Pole-Curb Fusion Based Robust and Efficient Autonomous Vehicle Localization System With Branch-and-Bound Global Optimization and Local Grid Map Method

Guang Chen, Fan Lu, Zhijun Li, Yinlong Liu, Jinhu Dong, Junqiao Zhao, Junwei Yu, and Alois Knoll

***Abstract*—The localization system is one of the key components of autonomous vehicles. Widely-used GNSS (low-cost) alone can not meet the centimeter accuracy and is unstable in urban environment. Typically LiDAR-based localization system is very computationally demanding due to the usage of the dense point cloud map. In this paper, we propose a novel lightweight LiDAR-based localization system for autonomous vehicle in this paper. The proposed system only relies on lightweight poles and curbs landmark map, which is highly robust and efficient compared to other localization systems. Poles and curbs are selected as landmarks because of their commonality and stability. We novelly propose a Branch-and-Bound (BnB)-based global optimization method to tackle the data association problem of poles. Motion decoupling is adopted to decouple translation and rotation to improve the efficiency of the BnB-based algorithm. Besides, we propose a new local grid map-based representation for curbs to make better use of curb information. Cost functions for pole and curb are defined respectively and then fused for the subsequent non-linear optimization method to obtain the vehicle location. Experiments on KITTI dataset and our self-collected dataset demonstrate the efficiency and accuracy of our system.**

***Index Terms*—Autonomous driving, map-based localization, Branch-and-Bound, landmark-based localization, pole landmark, road curb, grid map.**

Manuscript received August 27, 2020; revised May 14, 2021 and August 3, 2021; accepted September 7, 2021. Date of publication September 27, 2021; date of current version November 18, 2021. This work was supported in part by the Shanghai Rising Star Program under Grant 21QC1400900, in part by the National Natural Science Foundation of China under Grant 61906138, in part by Anhui Provincial Natural Science Foundation, Anhui Energy-Internet Joint Program under Grant 2008085UD01, in part by the National Natural Science Foundation of China under Grant U1913601, in part by the European Union's Horizon 2020 Framework Program for Research and Innovation under the Specific under Grant 945539 (Human Brain Project SGA3), and in part by the Major Science and Technology Projects of Anhui Province under Grant 202103a05020004. The review of this article was coordinated by Prof. Zhanyu Ma. *(Corresponding author: Zhijun Li.)*

Guang Chen, Fan Lu, Jinhu Dong, Junqiao Zhao, and Junwei Yu are with the Tongji University, Shanghai 200092, China (e-mail: guangchen@tongji.edu.cn; lufan@tongji.edu.cn; 1651873@tongji.edu.cn; zhaojunqiao@tongji.edu.cn; 13818102200@139.com).

Zhijun Li is with the Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, and Department of Automation, University of Science and Technology of China, Hefei 230026, China (e-mail: zjli@ieee.org).

Yinlong Liu and Alois Knoll are with the Chair of Robotics, Artificial Intelligence and Real-time Systems, Technical University of Munich, 80333 Munich, Germany (e-mail: yinlong.liu@tum.de; knoll@in.tum.de).

Digital Object Identifier 10.1109/TVT.2021.3114825

I. Introduction

ACCURATE and robust localization is one of the key components of intelligent vehicles [1]. The most common way to perform vehicle localization is using Global navigation satellite system (GNSS) with Inertial navigation system (INS). However, the cost of a high-accurate GNSS/INS system is extremely high and low-cost GNSS/INS systems can only achieve meter-level accuracy [2], which is insufficient for intelligent vehicles. Moreover, GNSS/INS systems suffer from signal block in urban scenarios [3], which makes the localization result unreliable.

Map-based method is an alternative to tackle the problem of localization for intelligent vehicles [4]–[22]. The most commonly used map for vehicle localization is LiDAR intensity map [4]–[6]. Although these methods can provide reliable and accurate localization, the memory storage consumption of the dense map will be unacceptable in large scale environments. Comparatively, lightweight landmark maps [8]–[22] are more compact and require less memory. Poles (*e.g.*, trees and street lamps) and curbs are common and stable in traffic scenarios. Hence, poles and curbs are applicable as landmarks for vehicle localization in urban scenarios.

However, the two kinds of landmarks have their inherent drawbacks. The data association problem of poles is intractable due to the lack of distinguishing features. Most of the current literature utilizes nearest neighbor search to perform pole-to-pole matching. The prerequisite of the method is that the detected poles are closest to the corresponding poles in the pre-built map after the transformation, which will be unsatisfied when the initial vehicle pose is inaccurate. The difficulty of curb is how to choose a proper representation. Generally, the detected curb is represented as a curve, which is not easy to utilize for localization. Most of the works only use the relative distance and angle between curb and the vehicle for the localization task while semantic information such as the curvature is ignored. Furthermore, almost all of the landmark-based localization methods use local optimization methods which can easily get stuck into local optima without careful initialization. The local optimum can be far from the true value and greatly reduces the robustness of the localization system.

To address the problem mentioned above, we propose a pole-curb fusion vehicle localization system. To tackle the data association problem of poles, we represent the detected poles and poles in the map as two 2D point sets and a Branch-and-Bound (BnB)-based method is utilized to generate the correct correspondences of poles. BnB is one of the most successful algorithms for global optimization problem [23]–[25]. However, it has not been widely used for localization problem in autonomous driving due to its high complexity and poor real time performance. The algorithm will take a considerable time for the convergence if the dimension of search space is 3 for the 3-DoF of vehicle localization. In the field of dense point cloud registration, motion decoupling is a common way to reduce the search dimension and improve the efficiency of algorithm although the real-time performance is still not guaranteed. [26], [27] proposed rotation-invariant features and transformation-invariant features for 3D point set registration, respectively. Similarly, [28] decouples scale, translation and rotation. Inspired by the above works, we propose a robust and efficient autonomous vehicle localization system with BnB-based global optimization and local grid map method. We utilize a motion decoupling method to solve the data association problem while achieve the real-time performance due to the usage of lightweight pole landmark map. The BnB-based rotation search method solves the data association problem of poles well and can also provide a rough pose estimation for the subsequent pose optimization. With the known correspondence, the pole cost function can be easily constructed. Grid map-based methods are widely used in 2D-SLAM [29], [30], however, few works leverage grid map in landmark-based localization problem. In this paper, we introduce a novel grid map-based representation for curbs, which is more complete and precise than general curve-based representation. The map of curb is represented as a set of local grid maps rather than a curve or several points. The values in each cell of the local grid map are defined using an inverse distance function. In localization process, several points are sampled from the detected curb points and projected onto the corresponding local grid map based on the rough pose estimation from the previous BnB-based rotation search method. The curb cost function is then defined based on the projection values. At last, the pole cost function and the curb cost function are fused and the subsequent non-linear optimization is utilized to refine the previous rough vehicle pose estimation.

To evaluate the accuracy and efficiency of our proposed pole-curb fusion localization system, we performed several experiments on our self-collected TJ-TiEV dataset and also KITTI dataset and the experiment results demonstrate the good performance of the proposed system.

Our main contributions are as follows:

- We propose a novel pole-curb fusion based autonomous vehicle localization system. Compared with dense map-based localization system, the lightweight pole-curb map requires much less memory and poles and curbs are common and stable in structured environments. Consequently, our system is robust and efficient without requirements for dense point cloud registration.
- We novelly utilize a BnB-based method to solve the data association problem of poles well. In comparison with commonly used local optimization methods, our proposed method avoids falling into local minima. Moreover, the introduction of motion decoupling significantly reduces the complexity of the algorithm and results in a real-time performance.
- We provide a novel map representation for curbs, which represents the curb map as a set of local grid maps, where the value in each cell represents the existence of curbs. Compared with the common curve-based method of representing curb as distance and angle, our representation is more complete and precise. The introduction of curbs improves the position accuracy of localization system based on only poles, especially the lateral accuracy.

II. Related Works

We briefly review related works about the map-based and landmark-based vehicle localization system from four aspects: dense map-based localization, pole-like landmark-based localization, curb/road marks-based localization and fusion method-based localization system.

A. Dense Map-Based Localization

Dense map-based localization system typically perform matching between current scans with point cloud map. [4] utilizes the LiDAR intensity of the environment to build a global point cloud map. [31] matches the point cloud with a self-adaptive multi-layer matching. [6] leverages deep learning-based method to perform point cloud matching. However, dense map-based localization system has its inherent drawback, i.e., the poor efficiency. Dense map-based method relies on dense point cloud matching and can take a considerable time for large-scale outdoor scenes.

B. Pole-Like Landmark-Based Localization

Currently, most of the pole-like landmark-based methods utilize filter-based methods (particle filter or kalman filter) to perform localization with fusion with GNSS/INS system. [9]–[11] all extract poles from point cloud and use particle filter to perform localization. [12] has the similar idea and the difference is that [12] detects pole from stereo camera system. All of the works above tackle the data association problem of poles via nearest neighbor search, which requires that the detected poles are closest to their corresponding poles in the map. When GPS signal is blocked and the vehicle motion is large in a frame or the poles are extremely dense, this prerequisite can not be satisfied and the wrong correspondence will lead to an incorrect localization result.

C. Curb/road Marking-Based Localization

Curb and road markings are common landmarks in urban scenarios which are generally used to improve the localization accuracy in lateral direction. [13] uses two cameras to detect road markings and then aids the vehicle localization using extended kalman filter (EKF). [14] uses a front camera to detected lane markings and perform localization via unscented kalman filter

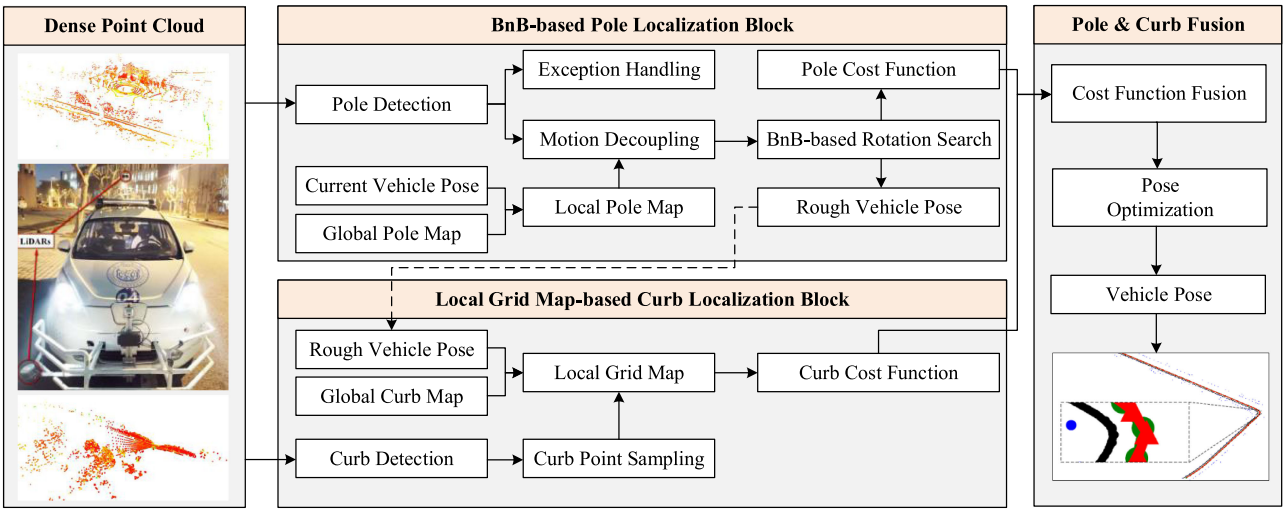


Fig. 1. The overview architecture of the proposed pole-curb fusion vehicle localization system, which consists of three blocks: BnB-based pole localization block, local grid map-based curb localization block and pole-curb fusion block.

(UKF). Although the measurement model of [14] and [13] is different, both of them treat the road marking as a point which can lead to a large information loss. [15] detects curbs from point cloud and the detection results work as an input to the Monte Carlo localization algorithm. [18] uses curb and intersection as the landmark of the Monte Carlo Localization and the curb is represented as virtual LiDARs in this paper. [17] detects lanes using around view monitoring camera and matches the lanes via ICP-based method. Nevertheless, none of the above methods provide a complete representation of curbs or lanes.

D. Fusion Method-Based Localization

Fusion method-based localization methods are the most relevant to our work. [19] fuses 3 landmarks: poles, facades and road markings. [19] accumulates detections over time to generate local map and matched the local map to the global map to find the best match to solve the data association problem. The whole framework is also based on non-linear optimization like us. Lanes and traffic signs are fused in [20] via a Bayesian filtering framework. Deep learning-based methods are utilized here to detect the landmarks and the detection results are represented as overview images. The detected image and the map are matched using a cross-correlation operation. [21] uses around view monitoring system to detect road boundaries and road markings. Road boundaries are used to obtain rough pose estimation and then the rough pose is refined by road markings matching. However, almost all of the above methods utilize local optimization methods to perform localization, which is easy to get stuck into local optimum.

III. METHOD

A. System Overview

The overview architecture of the proposed pole-curb fusion vehicle localization system is shown in Fig. 1. The system consists of 3 blocks: Branch-and-Bound (BnB)-based pole localization block, local grid map-based curb localization block and pole-curb fusion block.

The input of the system are two dense point clouds: point cloud for pole detection $\mathbf{X}_p$ and point cloud for curb detection $\mathbf{X}_c$. $\mathbf{X}_p$ is firstly fed into the BnB-based pole localization block and output a pole cost function with a rough estimation of the current vehicle pose. Similarly, the local grid map-based curb localization block receives $\mathbf{X}_c$ and output a curb cost function. The pole cost function and curb cost function are fused in the pole-curb fusion block to refine the rough vehicle pose. Several exceptions are also well handled in our system. The BnB-based pole localization is the dominant block and the local grid map-based curb localization block is only for refinement because poles provide both lateral and longitudinal constraints and curbs provide more lateral information. The detail process of the localization system is described in Algorithm 1. Noting that the initial vehicle pose P_i is rough and known by default (from GNSS or the vehicle pose of last frame).

B. BnB-Based Pole Localization Block

The process of the BnB-based pole localization block is shown in top row of the middle column of Fig. 1. Previous pole-based vehicle localization methods commonly adopt local optimization methods due to the complexity of global optimization method. However, the local optimization method can easily get stuck into local optimum without a careful initialization. Furthermore, most of the current methods do not well handle the correspondence problem of poles. Our proposed method tackles the above problems efficiently and effectively. First, a global pole map is built using the ground-truth vehicle pose. In the localization thread, the detected poles and the poles in the local map are represented as two 2D point sets. To reduce the complexity of the BnB algorithm, we propose a motion decoupling method to decouple rotation and translation and

Algorithm 1: Pole-Curb Fusion Vehicle Localization System.

Input Point cloud for pole detection $\mathbf{X}_p$, point cloud for curb detection $\mathbf{X}_c$, global pole map, global curb map.
Output: Vehicle pose P_o.

```
Initialize vehicle pose P_i.
Pole detection.
Search local pole map.
if Number of poles N_p > ε_n then
    Calculate rough vehicle pose P_r using BnB-based rotation search and generate pole cost function L_p.
    Curb detection.
    if Curb detected then
        Search local curb map.
        Curb points sampling.
        Generate curb cost function L_c.
        L = L_p + βL_c.
    else
        L = L_p.
    end if
    Refine P_r to P_o using L.
else if N_p ≠ 0 then
    Generate L_p using local grid map-based method.
    if Curb detected then
        Generate L_c.
        L = L_p + βL_c.
    else
        L = L_p.
    end if
    Refine P_i to P_o using L.
else
    Calculate P_o using uniform motion model.
end if
return Vehicle pose P_o.
```

generate several translation-invariant features (TIFs). Then the BnB-based rotation search method is utilized to search the rotation angle (*i.e.*, the yaw angle) and concurrently generate a rough estimation of vehicle pose. Based on the search result, pole-to-pole correspondences are straightway constructed, with which the pole cost function is defined based on the distance of the matching pole pairs. The details of this block will be discussed below and the exception handling module will be described at the end of this section.

1) Pole Detection: The LiDAR for pole detection is mounted on the roof of the vehicle in our system (see the left part of Fig. 1). Our pole detection method is based on [32] and the process consists of three steps: point cloud voxelization, horizontal cluster and vertical cluster.

Point cloud voxelization: The ground plane is firstly removed from the input point cloud using RANSAC-based method. For more convenient processing of point cloud, the point cloud is voxelized to a voxel grid. The resolution of the voxel grid is denoted as $(x_{res}, y_{res}, z_{res})$ and a cell is marked as valid if the number of points in the cell is greater than a threshold t_v.

TABLE I
PARAMETERS OF POLE DETECTION

x_{res}	y_{res}	z_{res}	t_v	t_{s1}	t_{s2}	$H_{\min}$	t_r
0.2m	0.2m	0.2m	5	15	3	1.0m	1.5

Horizontal cluster: The voxel grid is segmented into horizontal layers along the vertical direction. Based on the geometric characteristic of poles, the horizontal segment of the pole should satisfy two characteristics: a small dimension and isolation. All of the valid cells in the horizontal layer are clustered into a single segment based on the connectivity between cells. Then the spatial dimensions of the segments (i.e., the number of cells in the clustered segment) are checked to be smaller than a threshold t_{s1}. After that, we define two boxes (a smaller one and a bigger one) centered on the center of the segment and count the valid cells between two boxes to verify the isolation. The segment will be considered as unsatisfying the isolation if the number is larger than a threshold t_{s2}. The segments that pass the above two checks (*i.e.*, a small dimension and isolation) are marked as valid.

Vertical cluster: Similarly, the valid horizontal segments will be further aggregated along the vertical direction based on their vertical connectivity. Due to the discontinuity of the point cloud in its vertical direction, we expand the vertical search range to two layers above and below. Then the height and the height to width ratio of the candidate vertical clusters are checked. A cluster is considered as a detected pole if the height $H \geq H_{\min}$ and the ratio $r \geq t_r$.

The 2D vehicle localization only requires the 2D location of poles on the ground plane. Fitting the point cloud to a cylinder model is a common way to estimate the center of the pole, which is found to be unstable in our implementation. Therefore, we directly calculate the average location of the points belonging to a pole as the center of the pole.

The parameters of pole detection are presented in Table I. We also display several samples of pole detection results on point clouds of Velodyne HDL-64E and Velodyne VLP-16 in the first and second row of Fig. 2. Several detected poles are zoomed in to show the details and N, H and r denote the number of points, the height and the height to width ratio of the pole, respectively. The results demonstrate that the pole detection algorithm can successfully detect poles with various point densities and geometric sizes..

2) Motion Decoupling: BnB-based method has not been widely applied on vehicle localization problems due to the high time complexity, which increases exponentially with the search dimension. The search dimension of the 2D vehicle pose is 3 (*i.e.*, (x, y, θ)), which results in a considerable time for the BnB-based method to get convergence to global optima. Motion decoupling is an efficient and effective way to decrease the search dimension.

The currently detected poles and the poles in the local map are represented as two point sets, denoted as $\mathcal{M} = \{\boldsymbol{m}_i\}_{i=1}^{M}$ and $\mathcal{N} = \{\boldsymbol{n}_j\}_{j=1}^{N}$. $\mathcal{M}$ can be transformed to $\mathcal{N}$ using rotation matrix $\mathbf{R}$ and translation vector $\boldsymbol{t}$, the 3-dimensional vehicle

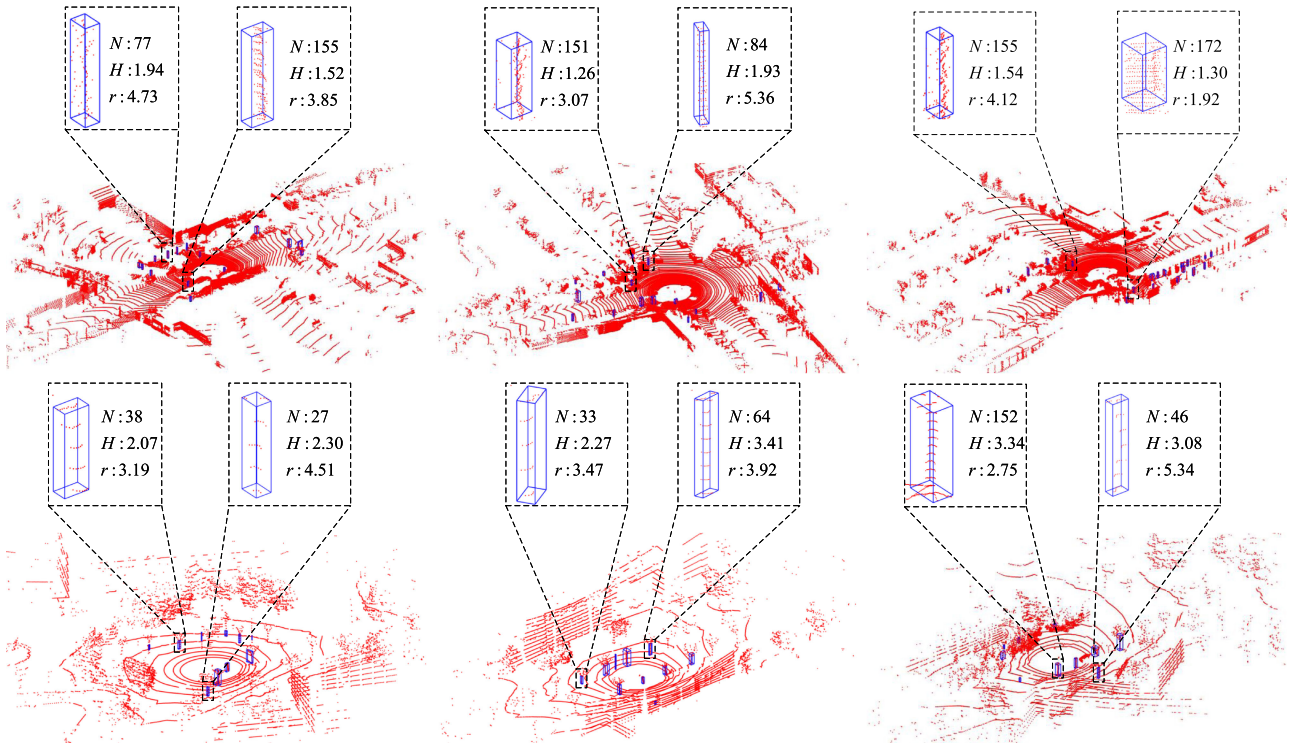


Fig. 2. Several examples of pole detection results. The first and second row display the detection results on point clouds of Velodyne HDL-64E and Velodyne VLP-16. We zoom in several detection results to show the details of detected poles, where N, H and r denote the number of points, the height and the height to width ratio of the pole, respectively.

pose (x, y, θ) can be recovered from $\mathbf{R}$ and $\boldsymbol{t}$. Given two points $\boldsymbol{m}_1, \boldsymbol{m}_2$ in $\mathcal{M}$ and the corresponding points $\boldsymbol{n}_1, \boldsymbol{n}_2$ in $\mathcal{N}$,

$$\boldsymbol{n}_1 = \mathbf{R}\boldsymbol{m}_1 + \boldsymbol{t} \tag{1}$$

$$\boldsymbol{n}_2 = \mathbf{R}\boldsymbol{m}_1 + \boldsymbol{t} \tag{2}$$

Subtracting Eq. 1 from Eq. 2, then,

$$\boldsymbol{n}_1 - \boldsymbol{n}_2 = \mathbf{R}(\boldsymbol{m}_1 - \boldsymbol{m}_2) \tag{3}$$

Obviously, the transformation from $\boldsymbol{m}_1 - \boldsymbol{m}_2$ to $\boldsymbol{n}_1 - \boldsymbol{n}_2$ is only related to the rotation matrix $\mathbf{R}$ rather than translation $\boldsymbol{t}$. The features like $\boldsymbol{m}_1 - \boldsymbol{m}_2$ and $\boldsymbol{n}_1 - \boldsymbol{n}_2$ are called translation-invariant features (TIFs). TIFs can be constructed by subtracting every two points in the point set. BnB-based algorithm can be performed to search $\mathbf{R}$ on TIFs rather than $\mathbf{R}$ and $\boldsymbol{t}$ on the two point sets. Consequently, the search dimension decreases from 3 to 1, which significantly improves the efficiency of the BnB-based search method.

3) BnB-Based Rotation Search: Two TIF sets $\mathcal{P} = \{\boldsymbol{p}_i\}_{i=1}^{P}$, $\mathcal{Q} = \{\boldsymbol{q}_j\}_{j=1}^{Q}$ can be constructed from the forementioned two point sets of detected poles and the poles in the local pole map.

The BnB-based rotation search aims to find the best rotation matrix $\mathbf{R}^*$ to match the two TIF sets. We seek for the rotation matrix $\mathbf{R}$ that maximises the quality function

$$Q(\mathbf{R}) = \sum_{i}^{P} \max_{j} \lfloor \left\| \mathbf{R}\boldsymbol{p}_i - \boldsymbol{q}_j \right\| \leq \epsilon \rfloor \tag{4}$$

where $\lfloor \cdot \rfloor$ is a function that returns 1 if the condition $\cdot$ is true and 0 otherwise, $\| \cdot \|$ is the Euclidean norm in $\mathbb{R}^2$, ϵ is the inlier threshold, which will be further discussed in the experiments.

From a point set with n points we can generate $n(n-1)/2$ TIFs. Consequently, searching will be time consuming due to the large scale of TIF sets. Noting that the norm of a TIF should be constant after a rotation, so a rough matching can be firstly established based on the norm of the TIFs.

Generally the number of the detected poles is smaller than the number of the poles in local map, and almost all of the detected poles should have a corresponding pole in the local map. Therefore, we choose the TIF set of the detected poles $\mathcal{P}$ to construct several equal norm TIF sets $\{\mathcal{P}_k\}_{k=1}^{K}$. TIFs with a difference in norm less than a threshold ϵ_s will be grouped into a single equal norm TIF set. And the average value of the norm in a single equal norm TIF set will be calculated and stored in the norm set which is denoted as $\{\mathcal{E}_k\}_{k=1}^{K}$. The norm of each TIF in $\mathcal{Q}$ will be compared to the norm in $\{\mathcal{E}_k\}_{k=1}^{K}$, if the difference is less than a threshold ϵ_s, the TIF will be assigned into $\mathcal{Q}_k$. As the result, a rough matching between $\mathcal{P}_k$ and $\mathcal{Q}_k$ is established. Hence, Eq. 4 can be rewrited as

$$Q(\mathbf{R}) = \sum_{k=1}^{K} \sum_{p_i \in \mathcal{P}_k} \max_{q_j \in \mathcal{Q}_k} \lfloor \left\| \mathbf{R}\boldsymbol{p}_i - \boldsymbol{q}_j \right\| \leq \epsilon \rfloor \tag{5}$$

Algorithm 2 summarizes the BnB-based search algorithm. The rotation search space is represented as a 1D line $\mathbb{L}$ with range $[-\pi, \pi]$. The BnB-based algorithm iteratively explores

Algorithm 2: BnB-Based Search Algorithm.

Input the equal norm TIF sets, threshold ϵ
Output: Optimal rotation $\mathbf{R}^*$ with quality Q^*
Initialize priority queue q, $Q^* \leftarrow 0$, $\mathbf{R}^* \leftarrow I$, $\mathbb{L} \leftarrow [-\pi, \pi]$
Insert $\mathbb{L}$ with priority $\hat{Q}(\mathbb{L})$ into q.
while q is not empty **do**
  Obtain highest priority subset $\mathbb{L}$ from q
  if $\hat{Q}(\mathbb{L}) = Q^*$ **then** Terminate
  end if
  $\mathbf{R}_c \leftarrow$ center rotation of $\mathbb{L}$.
  if $Q(\mathbf{R}_c) > Q^*$ **then**
    $Q^* \leftarrow Q(\mathbf{R}_c), \mathbf{R}^* \leftarrow \mathbf{R}_c$
  end if
  Divide $\mathbb{L}$ into to two subsets $\{\mathbb{L}_i\}_{i=1}^2$ at the center of $\mathbb{L}$.
  for each $\mathbb{L}_i$ **do**
    if $\hat{Q}(\mathbb{L}_i) \geq Q^*$ **then**
      Insert $\mathbb{L}_i$ with priority $\hat{Q}(\mathbb{L}_i)$ into q
    end if
  end for
end while
return Optimal rotation $\mathbf{R}^*$ with quality Q^*.

the branches of a rooted tree, each branch represents a subset of the search space and the root is the full search space. For each branch, the priority (e.g., the upper bound of the quality Q in this branch) is calculated. The search algorithm selects the branch with the highest priority, calculates the quality of the center of the branch, and updates the best rotation with the best quality. The branch is then divided into two sub-branches and the sub-branch will be cut if the priority is smaller than the best quality.

In Algorithm 2, the upper bound $\hat{Q}(\mathbb{L})$ (*i.e.*, the priority of a branch $\mathbb{L}$) should satisfies

$$\hat{Q}(\mathbb{L}) \geq \max_{\mathbf{R} \in \mathbb{L}} Q(\mathbf{R}) \tag{6}$$

In our algorithm, the upper bound can be calculated as,

$$\hat{Q}(\mathbb{L}) = \sum_{k=1}^{K} \sum_{p_i \in \mathcal{P}_k} \max_{q_j \in \mathcal{Q}_k} \lfloor \left\| \mathbf{R}_c \boldsymbol{p}_i - \boldsymbol{q}_j \right\| \leq \epsilon + \delta_r \rfloor \tag{7}$$

where $\delta_r = 2\|\boldsymbol{p}_i\| \sin(L_r/4)$, L_r is the length of $\mathbb{L}$, $\mathbf{R}_c$ is the center rotation of $\mathbb{L}$.

The proof of the upper bound is given below. First, given two vector $\mathbf{a}, \mathbf{b}$, it is obviously that $\|\boldsymbol{a}\| - \|\boldsymbol{b}\| \leq \|\boldsymbol{a} + \boldsymbol{b}\|$. Thus, given $\|\mathbf{R}\boldsymbol{p}_i - \boldsymbol{q}_j\| \leq \epsilon$, we have,

$$\begin{aligned} \left\| \mathbf{R}\boldsymbol{p}_i - \boldsymbol{q}_j \right\| &= \left\| \mathbf{R}\boldsymbol{p}_i - \boldsymbol{R}_c\boldsymbol{p}_i + \boldsymbol{R}_c\boldsymbol{p}_i - \boldsymbol{q}_j \right\| \\ &\geq \left\| \mathbf{R}_c\boldsymbol{p}_i - \boldsymbol{q}_j \right\| - \|(\mathbf{R} - \mathbf{R}_c)\boldsymbol{p}_i\| \\ &= \left\| \mathbf{R}_c\boldsymbol{p}_i - \boldsymbol{q}_j \right\| - 2\|\boldsymbol{p}_i\| \sin \angle(\mathbf{R}, \mathbf{R}_c)/2 \end{aligned} \tag{8}$$

where $\angle(\mathbf{R}, \mathbf{R}_c)$ is the relative angle between $\mathbf{R}$ and $\mathbf{R}_c$, and $\angle(\mathbf{R}, \mathbf{R}_c) \leq L_r/2$, thus,

$$\begin{aligned} \left\| \mathbf{R}_c\boldsymbol{p}_i - \boldsymbol{q}_j \right\| &\leq \left\| \mathbf{R}\boldsymbol{p}_i - \boldsymbol{q}_j \right\| + 2\|\boldsymbol{p}_i\| \sin(L_r/4) \\ &\leq \epsilon + \delta_r \end{aligned} \tag{9}$$

According to Eq. 9, if $\|\mathbf{R}\boldsymbol{p}_i - \boldsymbol{q}_j\| \leq \epsilon$, then $\|\mathbf{R}_c\boldsymbol{p}_i - \boldsymbol{q}_j\| \leq \epsilon + \delta_r$. Consequently, Eq. 7 is the upper bound of Eq. 5.

4) Rough Pose Estimation: After BnB-based rotation search, several matched pairs of are naturally constructed. However, the search result may differ from the real value by π. Suppose the true rotation angle is θ, given two points $\boldsymbol{m}_1, \boldsymbol{m}_2$ in $\mathcal{M}$ and the corresponding points $\boldsymbol{n}_1, \boldsymbol{n}_2$ in $\mathcal{N}$, $\boldsymbol{m}_1 - \boldsymbol{m}_2$ can be aligned with $\boldsymbol{n}_1 - \boldsymbol{n}_2$ with rotation angle θ, $\boldsymbol{m}_1 - \boldsymbol{m}_2$ can also be aligned with $\boldsymbol{n}_2 - \boldsymbol{n}_1$ with rotation angle $\theta - \pi$ or $\pi - \theta$.

Nevertheless, the midpoints of two matched TIFs should be matched regardless of the direction of the TIFs. Therefore, we calculate the translations of the matched midpoints and the average value of the translations is considered as the rough translation estimation (x_r, y_r). For each possible rotation angle, we can get a set of corresponding points pairs. Translations can also be calculated with these pairs. Apparently, the rotation angle θ_r with the translation most close to (x_r, y_r) is selected as the correct value. Consequently, the rough estimation of the pose should be (x_r, y_r, θ_r).

5) Pole Cost Function: When the correct value of the rotation angle is obtained, the correct correspondence of the poles is established concurrently. Suppose corresponding pairs set is $\{\boldsymbol{m}_k, \boldsymbol{n}_k\}_{k=1}^K$, the cost function is shown as,

$$\mathcal{L}_p = \sum_{k=1}^{K} \|\boldsymbol{m}_k - \boldsymbol{n}_k\| \tag{10}$$

Until now, the proposed BnB-based pole localization method solves the data association problem of poles well. Because the BnB-based rotation search seeks best rotation in the entire search space, our method ensures global optimization so that does not require careful initialization. With the help of motion decoupling, the method achieves real-time performance while promising robustness.

C. Local Grid Map-Based Curb Localization Block

The process of the local grid map-based curb localization block is shown in lower row of the middle column in Fig. 1. Unlike the pole localization block, the curb map is represented as a set of local grid maps. When curb is detected, several curb points are sampled to reduce the scale of the raw detected curb point set. Then the corresponding local grid map is searched based on the rough vehicle pose from pole localization block. After that the sampled points are projected onto the local grid map and a curb cost function is defined for the subsequent pole-curb fusion block.

1) Curb Detection: In our method, the LiDAR for curb detection is mounted on the right lower corner of the vehicle (see left column of Fig. 1) which reduces the complexity of the detection algorithm. We firstly filter the points out of the region of interest of the curbs and project the input LiDAR point cloud

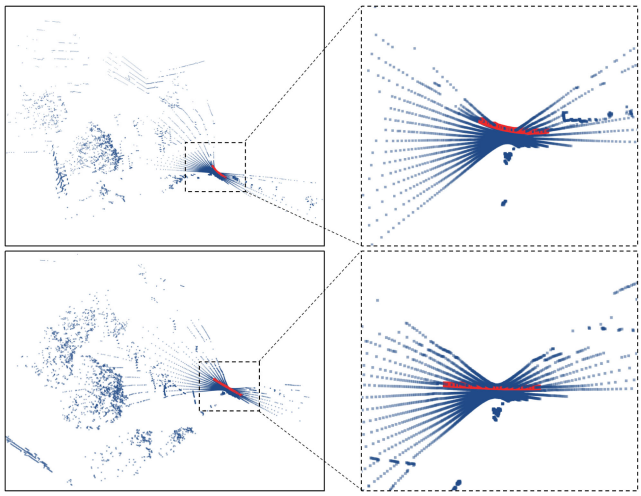


Fig. 3. Two samples of curb detection results. The blue points are the raw point clouds and the red points denote the detected curb points. We zoom in the region of interest in the right part for better visualization.

onto to horizontal 2D Cartesian grid. Each of the cells in the grid contains two parameters: the number of points in the cell (i.e., the density D) and the height of the cell (denoted as H). The cell that may contain curbs should meet two requirements: the density should larger than a certain threshold $D_{\min}$ and the height H should be within a certain range $[H_{\min}, H_{\max}]$. Several candidate cell is selected based on the requirements above. Then the candidate cells are clustered based on the connectivity and the cluster with the maximum size is selected as the candidate curb cluster. The candidate curb cluster is considered as a curb if the size is larger than a threshold $S_{\min}$. We display two samples of curb detection results in Fig. 3.

2) Curb Point Sampling: Different from poles, a curb can not be represented as a single point directly. And the common curve-based representation is not structured, thus is difficult to perform matching or define cost function. To retain complete information, a simple way is using all of the detected points to represent a curb. However, the scale of the point set is too large which consumes considerable memory resources. Therefore we calculate the average position of all points in a cell and represent the cell using this single average point. This process reduces the number of points from hundreds to tens and preserves the majority of information.

3) Local Grid Map: How to represent curb in the map is a remained problem to be solved. A better map representation can significantly improve the utilization of curb information. Currently, most of the works utilize only the relative distance and angle between the vehicle and the curb while most of the information is lost. To address the above problems, we propose a novel map representation for curbs. In the mapping thread, after curb detection and curb point sampling, several curb points are generated and further transformed onto the global curb map (see the black lines in Fig. 6). However, these raw curb points can not be directly used for localization and how to use the points is the key problem. Unlike other intelligent robotics, vehicles always move on the road. Thus, most of the regions in the global map are useless. Based on the above consideration, the global map is broken into a set of local grid maps along the road and each local grid map has its local coordinate frame and a global location (x_M, y_M, θ_M). The spacial size of the local grid map is chosen to be the receptive field of LiDAR. The resolution of the local grid map should be the same as the resolution of the grid for curb detection. As shown in Eq. 11, the value in each cell is calculated using a inverse distance function.

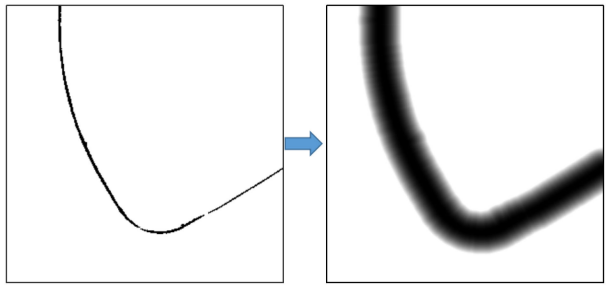


Fig. 4. An example of local grid map. The left column of is the original curb map and the right column is the local grid map. The larger the gray value, the larger the value in the cell.

$$value = \max\left\{\frac{1}{1+\alpha d_1}, \ldots, \frac{1}{1+\alpha d_n}\right\} \tag{11}$$

where d_i represent the distance of the cell to the closest curb point, n is the number of curb points and α is a control coefficient ($\alpha = 4$ in our implementation).

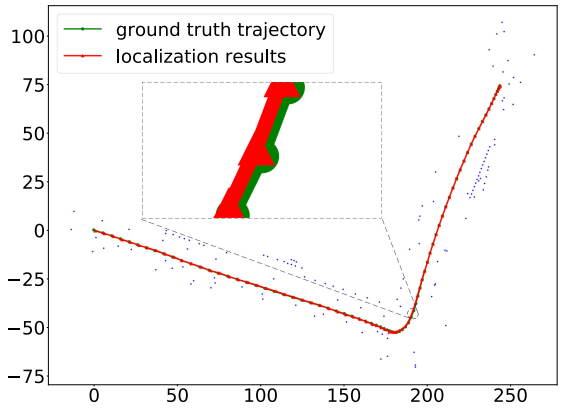
Two examples of local grid maps can be seen in Fig. 4. The further the cell away from the curve, the smaller the value in it and maximum value in the grid map is 1.

4) Curb Cost Function: As mentioned before, the global map is broken into a set of local grid maps. Given the rough pose estimation from pole localization, nearest neighbor search is performed to get the corresponding local grid map. In order to further improve the efficiency of the algorithm, we randomly sample N points from the pre-sampled curb points and N should strike a balance between accuracy and efficiency. In our implementation, N is set to 10 according to the localization result. Then the sampled curb points are projected onto the local grid map frame based on the rough pose estimation from pole localization block. The next step is to get the value of the local grid map at the projected location. We adopt a bicubic interpolation method like [30] instead of simply rounding the position to get the corresponding value. The interpolation method is convenient to implement with the help of Ceres-solver [33].

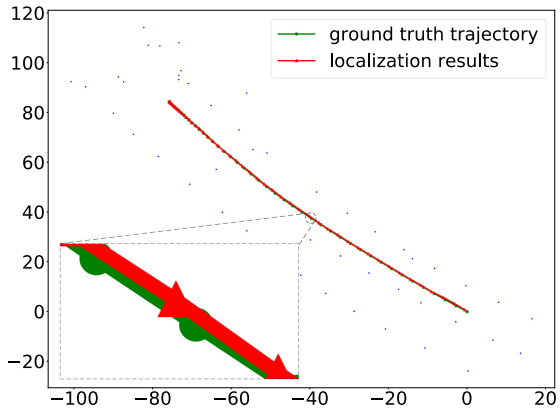
Suppose there are N sampled curb points and the projected location of the $i-th$ curb point is Y_i, $f(Y)$ represents the interpolation function, the curb cost function can be calculated as Eq. 12. Note that $f(Y_i) \leq 1$ according to the definition of the value in the local grid map.

$$\mathcal{L}_c = N - \sum_{i=1}^{N} f(Y_i) \tag{12}$$

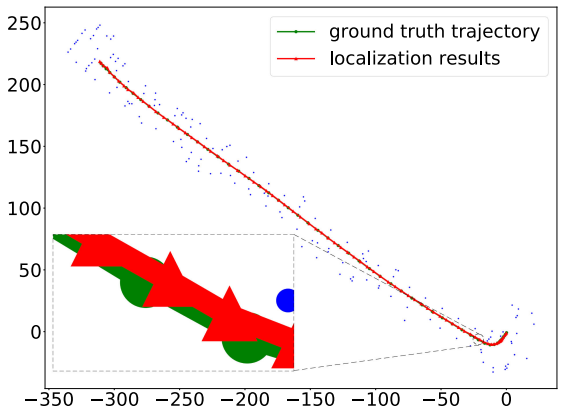
Compared to the methods using only the relative distance and angle between the vehicle and the curb, our map representation



(a)



(b)



(c)

Fig. 5. The ground-truth vehicle trajectories and the localization results on 3 sequences in KITTI dataset. The blue points represent the poles in the global map. The red lines are the localization results and the green lines are the ground-truth vehicle trajectories. We also perform a partial zoom to better visualize the results. (a) Sequence:0009 (b) Sequence:0011 (c) Sequence:0096.

is more complete and precise. Our representation works with curbs of various shapes, straight or curved curb does not affect the process of our method while the method based on the relative distance and angle can only be applied to straight curbs. Almost all of the geometric features of curbs (*e.g.*, distance, curvature and angle) can be presented in our local grid map.

D. Pole-Curb Fusion Block

The process of the pole-curb fusion block is presented on the right column of Fig. 1. We utilize a non-linear optimization method which significantly simplifies the fusion. Pole cost function ($\mathcal{L}_p$) and curb cost function ($\mathcal{L}_c$) are obtained from pole localization block and curb localization block, respectively. The final cost function $\mathcal{L}$ is a combination of $\mathcal{L}_p$ and $\mathcal{L}_c$,

$$\mathcal{L} = \mathcal{L}_p + \beta \mathcal{L}_c \tag{13}$$

where β is a coefficient that controls the weight of pole cost function and curb cost function, which will be further discussed in the experiments.

The non-linear optimization method receives $\mathcal{L}$ as cost function with the rough vehicle pose estimation as initialization and then generates the refined vehicle pose. As mentioned before, the whole system is based on the pole localization block and the system can work without curb and $\mathcal{L} = \mathcal{L}_p$ in this case.

E. Exception Handling

The above situation is the normal situation of the localization system, however, there are several exceptions to be handled to improve the robustness of the system. The first is that the stability of the BnB-based pole localization is unsatisfied when the number of detected poles is small. The BnB-based search algorithm is based on the generated TIFs and the matching result is unreliable when the number of TIFs is not sufficient. An extreme case is that there is only one pole detected, then there is no TIF generated and the algorithm will not work. To solve this problem, we utilize a similar local grid map-based method like curb-based localization when the number of detected poles is less than a threshold ϵ_n. We just replace the curb points with the pole points, others else are the same as curb-based localization. And the vehicle pose for projection and local grid map search is set to the vehicle pose of last frame. The second exception is that there will be short areas without poles. In this case, we simply adopt uniform motion model to estimate the current vehicle pose based on the pose of historical frame.

IV. Experiments

To demonstrate the performance of the proposed pole-curb fusion vehicle localization system, we performed several experiments. Our experiments can be divided into two parts: pole-based localization and pole-curb fusion-based localization. For pole-based localization, we performed experiments on KITTI dataset [34] and our self-collected TJ-TiEV dataset. For pole-curb fusion localization, we performed experiments only on TJ-TiEV dataset. All the experiments are implemented using ROS (Robot Operating System). PCL toolkit [35] and Ceres-solver [33] are utilized to perform point cloud processing and non-linear optimization, respectively. The computing platform is a PC with Intel Core i7-9750H CPU at the clock speed of 2.60 Hz.

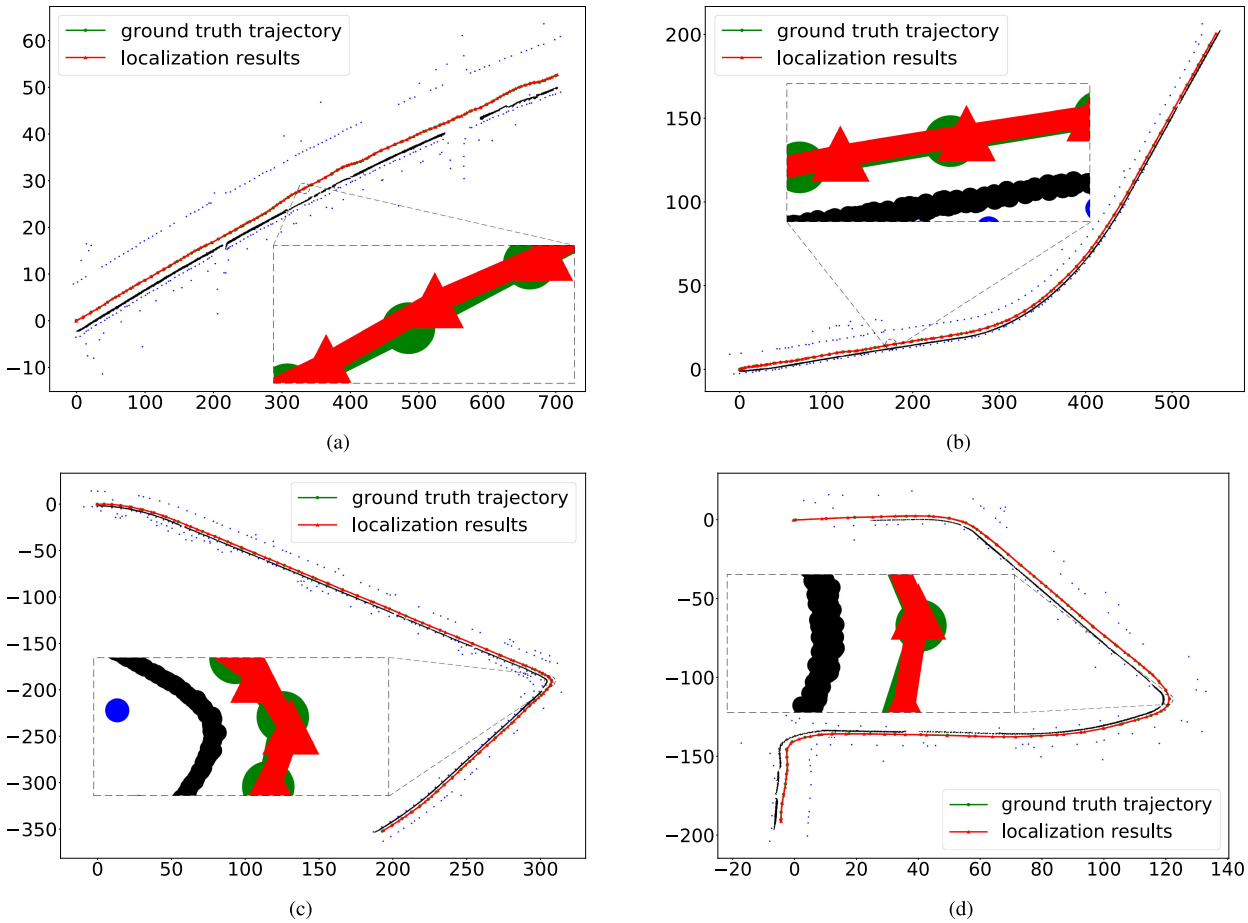


Fig. 6. The ground-truth vehicle trajectories and the localization results on 4 sequences of TJ-TiEV dataset. The blue points represent the poles and the black line represents curb in the global map. The red lines are the localization results of our pole-curb fusion localization system and the green lines are the ground-truth vehicle trajectories. (a) Sequence1: Straight (b) Sequence2: Small angle turn (c) Sequence3: Right-angle turn (d) Sequence4: Continuous turn.

A. Datasets

1) KITTI Dataset: KITTI dataset is one of the most important datasets in the field of autonomous driving. The LiDAR sensor used in KITTI dataset is Velodyne HDL-64E, which is mounted on the roof of the vehicle. Here we selected 3 sequences in the City scene of raw data in KITTI dataset, namely: *2011_09_26_drive_0009* (0009), *2011_09_26_drive_0011* (0011) and *2011_09_26_drive_0096* (0096). All of the selected sequences are recorded in typical city highways with street trees on both sides and the vehicle trajectories can be seen in Fig. 5.

2) TJ-TiEV Dataset: To evaluate the proposed pole-curb fusion vehicle localization system, we collected our own TJ-TiEV dataset [36]. The platform for data collection is TiEV from Tongji University and the data collection scenario is located in the Jiading campus of Tongji University. Two Velodyne VLP-16 LiDAR sensors are used here, one of which is mounted on the roof for pole detection and the other is mounted on the lower right front side of the vehicle for curb detection. The data collection platform and the mounting location can be seen in the left column of Fig. 1. Here we selected 4 sequences in TJ-TiEV dataset to evaluate the proposed localization system. As shown in Fig. 6, the 4 sequences cover 4 different shaped trajectories: straight, small-angle turn, right-angle turn and continuous turn, which cover most situations during normal vehicle driving.

B. Evaluation

The accuracy of our localization system is evaluated using root mean square error (RMSE) in position ($\mathrm{RMSE}_{\mathrm{pos}}$), yaw angle ($\mathrm{RMSE}_{\mathrm{yaw}}$), longitudinal ($\mathrm{RMSE}_{\mathrm{lon}}$) and lateral ($\mathrm{RMSE}_{\mathrm{lat}}$) direction. Mean absolute errors ($\Delta_{\mathrm{pos}}, \Delta_{\mathrm{yaw}}, \Delta_{\mathrm{lon}}$ and Δ_{lat}) are also provided for reference. The efficiency is evaluated using average runtime of one frame. In order to concentrate on the vehicle position estimation process, we exclude the runtime for pole detection and curb detection from the final runtime for evaluation.

C. Experiments Settings

As we mentioned before, the localization system can work well without curbs. To evaluate the performance of the proposed method using only poles as landmark, we performed experiments on the 3 sequences on KITTI dataset and 4 sequences of TJ-TiEV dataset. The qualitative results on KITTI dataset can be seen in Fig. 5 and the red lines (localization results) and the

TABLE II
THE LOCALIZATION PERFORMANCE OF POLE-BASED LOCALIZATION ON KITTI DATASET

Sequence	$\mathrm{RMSE}_{\mathrm{pos}}$ (m)	$\mathrm{RMSE}_{\mathrm{yaw}}$ (deg)	$\mathrm{RMSE}_{\mathrm{lon}}$ (m)	$\mathrm{RMSE}_{\mathrm{lat}}$ (m)	Δ_{pos} (m)	Δ_{yaw} (deg)	Δ_{lon} (m)	Δ_{lat} (m)
0009	0.262	0.688	0.194	0.176	0.220	0.516	0.155	0.132
0011	0.198	0.201	0.118	0.160	0.180	0.143	0.09	0.136
0096	0.264	0.527	0.153	0.215	0.225	0.378	0.121	0.160

TABLE III
THE LOCALIZATION PERFORMANCE OF POLE-BASED AND POLE-CURB FUSION LOCALIZATION ON TJ-TIEV DATASET

Sequence	landmark	$\mathrm{RMSE}_{\mathrm{pos}}$ (m)	$\mathrm{RMSE}_{\mathrm{yaw}}$ (deg)	$\mathrm{RMSE}_{\mathrm{lon}}$ (m)	$\mathrm{RMSE}_{\mathrm{lat}}$ (m)	Δ_{pos} (m)	Δ_{yaw} (deg)	Δ_{lon} (m)	Δ_{lat} (m)
1	pole	0.211	0.367	0.183	0.105	0.178	0.229	0.141	0.085
2	pole	0.170	0.281	0.123	0.117	0.157	0.200	0.140	0.073
3	pole	0.178	0.309	0.134	0.116	0.156	0.218	0.100	0.096
4	pole	0.189	0.453	0.140	0.127	0.172	0.297	0.114	0.108
1	pole+curb	0.199	0.315	0.178	0.090	0.169	0.217	0.140	0.073
2	pole+curb	0.160	0.298	0.124	0.102	0.147	0.212	0.105	0.085
3	pole+curb	0.158	0.327	0.120	0.102	0.141	0.229	0.095	0.083
4	pole+curb	0.179	0.400	0.140	0.111	0.162	0.275	0.116	0.093

green lines (ground-truth vehicle trajectories) roughly coincide. The detailed performance on the two datasets are displayed in Table II and III, respectively.

To demonstrate the performance of the whole pole-curb fusion system, we also performed experiments on TJ-TiEV dataset with both poles and curbs as landmark. The qualitative and detailed results are shown in Fig. 6 and Table III. The initial pose of vehicle is known by default in all experiments.

D. Performance Analysis

1) Accuracy: As shown in Table II, the maximum RMSE in position and yaw angle of the 3 sequences in KITTI dataset are 0.262 m and 0.688 °, respectively. The RMSE in longitudinal direction and lateral direction are controlled within 0.20 m and 0.22 m. On TJ-TiEV dataset, when poles are the only landmark, the maximum RMSE in position is 0.211 m and in yaw angle is 0.453 °. When curbs are introduced, the accuracy for position, longitudinal and lateral direction are basically all improved. The maximum RMSE in position and yaw angle are reduced to 0.199 m and 0.400 °, respectively. The most obvious sequence is sequence 3, the RMSE in position, longitudinal and lateral direction are decreased by 0.02 m, 0.014 m and 0.014 m respectively. For the other 3 sequences, the accuracy improvement in the lateral direction is apparently more than that in the longitudinal direction. This also matches our intuition: curbs mainly provide lateral constraint for vehicle localization.

2) Efficiency: The average runtime (without pole and curb detection) is displayed in Table IV. When only poles are used as landmark, the average runtime is controlled within 7 ms. When curbs are fused in the localization system, the runtime reached about 15 ms. Thanks to the motion decoupling method, our localization system achieves high efficiency even with the BnB-based global optimization algorithm. Although the introduction of curbs decreases the efficiency, the average runtime is also acceptable compared to the typical 10 Hz frame rate of LiDAR.

3) Comparison With Point Cloud Matching-Based Method: To compare the proposed pole-curb fusion based localization system with typical point cloud matching-based method, we generate a global dense point cloud map using the ground truth vehicle pose and then perform point cloud registration in the local point cloud map to obtain the vehicle pose. To improve the robustness of the registration, we extract FPFH features [37] from point clouds and adopt RANSAC [38] to calculate relative transformation. The experiments are performed on 4 sequences of TJ-TiEV dataset and we calculate the average value for better comparison. The results are shown in Table V and we denote the point cloud matching-based method as PCM. According to the results, the proposed method achieves comparable localization accuracy to point cloud matching-based method. However, since the proposed method only requires lightweight landmark map rather than dense point cloud map, the map storage of our method requires only about $1/90$ memory of point cloud matching-based method.

TABLE IV
THE RUNTIME OF OUR LOCALIZATION SYSTEM (WITHOUT POLE DETECTION AND CURB DETECTION)

Sequence	landmark	Runtime (ms)
0009	pole	6.0
0011	pole	5.2
0096	pole	6.4
1	pole	3.9
	pole+curb	14.3
2	pole	5.6
	pole+curb	13.0
3	pole	7.0
	pole+curb	15.1
4	pole	5.3
	pole+curb	12.0

TABLE V
LOCALIZATION PERFORMANCE OF THE PROPOSED METHOD AND POINT CLOUD MATCHING-BASED METHOD

Method	$\mathrm{RMSE}_{\mathrm{pos}}$ (m)	$\mathrm{RMSE}_{\mathrm{yaw}}$ (deg)	Memory (MB)
Ours	0.174	0.335	0.14
PCM	0.219	0.290	12.4

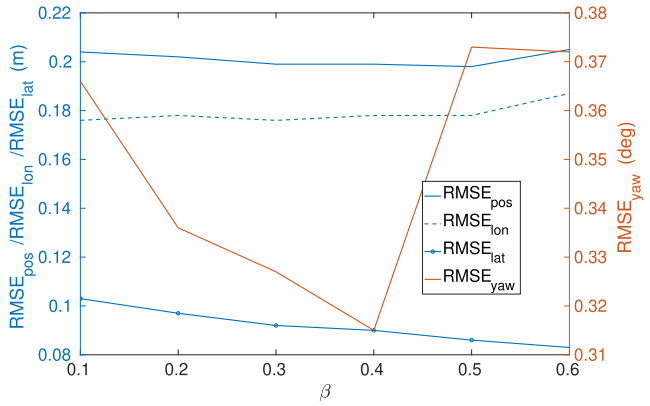


Fig. 7. Localization performance with different coefficients β.

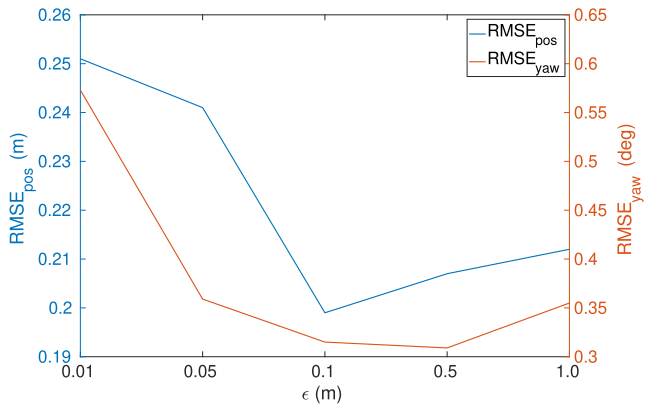


Fig. 8. Localization performance with different thresholds ϵ.

4) Hyperparameters Analysis: There are two import hyperparamters in the proposed localization system, namely the coefficient β in Eq. 13 and the threshold ϵ in Eq. 4. We perform experiments with different hyperparameters on Sequence 1 of Tj-TiEV dataset to analyze the effects. (a) **Coefficient β**: The coefficient β controls the contribution of the pole and curb cost function on the final localization results. The localization performance with different β is shown in Fig. 7. According to the results, the lateral localization error decreases with increasing β, which is consistent with our intuition that curbs mainly provide lateral constraints. However, due to possible detection errors, large β may cause some interference in the final optimization, thus reducing the overall localization accuracy. (b) **Threshold ϵ**: The threshold ϵ should be determined by the error of the pole detection algorithm. If ϵ is much smaller than the detection error, the number of selected matched pairs will be small or even zero, which can result in large localization errors and even cause failure results. However, a large ϵ will make the correspondence set contain inaccurate matching pairs, thus reducing the localization accuracy. Based on the above consideration, we perform experiments with different thresholds ϵ to compare the performance and experiment results are displayed in Fig. 8. According to the results, $\epsilon = 0.1m$ is a proper choice to achieve better performance.

5) Robustness: Due to the global optimal property of the BnB-based algorithm, the localization system avoids falling into local optima, which leads to better robustness. We perform experiments on Sequence 1 by adding random perturbations to the pole detection results to demonstrate the robustness. We introduce 3 perturbations here: (a) **Random noise (RN)**: random Gaussian noise with mean 0 and variance 0.1 is added to the pole detection results. (b) **Random discard (RD)**: 20% pole detection results are randomly discarded. (c) **Random add (RA)**: we randomly add 20% error detection results for interference. Besides, we also perform experiments with the combination of the 3 perturbations: **RN+RD**, **RN+RA**, **RA+RD**, **RN+RA+RD**. The experiment results are shown in Table VI. According to the results, the introduced perturbations will degrade the performance to some extent, but none of them will cause failure results, which demonstrates the robustness of the proposed algorithm.

TABLE VI
LOCALIZATION PERFORMANCE WITH PERTURBATIONS

Perturbations	$\mathrm{RMSE}_{\mathrm{pos}}$ (m)	$\mathrm{RMSE}_{\mathrm{yaw}}$ (deg)
Random noise (RN)	0.211	0.453
Random discard (RD)	0.212	0.378
Random add (RA)	0.205	0.372
RN + RD	0.221	0.443
RN + RA	0.227	0.456
RA + RD	0.218	0.399
RA + RN + RD	0.242	0.487
None	0.199	0.315

V. CONCLUSION AND FUTURE WORK

In this paper we proposed a pole-curb fusion vehicle localization system. The BnB-based pole localization tackles the intractable data association problem of pole well where global optimization method brings better robustness and motion decoupling greatly improves the efficiency. The proposed local grid map-based curb localization provides a novel representation of curbs which is more complete and precise. The experiments on KITTI dataset and TJ-TiEV dataset demonstrates the performance of the proposed method. The pole-curb fusion system achieves a RMSE in position of less than 0.20 m and in yaw angle of less than 0.40 ° in TJ-TiEV dataset.

The BnB-based pole localization is convenient to be extended to various landmarks that can be represented as 2D points and the local grid map-based representation can be used on other landmarks which can be formed as a curve or an area. The two methods covers almost all of the 2D landmarks for vehicle localization. Moreover, explicit correspondences of these landmarks are not required for the two methods. Thus, our future work is to extend the proposed system to a more complete system which contains more landmarks such as road markings, traffic sign or just a segment of point cloud.

REFERENCES

[1] G. Chen, H. Cao, J. Conradt, H. Tang, F. Rohrbein, and A. Knoll, "Event-based neuromorphic vision for autonomous driving: A paradigm shift for bio-inspired visual sensing and perception," *IEEE Signal Process. Mag.*, vol. 37, no. 4, pp. 34–49, Jul. 2020.

[2] M. Modsching, R. Kramer, and K. ten Hagen, "Field trial on gps accuracy in a medium size city: The influence of built-up," in *Proc. 3rd Workshop Positioning, Navigation Commun.*, 2006, vol. 2006, pp. 209–218.

[3] Y. Gu, L. Hsu, and S. Kamijo, "Gnss/onboard inertial sensor integration with the aid of 3-D building map for lane-level vehicle self-localization in urban canyon," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 4274–4287, Jun. 2016.

[4] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments," in *Robot., Sci. Syst.*, 2007, vol. 4, pp. 1.

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

[6] I. A. Barsan, S. Wang, A. Pokrovsky, and R. Urtasun, "Learning to localize using a lidar intensity map," in *Proc. Conf. Robot Learn.*, 2018, pp. 605–616.

[7] D. Chen, J. Weng, F. Huang, J. Zhou, Y. Mao, and X. Liu, "Heuristic monte carlo algorithm for unmanned ground vehicles realtime localization and mapping," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 10642–10655, Oct. 2020.

[8] S. Han, J. Kim, and H. Myung, "Landmark-based particle localization algorithm for mobile robots with a fish-eye vision system," *IEEE/ASME Trans. Mechatronics*, vol. 18, no. 6, pp. 1745–1756, Dec. 2013.

[9] L. Weng, M. Yang, L. Guo, B. Wang, and C. Wang, "Pole-based real-time localization for autonomous driving in congested urban scenarios," in *Proc. IEEE Int. Conf. Real-Time Comput. Robot.*, 2018, pp. 96–101.

[10] M. Sefati, M. Daum, B. Sondermann, K. D. Kreisköther, and A. Kampker, "Improving vehicle localization using semantic and pole-like landmarks," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2017, pp. 13–19.

[11] A. Schaefer, D. Büscher, J. Vertens, L. Luft, and W. Burgard, "Long-term urban vehicle localization using pole landmarks extracted from 3D lidar scans," in *Proc. IEEE Eur. Conf. Mobile Robots*, 2019, pp. 1–7.

[12] R. Spangenberg, D. Goehring, and R. Rojas, "Pole-based localization for autonomous vehicles in urban scenarios," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 2161–2166.

[13] D. Gruyer, R. Belaroussi, and M. Revilloud, "Accurate lateral positioning from map data and road marking detection," *Expert Syst. Appl.*, vol. 43, pp. 1–8, 2016.

[14] Y.-W. Seo and M. Hwangbo, "A computer vision system for lateral localization," *J. Field Robot.*, vol. 32, no. 7, pp. 1004–1014, 2015.

[15] A. Y. Hata, F. S. Osorio, and D. F. Wolf, "Robust curb detection and vehicle localization in urban environments," in *Proc. IEEE Intell. Veh. Symp. Proc.*, 2014, pp. 1257–1262.

[16] D. Cui, J. Xue, and N. Zheng, "Real-time global localization of robotic cars in lane level via lane marking detection and shape registration," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1039–1050, Apr. 2016.

[17] D. Kim, B. Kim, T. Chung, and K. Yi, "Lane-level localization using an avm camera for an automated driving vehicle in urban environments," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 1, pp. 280–290, Feb. 2017.

[18] B. Qin, Z. J. Chong, T. Bandyopadhyay, M. H. Ang, E. Frazzoli, and D. Rus, "Curb-intersection feature based monte carlo localization on urban roads," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 2640–2646.

[19] J. Kümmerle, M. Sons, F. Poggenhans, T. Kühner, M. Lauer, and C. Stiller, "Accurate and efficient self-localization on roads using basic geometric primitives," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 5965–5971.

[20] W.-C. Ma *et al.*, "Exploiting sparse semantic hd maps for self-driving vehicle localization," 2019, *arXiv:1908.03274*.

[21] L. Deng, M. Yang, B. Hu, T. Li, H. Li, and C. Wang, "Semantic segmentation-based lane-level localization using around view monitoring system," *IEEE Sensors J.*, vol. 19, no. 21, pp. 10077–10086, Nov. 2019.

[22] J. Suh, E. Y. Choi, and F. Borrelli, "Vision-based race track slam based only on lane curvature," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1495–1504, Feb. 2020.

[23] T. M. Breuel, "Implementation techniques for geometric branch-and-bound matching methods," *Comput. Vis. Image Understanding*, vol. 90, no. 3, pp. 258–294, 2003.

[24] J. Yang, H. Li, and Y. Jia, "Go-ICP: Solving 3D registration efficiently and globally optimally," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 1457–1464.

[25] R. I. Hartley and F. Kahl, "Global optimization through rotation space search," *Int. J. Comput. Vis.*, vol. 82, no. 1, pp. 64–79, 2009.

[26] Y. Liu, C. Wang, Z. Song, and M. Wang, "Efficient global point cloud registration by matching rotation invariant features through translation search," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 448–463.

[27] X. Li, Y. Liu, Y. Wang, C. Wang, M. Wang, and Z. Song, "Fast and globally optimal rigid registration of 3D point sets by transformation decomposition," 2018, *arXiv:1812.11307* .

[28] H. Yang and L. Carlone, "A polynomial-time solution for robust registration with extreme outlier rates," in *Proc. Robot., Sci. Syst.*, Freiburg im Breisgau, Germany, 2019.

[29] H. G. Jo, H. M. Cho, S. Jo, and E. Kim, "Efficient grid-based rao-blackwellized particle filter slam with interparticle map sharing," *IEEE/ASME Trans. Mechatronics*, vol. 23, no. 2, pp. 714–724, Apr. 2018.

[30] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2 d lidar slam," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1271–1278.

[31] K. Yoneda, C. Yang, S. Mita, T. Okuya, and K. Muto, "Urban road localization by using multiple layer map matching and line segment matching," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2015, pp. 525–530.

[32] C. Cabo, C. Ordoñez, S. García-Cortés, and J. Martínez, "An algorithm for automatic detection of pole-like street furniture objects from mobile laser scanner point clouds," *ISPRS J. Photogrammetry Remote Sens.*, vol. 87, pp. 47–56, 2014.

[33] S. Agarwal and K. Mierle, Ceres Solver: Tutorial Reference, Google Inc. [Online]. Available: http://ceres-solver.org

[34] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[35] B. R.Rusu, and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 1–4.

[36] J. Zhao *et al.*, "Tiev: The tongji intelligent electric vehicle in the intelligent vehicle future challenge of China," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 1303–1309.

[37] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3D registration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2009, pp. 3212–3217.

[38] A. M. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.