178	0.449	0.693	9.682	0.235	0.001238	0.355	5.017
Ours	**0.146**	**0.063**	**0.289**	**4.086**	0.277	**0.000693**	**0.342**	**1.792**

Table 3. Result of the localization in the urban region

Methods	Translation Error				Rotation Error			
	Mean (m)	Var. (m)	RMSE (m)	Max(m)	Mean (deg)	Var. (deg)	RMSE (deg)	Max (deg)
NDT	0.167	0.566	0.771	12.038	**0.401**	0.022	1.203	17.810
PW-NDT	**0.164**	0.541	0.753	11.602	**0.401**	0.013	0.974	7.902
Ours	**0.164**	**0.034**	**0.247**	**1.723**	0.458	**0.006**	**0.745**	**4.587**

NDT. Finally, we tested the matching method on a SLAM system. Although we only get a little improvement on the highway, we know that the performance of our module depends on the quality of the pole map. In order to obtain a better SLAM result, we should increase the detection precision and reduce false detection on modeling.

We can consider several approaches to further strengthen our method. We can include more features in road scenes, such as traffic signs and building edges. These features have some common characteristics: they often appear in road scenes, and usually can be simplified to geometrical primitives. Furthermore, we may use a stronger association method to reject the outlier, which may be false detection or even changes on the map. Although the detection method of objects may not be precise enough, outlier rejection is capable of filtering bad associations. A maximum clique technique is suitable for these features. Since we have only geometry primitives to associate with, the computing time may be acceptable. It is also a good idea to adjust the pole weight adaptively in different scenes. As our experiment shows, we should use a lower weight on pole residuals when there are more salient features in the environments. These are some possible ways to improve our current work in the future.

References

[1] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992. 1, 2

[2] P. Biber and W. Strasser. The normal distributions transform: a new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, volume 3, pages 2743–2748 vol.3, 2003. 1, 2

[3] Mary Ann Branch, Thomas F. Coleman, and Yuying Li. A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems. *SIAM Journal on Scientific Computing*, 21(1):1–23, 1999. 4

[4] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pages 2724–2729 vol.3, 1991. 1, 2

[5] Hao Dong, Xieyuanli Chen, and Cyrill Stachniss. Online range image-based pole extractor for long-term lidar localization in urban environments. In *2021 European Conference on Mobile Robots (ECMR)*, pages 1–6, 2021. 2

[6] Hao Dong, Xieyuanli Chen, Simo Särkkä, and Cyrill Stachniss. Online pole segmentation on range images for long-term lidar localization in urban environments, 2022. 2

[7] Hyunki Hong and B. H. Lee. Probabilistic normal distributions transform representation for accurate 3d point cloud registration. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3333–3338, 2017. 2, 4, 5

[8] Victor Lu. Refining the detection and modelling of poles in street lidar by fitting piecewise arcs to the lidar point sequence, https://bit.ly/3evg3z8. 2019. 3, 4

[9] Martin Magnusson. *The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection*. PhD thesis, Örebro universitet, 2009. 1, 2, 4, 5

[10] Szymon Rusinkiewicz. A symmetric objective function for ICP. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 38(4), July 2019. 1

[11] A. Schaefer, D. Büscher, J. Vertens, L. Luft, and W. Burgard. Long-term urban vehicle localization using pole landmarks extracted from 3-d lidar scans. In *2019 European Conference on Mobile Robots (ECMR)*, pages 1–7, 2019. 2

[12] M. Sefati, M. Daum, B. Sondermann, K. D. Kreisköther, and A. Kampker. Improving vehicle localization using semantic and pole-like landmarks. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 13–19, 2017. 2

[13] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009. 1

Table 4. Relative pose error (RPE) of SLAM on highway and urban area.

Method	Urban Trans. Error (m/frame)	 Rot. Error (deg/frame)	Highway Trans. Error (m/frame)	 Rot. Error (deg/frame)
NDT	**0.142**	0.340	0.548	0.261
Ours	0.145	**0.299**	**0.503**	**0.259**

[14] Todor Stoyanov, Martin Magnusson, Henrik Andreasson, and Achim J Lilienthal. Fast and accurate scan registration through minimization of the distance between compact 3d ndt representations. *The International Journal of Robotics Research*, 31(12):1377–1393, 2012. 1

[15] Eijiro Takeuchi and Takashi Tsubouchi. A 3-d scan matching using improved 3-d normal distributions transform for mobile robotic mapping. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3068–3073, 2006. 1

[16] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995. 5

[17] Lihong Weng, Ming Yang, Lindong Guo, Bing Wang, and Chunxiang Wang. Pole-based real-time localization for autonomous driving in congested urban scenarios. In *2018 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 96–101, 2018. 2