Robotica (2022), **40**, pp. 38–55
doi:10.1017/S0263574721000369

CAMBRIDGE
UNIVERSITY PRESS

RESEARCH ARTICLE

An efficient LiDAR-based localization method for self-driving cars in dynamic environments

Yihuan Zhang*, Liang Wang, Xuhui Jiang, Yong Zeng and Yifan Dai

Intelligent Connected Vehicle Center, Tsinghua Automotive Research Institute, Suzhou, China
*Corresponding author. Email: zhangyihuan@tsari.tsinghua.edu.cn

Received: 4 December 2020; **Revised:** 17 March 2021; **Accepted:** 18 March 2021; **First published online:** 20 April 2021

Keywords: Self-driving cars, Curb detection, Map matching, Real-time localization

Abstract

Real-time localization is an important mission for self-driving cars and it is difficult to achieve precise pose information in dynamic environments. In this paper, a novel localization method is proposed to estimate the pose of self-driving cars using a 3D-LiDAR sensor. First, the multi-frame curb features and laser intensity features are extracted. Meanwhile, based on the high-precision curb map generated offline, obstacles on road are detected using region segmentation methods and their features are removed. Furthermore, a map-matching method is proposed to match the features to the map, a robust iterative closest point algorithm is utilized to deal with curb features along with a probability search method dealing with intensity features. Finally, two separate Kalman filters are used to fuse the low-cost global positioning systems and map-matching results. Both offline and online experiments are carried out in dynamic environments and the results demonstrate the accuracy and robustness of the proposed method.

1. Introduction

Self-driving cars are developing rapidly to improve the driving safety and transportation efficiency. Self-driving cars are qualified in many scenarios that are dangerous, inconvenient for human drivers. One of the essential problems of self-driving technology is how to achieve high-precision pose of self-driving cars, which leads to the development of localization methods.

The differential global positioning systems (DGPS) and inertial measurement units (IMU) were normally used for localization system in the past few decades [1]. However, in most city driving scenarios, there are tall buildings, viaducts, and tunnels where the accuracy of DGPS cannot be guaranteed because of the lack of visible satellites. Simultaneous localization and mapping (SLAM) methods were commonly used in robots localization systems [2]. The loop detection is a key component of SLAM methods to correct the global position error. However, looped paths can hardly be guaranteed in most outdoor driving cases.

1.1. Related work on localization in static environment

Due to the availability of accurate digital navigation maps, and the onboard sensors like camera [3, 4], Radar or LiDAR gradually become a standard configuration. Researchers proposed many localization algorithms to obtain accurate position of vehicle by matching the data from onboard sensors and the digital map. In refs. [5, 6], two down-looking cameras mounted on both sides of the vehicle were used to detect the road markings and an Extended Kalman Filter (EKF) was then applied to couple the road marking information with the map data to estimate the lateral and directional deviations. The accurate longitudinal position of autonomous vehicles is also required in actual driving scenarios. A stereo camera system can solve the problem on how to recognize the lane markers and curbs. Then, an ICP algorithm

was used to match the pre-built accurate map data with the detected features [7, 8]. It compensates for the error of the horizontal and vertical position of the autonomous vehicle. However, under the assumption of a fixed vehicle posture and a flat ground, the position of the lane markings is detected and calculated. The intrinsic and extrinsic parameters sensitivity of camera were analyzed by ref. [9] and only use the lateral position of the lane markings to achieve vehicle position. However, the disadvantage is that it is difficult for the camera to accurately extract road features under shadow or insufficient light conditions.

The change of light intensity has little effect on the performance of LiDAR [10]. The point cloud data returned by LiDAR contains three-dimensional context information [11]. In ref. [12], the road planes and edge-like objects were extracted and matched by pattern align algorithm, but the correct matching rate was low because of the sparsity. In refs. [13, 14], another road feature, lane marking, was extracted by LiDAR sensor according to the extinguishable intensity value between lane marking and road surface. A lane-map-based localization method was proposed using lane markings and road surfaces as features. Generally speaking, in most urban driving scenes, the positions of curbs on both sides can define the road boundary, which can be used to distinguish traffic lanes and restricted areas. The curbs detected by a LiDAR have been used for localization [15]. Extract curbs is based on monocular cameras and LiDAR, and use particle filters to locate autonomous vehicles [16, 17]. The localization algorithm was performed under the assumption of flat road surface, which is hard to guarantee in most scenarios. In recent years, deep learning based methods have been proposed to show promising results in localization problem. A localization framework proposed by ref. [18] directly processes point clouds and estimates the vehicle's position accurately. Another deep neural network LocNet [19] was proposed to re-localize the vehicle globally after ICP registration in multisteps. Although the deep learning-based methods have shown the capability to match the performance of the state-of-the-art localization pipeline, they require high-cost hardware and long-time training to adapt in certain scenarios.

1.2. Related work on localization in dynamic environment

Due to the complex environment, the performance of feature detection algorithms is poor, and outliers usually exist in actual driving scenarios [20, 21]. A probabilistic grid map was adopted by refs. [22, 23] for localization so as to achieve the purpose of skipping the feature detection process. The point cloud data of LIDAR was projected into a grid image with a Gaussian intensity distribution. Then, the map-matching algorithm was performed by traversing the lateral and longitudinal search space. Instead of projecting the point cloud, raw data of a 2D-LiDAR [24] or 3D-LiDAR [25, 26] was directly matched with the pre-built dense 3D point cloud map using ICP algorithm. Nevertheless, the computational complexity of the raw data-based matching algorithms are high because all the sensor data are used for matching the map.

In the autonomous vehicles research community, the design of robust localization system is of great interest. Robustness of localization can be improved at different levels, including the processing of sensor measurement [27, 28], map matching [29], and adapting localization strategies [30]. Some diagnostic approaches were also integrated into localization system to detect the failures of localization [31, 32]. Semantic features and correspondence-based algorithms such as ICP are a popular choice for map matching. In order to reduce the influence of outliers, robust variants ICP algorithms such as M-ICP have been proposed [33, 34]. The cost function of M-ICP was designed under the assumption that the outliers are more far away from the map than inliers. The Fourier–Mellin transformation (FMT) has been used by ref. [35] for indoor applications. Ref. [36] expanded former works on FMT-based map matching by further adaptation of this method for the usage with dense grid maps. However, it is still a problem to robustly locate a autonomous vehicles under dynamic environment by matching sparse environment features.

In this paper, a 3D-LiDAR is utilized to sense the surrounding environment. Multi-frame curb features are generated based on vehicle dynamics. Meanwhile, the point cloud of obstacles are extracted from driving corridor which is calculated based on pre-build curb map. In the feature extraction procedure, the

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

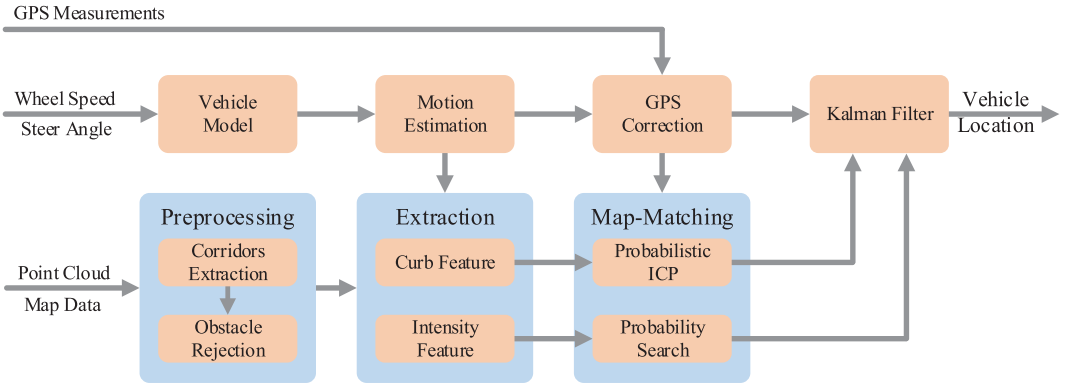

Figure 1. *The flowchart of the proposed method. The Wheel Speed and Steer Angle information are used to estimate the vehicle motion through dynamic model. The GPS measurements is used to correct the motion estimation result with global information. The Point Cloud and Map Data are used to locate the vehicle and two Kalman filters are applied to fuse the location results and output the final vehicle location.*

beam model and region segmentation methods are used to reject the outliers. Furthermore, a probabilistic ICP algorithm is proposed for matching map and estimating the ratio of outliers. Due to the reason that curb features lack of longitudinal characteristics in some driving scenarios, for example, driving on straight roads, the intensity features are also integrated into the localization system. Kalman filters were implemented for fusion of the map-matching results of curb and intensity features. The framework of the proposed method is shown in Fig. 1. This paper makes the following contributions:

1. A fast and accurate obstacle detection method is proposed to efficiently remove the obstacles on road and enhance the localization stability in dynamic environments.
2. A probabilistic ICP method considering the ratio of outliers is proposed, which improves the robustness of the localization results.
3. A Kalman filter-based optimization method is applied to integrate the curb and intensity features, which enhances the lateral and longitudinal accuracy. In addition, the whole framework is able to implement in real time, which is essential for autonomous vehicles.

The remainder of this paper is organized as follows. Section 2 describes map-based obstacles detection and outlier rejection algorithms. Section 3 details the probabilistic ICP and intensity matching algorithms. Section 4 evaluates the proposed method through comprehensive experiments. Section 5 summarizes the contribution of the paper and maps out the direction for future research.

2. Preprocessing and feature extraction

This section detailed the point cloud preprocessing module and feature extraction module. In this paper, the Velodyne HDL-32E LiDAR sensor has 32 lasers aligned vertically from $+10°$ to $-30°$, and it has a $360°$ horizontal field of view. A total of 700,000 points are generated each second in 70 m with 2 cm accuracy [37]. In this paper, the LiDAR is mounted on top of the vehicle as shown in Fig. 2(a). The raw data is shown in Fig. 2(b). The ith point is represented by $q_{c,i} = (x_{c,i}, y_{c,i}, z_{c,i}, r_{c,i})$. $r_{c,i}$ is the intensity value.

2.1. Preprocessing

When autonomous vehicles drives in the complex scenarios, the static or dynamic obstacles block the view of onboard sensors and influence the feature extraction algorithms significantly. Thus, the feature

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

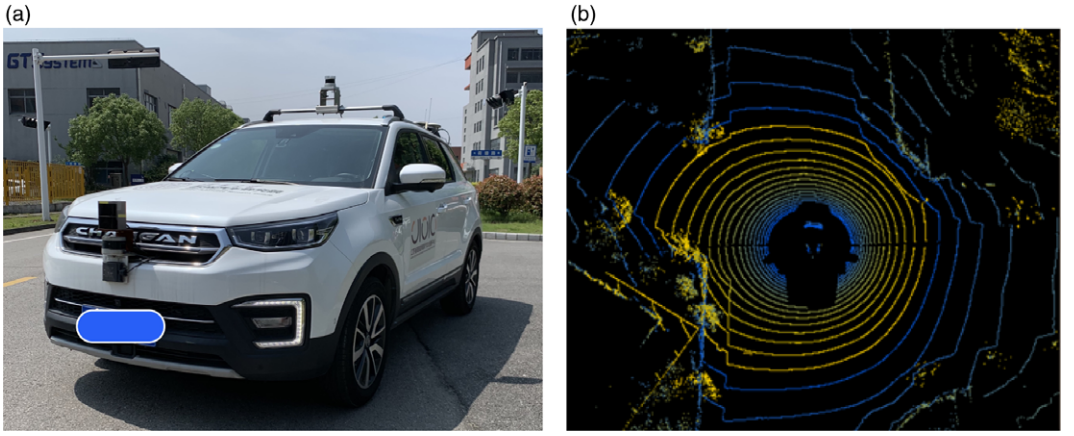

Figure 2. *Experimental platform. (a) Autonomous vehicles. (b) 3D point cloud.*

detection algorithms perform poorly and generate outliers which further make map-matching difficult. In this paper, the processing of sensor measurement is improved first. Different from deep leaning-based obstacle detection algorithms [38, 39, 40] that require large training data, we proposed a fast and accurate obstacle detection algorithm using a 3D-LiDAR based on the curb map.

2.1.1. Driving corridor extraction

The road boundary is defined by the position of curbs on both sides. Curb is an essential feature distinguishing the driving corridors and restricted areas, and is also significant for the safety of autonomous vehicles. Autonomous vehicles usually care about the obstacles inside the driving corridors. The driving corridors can be extracted based on curb map primarily. Curb map contains the vertices that are described in WGS-84 coordinates. $(B_{m,i}, l_{m,i})$ is the ith curb vertex coordinate in the map. The Gaussian projection algorithm is applied for transforming the curb vertices from WGS-84 coordinates to local LiDAR coordinates. The transformed ith curb vertex is denoted as $(x_{m,i}, y_{m,i})$. The driving corridors are usual irregular. In this paper, the driving corridors are represented by an occupied grid G with width w_g and height h_g. Each cell represents a $p_w \times p_h$ area where p_w and p_h are the width and height of each cell. Then, all the transformed curbs are projected into grid G. The occupied grid G is computed as follows:

$$\begin{cases} G(c_w, c_h) = 1 \ \ , \text{if curb fall in cell}(c_w, c_h) \\ G(c_w, c_h) = 0 \ \ , \text{otherwise} \end{cases} \tag{1}$$

where 1 and 0 represent that the cell is occupied and unoccupied, respectively.

The occupied grid G is donated by a binary image. After projecting curbs, image dilation operation is performed for expanding the shapes and filling up the gaps. Connected regions are labeled based on the dilated image and finally the driving corridors are extracted. The processing procedure is shown in Fig. 3.

2.1.2. Obstacle rejection

The point cloud of LiDAR is separated by the driving corridors and the ith point, which falls in driving corridor is denoted by $q_{d,i} = (x_{d,i}, y_{d,i}, z_{d,i}, r_{d,i})$. The separated points contain the obstacle part and ground part. The elevation of obstacles is commonly higher than the ground. Thus, the points are sorted by their elevation in ascending order and select $\xi\%$ points for fitting the ground. Parameter ξ is selected

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

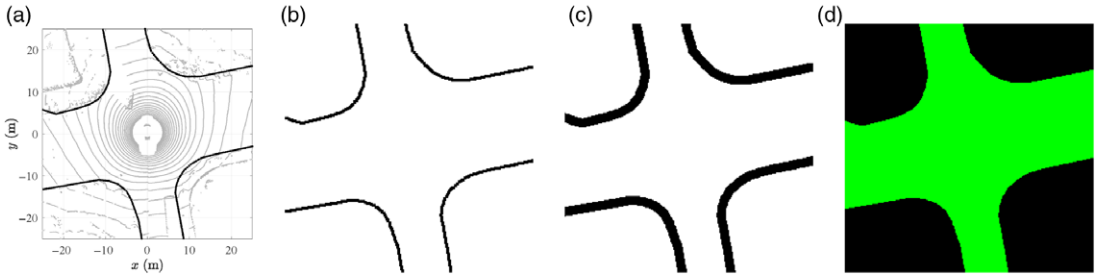

Figure 3. *Procedures of driving corridors extraction. In (a), the black dots represent the curbs of map that are transformed to local LiDAR coordinates. In (d), the green area is the extracted driving corridor. (a) Curb map. (b) Grid image. (c) Dilated image. (d) Driving corridors.*

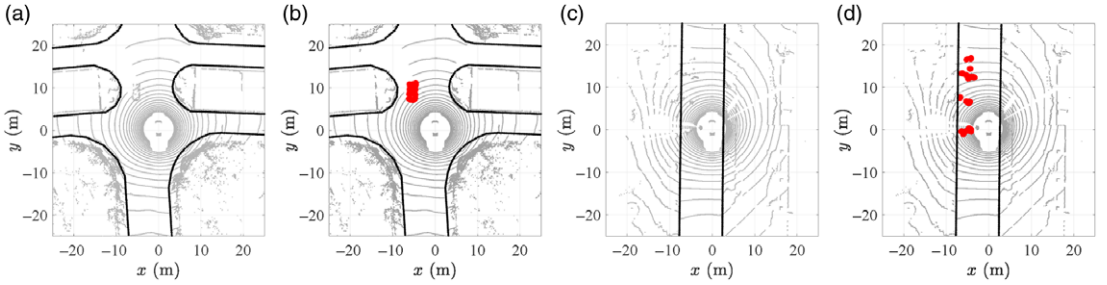

Figure 4. *Obstacle detection results. The red dots represent the detected points of obstacles. (a) Raw data. (b) Obstacles. (c) Raw data. (d) Obstacles.*

according to the statistics information of obstacle points. A flat plane model is applied for fitting the ground surface using least square method:

$$\begin{cases} a_0 \sum_i^{n_d} x_{d,i}^2 + a_1 \sum_i^{n_d} x_{d,i} y_{d,i} + a_2 \sum_i^{n_d} x_{d,i} = \sum_i^{n_d} x_{d,i} z_{d,i} \\ a_0 \sum_i^{n_d} x_{d,i} y_{d,i} + a_1 \sum_i^{n_d} y_{d,i}^2 + a_2 \sum_i^{n_d} y_{d,i} = \sum_i^{n_d} y_{d,i} z_{d,i} \\ a_0 \sum_i^{n_d} x_{d,i} + a_1 \sum_i^{n_d} y_{d,i} + a_2 n_d = \sum_i^{n_d} z_{d,i} \end{cases} \tag{2}$$

where a_0, a_1, a_2 are the coefficients of plane equation: $z = a_0 x + a_1 y + a_2$. n_d is the number of selected points. Then, the obstacle points are extracted based on distance criteria:

$$\begin{cases} d_i = \dfrac{|a_0 x_{d,i} + a_1 y_{d,i} - z_{d,i} + a_2|}{\sqrt{a_0^2 + a_1^2 + 1}} \\ L_{d,i} = \text{obstacle} \quad \text{if } d_i > d_\sigma \\ L_{d,i} = \text{ground} \quad \text{otherwise} \end{cases} \tag{3}$$

where $L_{d,i}$ is the label of point $q_{d,i}$. d_i is distance of ith point to the fitting plane. d_σ is the preset threshold. The detection results are shown in Fig. 4. In this paper, the classes of obstacles are not cared. The detected points of obstacles are eliminated from the raw point cloud and the remained point cloud is feed to the next processings.

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

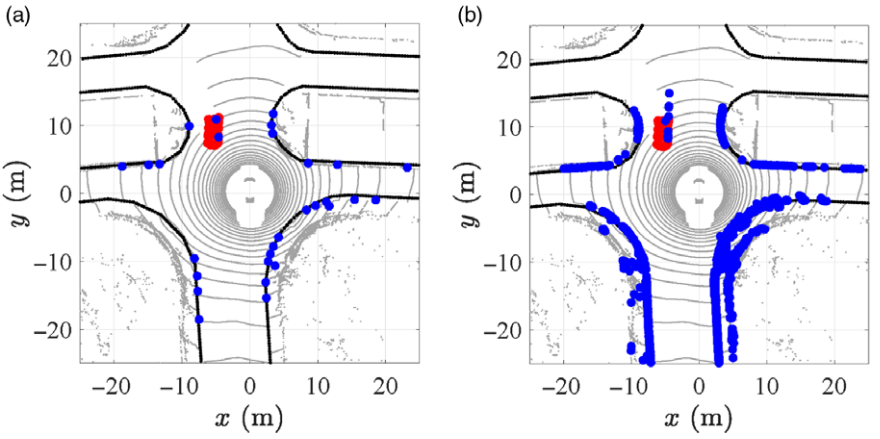

Figure 5. *Curb detection comparison between single frame and multi-frame. (a) Single-frame curb. (b) Multi-frame curb.*

2.2. Features extraction

The features used in this paper are curb and intensity features. In this subsection, the extraction methods of these features are detailed.

2.2.1. Curb feature detection

After eliminating the dynamic obstacles, an improved curb detection method based on ref. [41] is proposed. According to the previous method, the multi-frame point cloud data is used to multiply the curb in each frame. As the moving information of the vehicle is denoted as d_x, d_y, and d_ψ on the x-y plane. The detected curbs of single frame are denoted by $\mathbf{C}_k$, where the coordinates are denoted as $\begin{bmatrix}\mathbf{x}_k & \mathbf{y}_k\end{bmatrix}^T$. The transformation is used to generate the coordinates of multi-frame curbs:

$$\mathbf{C} = \left[\mathbf{C}_k \, f(\mathbf{C}_{k-1}) \, f^2(\mathbf{C}_{k-2}) \, \cdots \, f^n(\mathbf{C}_{k-n})\right] \tag{4}$$

where $f^n(\mathbf{C}) = \underbrace{f(f(\cdots f}_{n}(\mathbf{C})))$. The transformation f is defined by

$$f(\mathbf{C}) \triangleq \begin{bmatrix}\cos(d_\psi) & -\sin(d_\psi) \\ \sin(d_\psi) & \cos(d_\psi)\end{bmatrix} \left(\mathbf{C} - \begin{bmatrix}d_x \\ d_y\end{bmatrix}\right) \tag{5}$$

An example of single-frame curbs and multi-frame curbs is shown in Fig. 5.

The connection regions between obstacles and ground are similar to curbs. Thus, the false detection of curb features are often occurred around the obstacles. In this paper, the obstacle regions are segmented based on distance criteria for eliminating the false detection. The obstacle regions are defined that the distance of every point in the region to the closest obstacle is less than d_o. Based on KD-tree method, the detected curb features which are within the obstacle regions are recognized as outliers and elimated.

After the extraction of the curbs and rejection of the outliers, a beam model [27] is used to generate a smooth curb contour for localization. In this paper, a set of beam models are applied in each step with multi-frame. The angular resolution is denoted as $\delta = 2\pi/n_b$ and the launching point of beam model is set by the position of the vehicle. The full equations are denoted as follows:

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

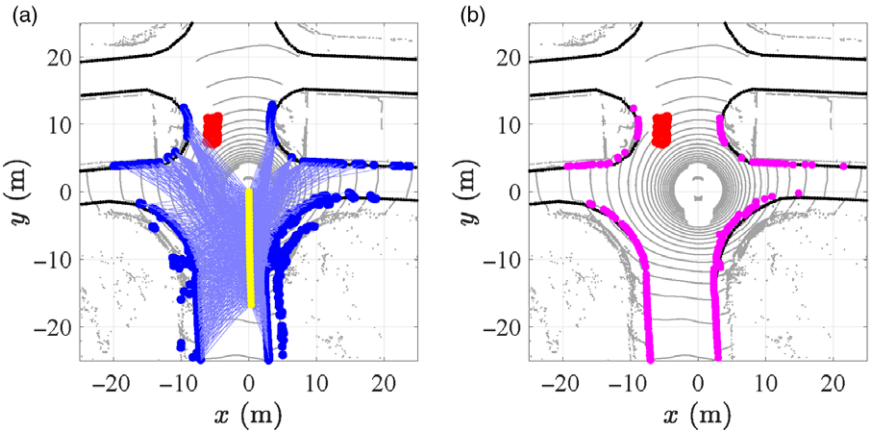

Figure 6. *Beam model method. The yellow dots represent the trajectory points of the autonomous vehicles and the light blue lines represent the beam lines launched at the each trajectory point. The pink dots represent the extracted curbs using the beam model method. (a) Beam launching. (b) Extracted curbs.*

$$\mathbf{Z}_k = \left\{ \frac{(k-1)\cdot\pi}{n_b} < \arctan\left(\frac{y - y_{1,j}}{x - x_{1,j}}\right) \le \frac{k\cdot\pi}{n_b}, \right.$$
$$\left. -25 \le x, y \le 25, k = 1, 2, \cdots, n_b \right\} \tag{6}$$

$$I_k = \left\{ \arg \min_{q_{b,i} \in \mathbf{Z}_k} \sqrt{x_i^2 + y_i^2} \right\} \tag{7}$$

where $q_{b,i} = (x_i, y_i)$ is the ith curb coordinate in $\mathbf{C}$. $\mathbf{Z}_k$ means the kth beam area. The coordinate $(x_{1,j}, y_{1,j})$ is the jth launching point of beam model. I_k is the index of the curb with the shortest distance among the curbs in $\mathbf{Z}_k$. Thus, all the I_kth curbs are extracted to represent the contour. The procedure is shown in Fig. 6.

2.2.2. Intensity feature generation

Although the curb features have a good performance of correcting the error of lateral position. In some straight road, curbs lack longitudinal feature for localization. In this paper, the intensity measurements of LiDAR are combined with curb features for enhancing the localization system. The pre-built intensity map is represented as a grid, which is considered as probability distributions over environment described in ref. [22]. The mean and variance of intensity are contained in each cell of the map. The advantage of this representation is an increased robustness to dynamic obstacles. Because the dynamic obstacles can cause large intensity variances for the cells in which they pass.

For intensity feature extraction, a three-dimensional grid S is established with width w_s and height h_s. The cell (i, j) contains three elements $S_{n,(i,j)}$, $S_{r,(i,j)}$, and $S_{\sigma,(i,j)}$. Where $S_{n,(i,j)}$ is the number of points falling in cell (i, j), $S_{r,(i,j)}$, and $S_{\sigma,(i,j)}$ are the mean and variance of intensity. The grid S is calculated using the following iterative equations:

$$\begin{cases} S'_{n,(i,j)} \leftarrow S_{n,(i,j)} + 1 \\ S'_{\sigma,(i,j)} \leftarrow \dfrac{S_{n,(i,j)} - 1}{S_{n,(i,j)}} S_{\sigma,(i,j)} + \dfrac{\left(r_{c,i} - S_{r,(i,j)}\right)^2}{S_{n,(i,j)} + 1} \\ S'_{r,(i,j)} \leftarrow S_{r,(i,j)} + \dfrac{r_{c,i} - S_{r,(i,j)}}{S_{n,(i,j)} + 1} \end{cases} \tag{8}$$

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

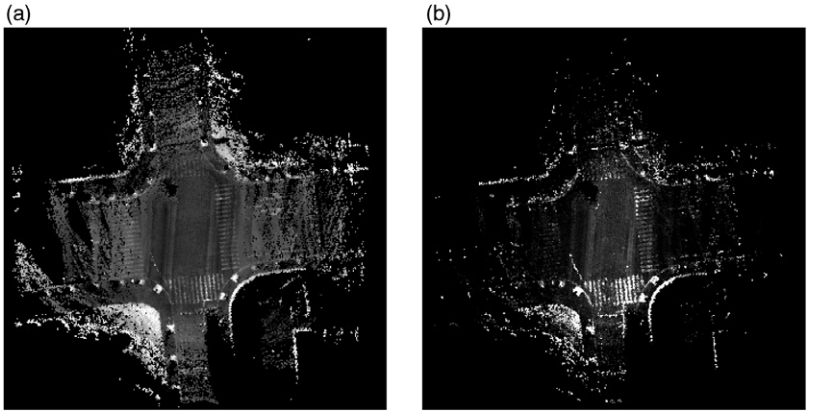

Figure 7. *Intensity grid. Due to the laser reflection, different materials show different intensities. For example, the cross-walk area has higher intensity value than the road area. (a) Mean intensity. (b) Variance intensity.*

where $r_{c,i}$ is the intensity value of ith point. $S'_{n,(i,j)}$, $S'_{\sigma,(i,j)}$ and $S'_{r,(i,j)}$ are new iterative values. The result is shown in Fig. 7.

3. Map matching and localization

The map-matching process intends to estimate the deviation between features detected by the autonomous vehicles and features provided by the digital map. In this section, the probabilistic ICP and Area probability Search (APS) algorithms are utilized for map matching of curb and intensity features, respectively.

3.1. Probabilistic ICP

The extracted curbs and curbs in map are represented as two point clouds. The matching procedure of curbs is formulated as a point cloud registration problem. The ICP algorithm is a general matching algorithm proposed by Besl [42]. The extracted curb points are denoted by $\mathbf{C}$ and the curb points in the map are denoted by $\mathbf{M}$. The standard ICP algorithm is to find a transformation $\mathbf{T}$ by minimizing the cost function:

$$J = \sum_{i}^{n_c} d\left(\mathbf{TC}_i, \mathbf{M}\right) \tag{9}$$

where d represents the Euclidean distance. The iterative process is denoted as follows. First, a correspondence of $\mathbf{C}_i$ in map $\mathbf{M}$ is found by a k-dimensional tree search. Then the transformation of each correspondence is computed based on the singular value decomposition (SVD) [43]. After that, the cost J is calculated based on the transformation $\mathbf{C} = \mathbf{TC}$. Finally, the iteration is terminated when the cost J falls below a pre-defined threshold τ.

However, the standard ICP algorithm cannot deal with the cases where outliers exist in point cloud $\mathbf{C}$. M-ICP is a robust variant of ICP algorithm which designs another cost function to reduce the influence of outliers. The cost function of M-ICP is designed under the assumption that the outliers are more far away from the map than inliers. Due to the initial position error of vehicle, this assumption is unreasonable for map matching. The outliers should be identified by whether they are consistent with the map. In this paper, an improved ICP algorithm is proposed to reduce the influence of outliers as much as possible through a double-layer iteration processing. The first layer called local ICP iteration is designed for

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

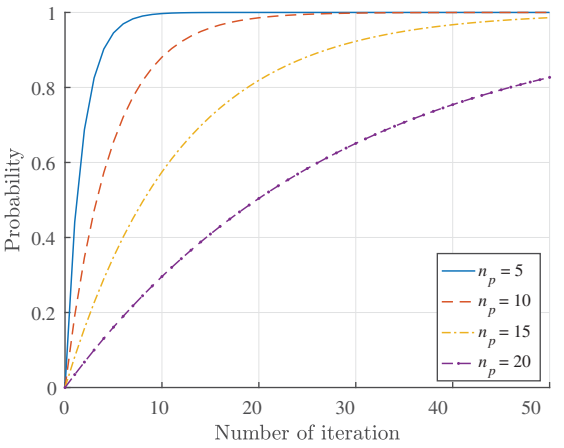

Figure 8. *Relation p_t of n_c and n_t, at the assumption of $n_c = 300$ and $n_f = 45$, which are determined by statistic information.*

finding a relative accurate transformation matrix based on a probabilistic method. The second layer called global ICP iteration aims to fine tune the result obtained by the first layer and evaluate the quality of iteration.

At the *i*th iteration, the feature points are randomly sampled by local ICP to calculate rough transformation matrix $\mathbf{T}_{l,i}$ and initialize the global ICP. In the processing of global ICP, the $\mathbf{T}_{l,i}$ is first applied to the curb features. Then, the outliers of curb features are eliminated based on the distance to the map. The matrix $\mathbf{T}_{l,i}$ is fine tuned and ratio of outliers is calculated to evaluate the quality of the iteration. The proposed algorithm is performed under the assumption that if the accurate $\mathbf{T}_{g,i}$ is obtained by global ICP, the more inliers will be consistent with the map and the outlier ratio is lower. The probability of the proposed algorithm is analysed as follows.

The number of randomly selected curbs for local ICP denoted as n_p and number of iteration denoted as n_t are needed to choose. The total number of curbs is denoted as n_c, number of outliers is n_f. Thus, in the *i*th iteration, the probability of all randomly selected curbs are inliers denoted as p_r is calculated:

$$p_r = \Pi_{i=0}^{n_p-1} \frac{n_c - n_f - i}{n_c - i} \tag{10}$$

In the iteration of n_t, the probability of inliers are denoted as p_t, note that the inliers are the curbs which at least occurred once in the previous iterations.

$$p_t = 1 - (1 - p_r)^{n_t} \tag{11}$$

Figure 8 shows the relation p_t of n_c and n_t.

At the *i*th iteration, the outliers are recoginzed as follows:

$$\begin{cases} \mathbf{C}_i = \text{outlier} & \text{if } dist\left(\mathbf{T}_{g,i}\mathbf{C}_i, \mathbf{M}\right) > d_g \\ \mathbf{C}_i = \text{inlier} & \text{otherwise} \end{cases} \tag{12}$$

where d_g is the distance threshold. After all iterations, the final result outputted by probabilistic ICP is computed as

$$\mathbf{T} = \frac{\sum_{i}^{n_t} \eta_i \cdot \mathbf{T}_{g,i}}{\sum_{i}^{n_t} \eta_i} \tag{13}$$

and η_i is the ratio of outliers at the *i*th iteration. Finally, $\hat{p}_c(t) = \left[\hat{x}_c(t)\ \hat{y}_c(t)\ \hat{\psi}_c(t)\right]^{\mathrm{T}}$ is the observation of map matching of curb features computed based on $\mathbf{T}$.

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

3.2. Intensity feature matching

The intensity feature matching is implemented by APS algorithm [22]. To calculate the probability of offset (x_o, y_o) given sensor data z and intensity map data m, the product is token over all cells of the probability of observing the sensor data cell's average intensity given the map cell's average intensity and both of their variances. The mean and variance of intensity of cell (i, j) in the intensity map are denoted as $m_{r,(i,j)}$ and $m_{\sigma,(i,j)}$. The effective area in which S_n of cells greater than n_e is considered for calculation. Thus, the possibility of vehicle locating at offset (x_o, y_o), which is related to current position is computed as

$$P(x_o, y_o|z, m) = \prod_{i,j} \exp\left(\frac{-(m_{r,(i-x_o,j-y_o)} - s_{r,(i,j)})^2}{2(m_{\sigma,(i-x_o,j-y_o)} + s_{\sigma,(i,j)})^2}\right)^{\alpha} \tag{14}$$

where α is a parameter that determines the shape of the probabilistic distribution. At the foundation of curb features, the lateral searching space is restricted within 1 m and longitudinal searching space is restricted within 3 m. Thus, the processing of intensity feature matching is fast and capable for real-time computation. The center of mass of the distribution $P(x_o, y_o|z, m)$ is recognized as the most likely position:

$$x_o = \frac{\sum_{x_o,y_o} P(x_o, y_o|z, m) \cdot x_o}{\sum_{x_o,y_o} P(x_o, y_o|z, m)} \tag{15}$$

$$y_o = \frac{\sum_{x_o,y_o} P(x_o, y_o|z, m) \cdot y_o}{\sum_{x_o,y_o} P(x_o, y_o|z, m)} \tag{16}$$

$$\hat{x}_s(t) = x_v(t-1) + x_o \tag{17}$$

$$\hat{y}_s(t) = y_v(t-1) + y_o \tag{18}$$

where $p_v(t-1) = \left[x_v(t-1)\ y_v(t-1)\ \psi_v(t-1)\right]^{\mathrm{T}}$ is the position of autonomous vehicles at time $t-1$. $\hat{p}_s(t) = \left[\hat{x}_s(t)\ \hat{y}_s(t)\right]^{\mathrm{T}}$ is the observation of map matching of intensity features.

3.3. Localization optimization

After obtaining the observations of map matching of curb and intensity features, Kalman filter fusion framework is employed for fusion of observations and provide an optimized location of the autonomous vehicle. The predicted vehicle position is calculated based on the vehicle dynamic model:

$$\begin{cases} \dot{\beta} = \dfrac{2C_f}{Mv_x}\left[\delta_f - \beta - \dfrac{l_f\dot{\psi}}{v_x}\right] + \dfrac{2C_r}{Mv_x}\left[-\beta + \dfrac{l_f\dot{\psi}}{v_x}\right] - \dot{\psi} \\ \ddot{\psi} = \dfrac{2l_fC_f}{I_z}\left[\delta_f - \beta - \dfrac{l_f\dot{\psi}}{v_x}\right] - \dfrac{2l_rC_r}{I_z}\left[-\beta + \dfrac{l_f\dot{\psi}}{v_x}\right] \\ \dot{x} = v_x\cos\psi - v_x\tan\beta\sin\psi \\ \dot{y} = v_x\sin\psi + v_x\tan\beta\cos\psi \end{cases} \tag{19}$$

where C_f, C_r are the stiffnesses of front and back wheels. l_f, l_r are the distances from mass to front and back wheels. β, v_x, I_z, M, and δ_f represent the slip angle of vehicle, longitudinal speed, moment of inertia, weight of vehicle, and front wheel angle. The vehicle dynamic equations are solved through Euler's formula. $p'_v(t) = \left[x'_v(t)\ y'_v(t)\ \psi'_v(t)\right]^{\mathrm{T}}$ is the predicted vehicle position at time t. The predicted

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

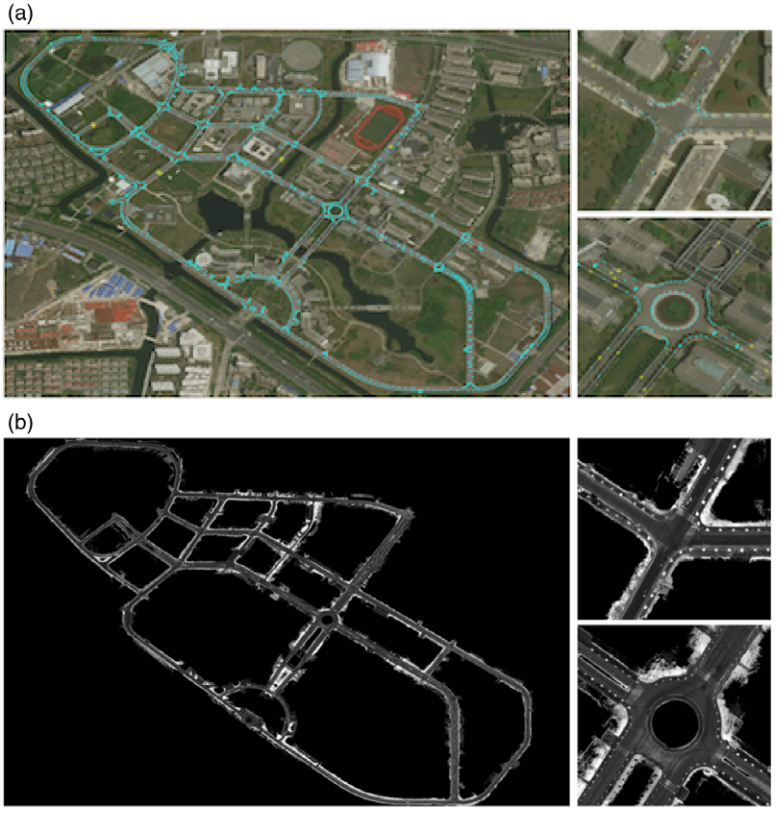

Figure 9. *High-precision digital map. In (a), the blue dots represent the road boundary, which are used as the reference of the proposed method. (a) Curb map. (b) Intensity map.*

pose of the vehicle is measured by low-cost GPS, and it is optimized through the first Kalman filter which is denoted by:

$$p'_{v,\text{gps}}(t) = \left[x'_{v,\text{gps}}(t)\ y'_{v,\text{gps}}(t)\ \psi'_{v,\text{gps}}(t)\right]^T \tag{20}$$

The second Kalman filter is applied to optimize the relative pose by integrating the results of map matching. The optimized pose is denoted as $p'_{v,\text{gps}}(t)$, and the states of the second Kalman filter is $\left[x\ y\ \psi\right]^T$ and the observation vector is $\left[\hat{x}_c(t)\ \hat{y}_c(t)\ \hat{\psi}_c(t)\ \hat{x}_s(t)\ \hat{y}_s(t)\right]^T$. Both online and offline experiments are carried out to demonstrate the advantages of the proposed method.

4. Experiments

4.1. Implementation details

The experiments are divided into two parts: offline and online. In the offline experiment, the proposed localization method is tested on static and dynamic environments. Before testing the localization method, a high-precision map is created with two basic layers: curb layer and intensity layer. The map is shown in Fig. 9, the size of the map is around 1800 m × 1200 m. All reference positions of the vehicle is acquired by the real-time kinematic (RTK) GPS, which has an average accuraty of 3 cm. The key parameters used in the proposed method are shown in Table I. The estimated position is projected with the origin of the ground truth and a set of metrics including root mean square error (RMSE) of lateral, longitudinal, and heading are reported for evaluation.

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

Table I. *Key parameters used in the proposed method.*

Parameters	Definitions	Values	Units
w_g	Width of grid image G	400	–
h_g	Height of grid image G	400	–
p_w	Width of cell in G	0.2	m
p_h	Height of cell in G	0.2	m
ξ	Points for ground fitting	95	%
d_σ	Distance to the ground	0.5	m
w_s	Width of grid image S	200	–
h_s	Height of grid image S	200	–
d_o	Distance to the obstacle	0.5	m
n_b	Number of beams	36	–
n_p	Number of randomly select curbs	15	–
n_t	Number of ICP iteration	30	–
d_g	Distance to the map	1.0	m
C_f/C_r	Stiffness of front/back wheels	45,980	–
l_f	Distance from mass to front wheel	1.4	m
l_r	Distance from mass to back wheel	1.2	m
I_z	Moment of inertia	6000	$kg \cdot m^2$
M	Weight of vehicle	2300	kg

Figure 10. *Trajectory in static environment.*

4.2. Experiment results

4.2.1. Static environment experiment

For the experiment in static environment, the dataset is recorded by our autonomous vehicle on the campus road in Tongji University. The recorded dataset composed of 4260 frames contain few static or dynamic obstacles. The trajectory of autonomous vehicles is shown in Fig. 10.

We compare the proposed method with those only using low-cost GPS, standard ICP based on the ref. [44], ICP-obs that is further integrated with obstacle detection module, M-ICP method [33] and FMT-based method [36]. The objective function of M-ICP is used as follows:

$$J' = \sum \rho\left(\sqrt{e_x^2 + e_y^2}\right) \tag{21}$$

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

Table II. *Error comparison of static environment.*

Method	Lateral (m)	Longitudinal (m)	Heading (°)
Low-cost GPS	1.9942	2.5449	1.5034
ICP [44]	0.2131	0.4436	0.7184
ICP-obs	0.1943	0.4353	0.6478
M-ICP [33]	**0.1855**	0.4074	0.5766
Intensity [22]	0.2983	0.4145	–
FMT [36]	0.4101	0.5905	0.8929
Proposed	0.1885	**0.3933**	**0.5195**

Note: Boldface means the best results in comparison.

Table III. *Error comparison of dynamic environment.*

Method	Lateral (m)	Longitudinal (m)	Heading (°)
Low-cost GPS	1.9509	1.8668	2.1533
ICP [44]	0.5812	2.6381	3.1461
ICP-obs	0.2210	0.4990	0.8338
M-ICP [33]	0.2035	0.4964	0.9514
Intensity [22]	0.3130	0.5614	–
FMT [36]	0.7467	0.8975	0.8356
Proposed	**0.1884**	**0.4568**	**0.5283**

Note: Boldface means the best results in comparison.

where e_x and e_y are the residual errors of map matching in x-y plane. $\rho(\cdot)$ is a biweight function, and B is set to be 4.681 that fits the minimized normal distribution:

$$\rho(e) = \begin{cases} \frac{B^2}{2}\left(1-\left(1-\left(\frac{e}{B}\right)^2\right)^3\right) & (if\ |e| \le B) \\ \frac{B^2}{2} & (if\ |e| \ge B) \end{cases} \tag{22}$$

The FMT-based method is implemented based on grid image. For comparison, grid images are generated and the curb features are projected into the grid image. According to ref [36], the width, height, and precision of the grid images are set as 500, 500, and 0.1 m. The simulation is implemented with MATLAB. The statistical results of six methods are reported in Table II. In static driving environment, the ICP method based on our previous work achieves a satisfied result. By integrating the obstacle detection module, the result has a little improvement. Comparing to other methods, the result of proposed method is similar to M-ICP. Due to the reason that the curb feature is sparse, the FMT-based method is unstable for map matching.

4.2.2. Dynamic environment experiment

For the experiment in dynamic environment, the dataset is recorded at daytime in the campus. The dataset composed of 4513 frames contain many pedestrian, trucks, and cars. More than half of frames contain at least one obstacle. Also, there are different driving scenarios such as normal roads, intersections and one-way roads in the dataset. The localization methods are implemented as the same as the methods in static environment experiment. The curves of lateral, longitudinal, and heading error of localization are shown in Fig. 11 and Table III is the statistical results.

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

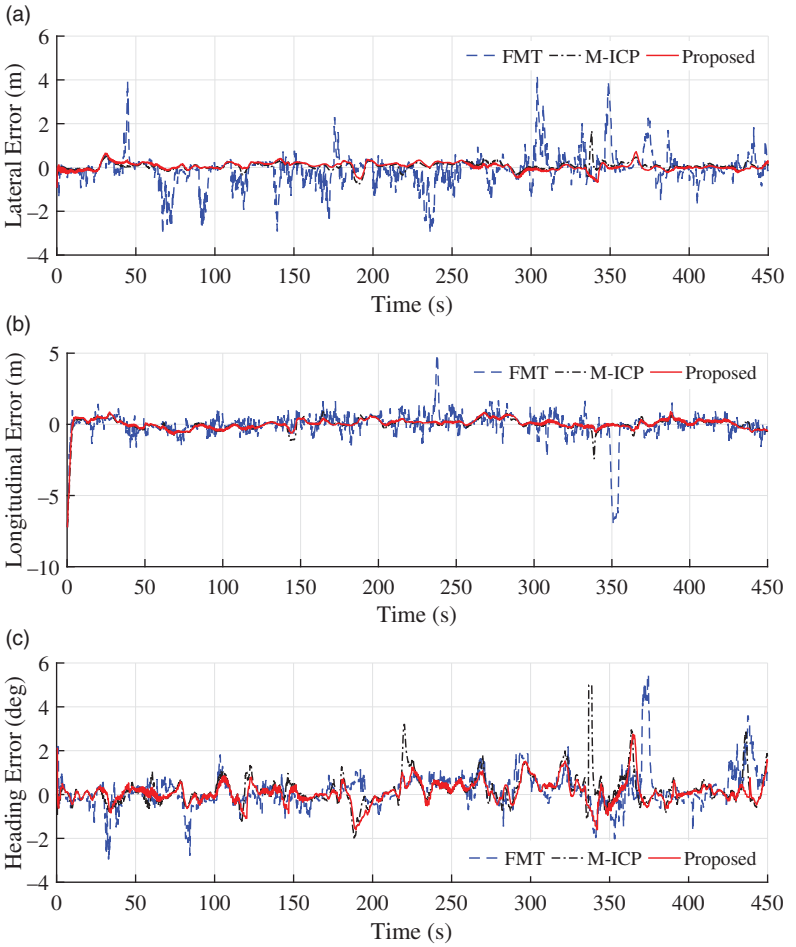

Figure 11. *Curves of localization error in dynamic environment. (a) Lateral error. (b) Longitudinal error. (c) Heading error.*

Based on the experiment results, it can be seen that the M-ICP and FMT-based methods are unstable when there are moving objects. With the rejection of the obstacles in the proposed method, the results are more stable and accurate in dynamic environments. The M-ICP and FMT-based method failed to locate the vehicle at about 330 and 340 s, respectively, as shown in Fig. 12. However, the proposed method can deal with the cases robustly and achieve a precision localization with average 19 cm lateral error, 46 cm longitudinal error, and 0.53° heading error. The whole trajectory with position error is shown in Fig. 13.

4.2.3. Online experiment

The aim of using online experiment system is to test the ability of real-time computing. In autonomous vehicle system, the real-time performance is essential and the computing platform is limited. There is no other mathematical model running inside online experiment, and this is the real performance running in real time on our vehicle platform. The proposed algorithm is implemented with C/C++ in the Robot Operating System (ROS). The online experiment is tested on our autonomous vehicles platform. The controller is the ADLINK Industrial Personal Computer (IPC) with 16GB of RAM and Intel Core i7-3610QE CPU at 2.3GHz. In addition, the probabilistic ICP is implemented using parallel programming

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

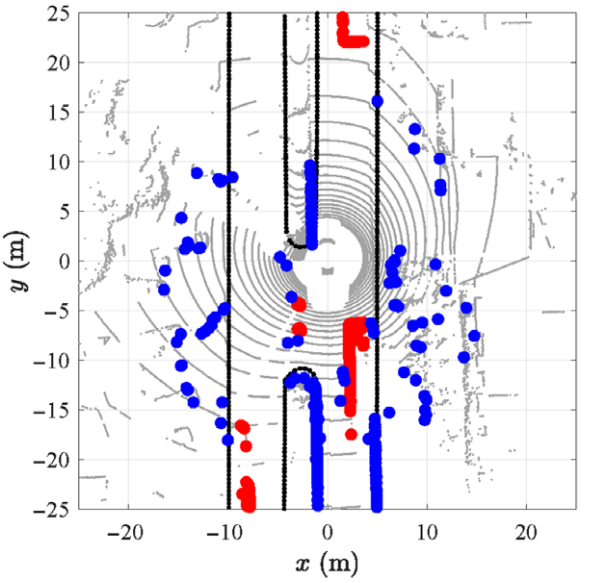

Figure 12. *Scenario at 330.6 s with a lot outliers. The coach represented by the red points blocks the right side curbs.*

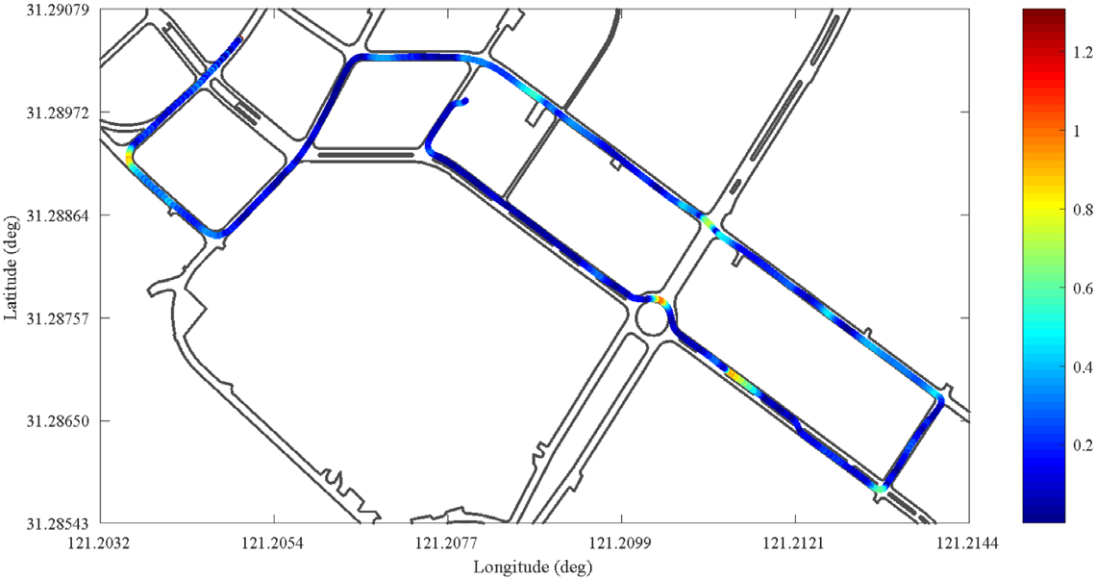

Figure 13. *Trajectory with position error. The unit of the color bar is meter.*

and the average processing period of the proposed method is around 60 ms, which is possible for the real-time operation of autonomous vehicles. The online experiment is carried out on our campus. In Fig. 14, the blue dots represent the curb features. The yellow dots represent the detected obstacle points. The position of blue car model is received from low-cost GPS and the position based on proposed method is labeled in a red car model.

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

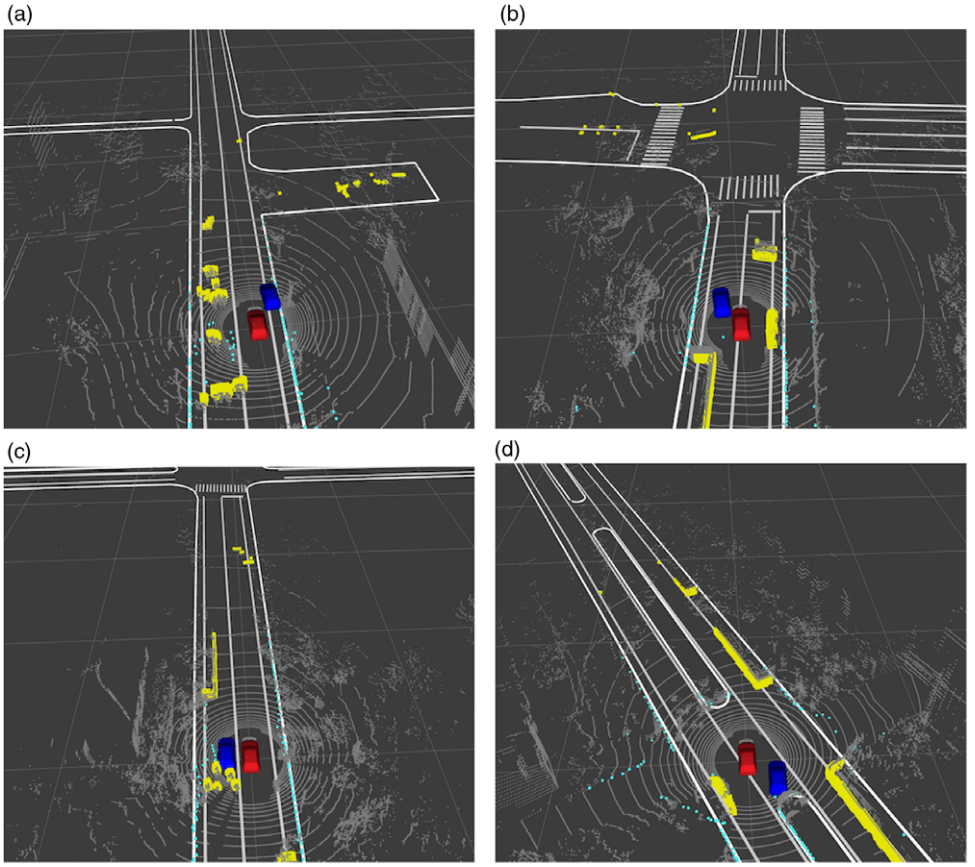

Figure 14. *Real-time localization results. The road boundary, lane line, and cross-walk are denoted in white lines in the visualization.*

5. Conclusion and future work

For autonomous vehicles, this paper develops a robust method to locate an autonomous vehicles using a Velodyne 3D-LiDAR. The obstacles are detected based on curb map to form the foundation of feature extraction. Then, curb and intensity features are extracted from the environment, and region segmentation and beam model method are utilized to reject the outliers as much as possible in the feature extraction procedure. The probabilistic ICP algorithm is proposed to match the high-precision map to increase the robustness of localization system. The simulation and real-time experiments demonstrates that the vehicle can be robustly located. A precision localization is achieved with average 19 cm lateral error, 46 cm longitudinal error, and 0.53° heading error in the dynamic environment.

However, the proposed algorithm depends on a rough position that is provided by a low-cost GPS, and some route-based localization methods can be integrated into the proposed method to reduce the dependence of the GPS. The proposed method also has the limitation in locating the vehicle where there is no curb. In the future work, other environmental features like road markings will be detected to improve the robustness of the localization system.

Acknowledgments. The authors declare no competing interests. This work was supported in part by the National Key Research and Development Program of China under Contract 2018YFB0105000 and in part by the Natural Science Foundation of Jiangsu Province under Contract BK20200271 and Suzhou Science and Technology Project under Contract SYG202014. We would like to thank VeCaN lab in Tongji University for data support and fruitful discussion.

https://doi.org/10.1017/S0263574721000369 Published online by Cambridge University Press

References

[1] J. Knaup and K. Homeier, "Graph-based Environmental Modelling and Function Independent Situation Analysis for Driver Assistance Systems," *Proceedings of the IEEE International Conference on Intelligent Transportation Systems* (2010) pp. 428–432.

[2] J. Levinson, M. Montemerlo and S. Thrun, "Map-based precision vehicle localization in urban environments," *Rob. Sci. Syst.* **4**(2007), 1–6 (2007).

[3] W. Maddern and P. Newman, "Real-Time Probabilistic Fusion of Sparse 3D Lidar and Dense Stereo," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2016) pp. 2181–2188.

[4] X. Elias, Z. Paraskevi and N. Andreas, "Path planning and scheduling for a fleet of autonomous vehicles," *Robotica* **34**(10), 2257–2267 (2016).

[5] D. Gruyer, R. Belaroussi and M. Revilloud, "Map-Aided Localization with Lateral Perception," *Proceedings of IEEE Intelligent Vehicles Symposium* (2014) pp. 674–680.

[6] N. Mattern and G. Wanielik, "Camera-based Vehicle Localization at Intersections Using Detailed Digital Maps," *Proceedings of IEEE/ION Position, Location and Navigation Symposium* (2010) pp. 1100–1107.

[7] M. Schreiber, C. Knöppel and U. Franke, "Laneloc: Lane Marking Based Localization Using Highly Accurate Maps," *Proceedings of IEEE Intelligent Vehicles Symposium* (2013) pp. 449–454.

[8] 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.* **17**(4), 1039–1050 (2016).

[9] Z. Tao, P. Bonnifait, V. Fremont and J. Ibanez-Guzman, "Lane Marking Aided Vehicle Localization," *Proceedings of the IEEE International Conference on Intelligent Transportation Systems* (2013) pp. 1509–1515.

[10] Á. Llamazares, E. Molinos and M. Ocaña, "Detection and tracking of moving obstacles (DATMO): A review," *Robotica* **38**(5), 761–774 (2020).

[11] R. Giulio, M. Annalisa and W. Rainer, "LIDAR and stereo combination for traversability assessment of off-road robotic vehicles," *Robotica* **34**(12), 2823–2833 (2016).

[12] A. Schlichting and C. Brenner, "Localization Using Automotive Laser Scanners and Local Pattern Matching," *Proceedings of IEEE Intelligent Vehicles Symposium* (2014) pp. 414–419.

[13] D. Kim, T. Chung and K. Yi, "Lane Map Building and Localization for Automated Driving Using 2D Laser Rangefinder," *Proceedings of IEEE Intelligent Vehicles Symposium* (2015) pp. 680–685.

[14] R. Matthaei, G. Bagschik and M. Maurer, "Map-Relative Localization in Lane-Level Maps for ADAS and Autonomous Driving," *Proceedings of IEEE Intelligent Vehicles Symposium* (2014) pp. 49–55.

[15] A. Hata, F. Osorio and D. Wolf, "Robust Curb Detection and Vehicle Localization in Urban Environments," *Proceedings of IEEE Intelligent Vehicles Symposium* (2014) pp. 1257–1262.

[16] A. Schindler, "Vehicle Self-localization with High-Precision Digital Maps," *Proceedings of IEEE Intelligent Vehicles Symposium* (2013) pp. 141–146.

[17] K. Jo, Y. Jo, J. Suhr, H. Jung and M. Sunwoo, "Precise localization of an autonomous car based on probabilistic noise models of road surface marker features using multiple cameras," *IEEE Trans. Intell. Transp. Syst.* **16**(6), 3377–3392 (2015).

[18] W. Lu, Y. Zhou, G. Wan, S. Hou and S. Song, "L3-Net: Towards Learning Based Lidar Localization for Autonomous Driving," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019) pp. 6389–6398.

[19] H. Yin, L. Tang, X. Ding, Y. Wang and R. Xiong, "LocNet: Global Localization in 3D Point Clouds for Mobile Vehicles," *Proceedings of IEEE Intelligent Vehicles Symposium* (2018) pp. 728–733.

[20] H. Chen, C. Liu and Q. Chen, "Self-localization in highly dynamic environments based on dual-channel unscented particle filter," *Robotica*, 1–14 (2020). doi: 10.1017/S0263574720001046.

[21] Y. Sofia and K. Muhammad Bilal, "Information fusion of GPS, INS and odometer sensors for improving localization accuracy of mobile robots in indoor and outdoor applications," *Robotica* **39**(2), 250–276 (2021).

[22] J. Levinson and S. Thrun, "Robust Vehicle Localization in Urban Environments Using Probabilistic Maps," *Proceedings of IEEE Robotics and Automation* (2010) pp. 4372–4378.

[23] R. Wolcott and R. Eustice, "Visual Localization Within Lidar Maps for Automated Urban Driving," *Proceedings of IEEE/RSJ Intelligent Robots and Systems* (2014) pp. 176–183.

[24] I. Baldwin and P. Newman, "Laser-Only Road-Vehicle Localization with Dual 2D Push-Broom LIDARS and 3D Priors," *Proceedings of IEEE/RSJ Intelligent Robots and Systems* (2012) pp. 2490–2497.

[25] K. Yoneda, H. Tehrani, T. Ogawa, N. Hukuyama and S. Mita, "Lidar Scan Feature for Localization with Highly Precise 3-D Map," *Proceedings of IEEE Intelligent Vehicles Symposium* (2014) pp. 1345–1350.

[26] K. Yoneda, C. Yang, S. Mita, T. Okuya and K. Muto, "Urban Road Localization by Using Multiple Layer Map Matching and Line Segment Matching," *Proceedings of IEEE Intelligent Vehicles Symposium* (2015) pp. 525–530.

[27] S. Thrun, "Probabilistic Robotics," *Commun. ACM* **45**(3), 52–57 (2006).

[28] P. Pfaff, C. Plagemann and W. Burgard, "Gaussian Mixture Models for Probabilistic Localization," *Proceedings of IEEE Robotics and Automation* (2008) pp. 467–472.

[29] J. Oberlander, A. Roennau and R. Dillmann, "Hierarchical SLAM Using Spectral Submap Matching with Opportunities for Long-Term Operation," *Proceedings of the 16th Advanced Robotics* (2013) pp. 1–7.

[30] K. Wurm, C. Stachniss and G. Grisetti, "Bridging the gap between feature- and grid-based SLAM," *Rob. Auton. Syst.* **58**(2), 140–148 (2010).
[31] C. Zinoune, P. Bonnifait and J. Ibañez-Guzmán, "Sequential FDIA for autonomous integrity monitoring of navigation maps on board vehicles," *IEEE Trans. Intell. Transp. Syst.* **17**(1), 143–155 (2015).
[32] A. Fujii, M. Tanaka, H. Yabushita T. Mori and T. Odashima, "Detection of Localization Failure Using Logistic Regression," *Proceedings of IEEE/RSJ Intelligent Robots and Systems* (2015) pp. 4313–4318.
[33] S. Kaneko, T. Kondo and A. Miyamoto, "Robust matching of 3D contours using iterative closest point algorithm improved by M-estimation," *Pattern Recogn.* **36**(9), 2041–2047 (2003).
[34] O. Pink, "Visual Map Matching and Localization Using a Global Feature Map," *Proceedings of Computer Vision and Pattern Recognition Workshops* (2008) pp. 1–7.
[35] J. Oberlander, A. Roennau and R. Dillmann, "Hierarchical SLAM Using Spectral Submap Matching with Opportunities for Long-Term Operation," *Proceedings of the 16th Advanced Robotics* (2013) pp. 1–7.
[36] J. Rohde, B. Völz, H. Mielenz and J. Zöllner, "Precise Vehicle Localization in Dense Urban Environments," *Proceedings of the 19th IEEE Intelligent Transportation Systems* (2016) pp. 853–858.
[37] "Velodyne HDL 32-E LiDAR," http://www.velodynelidar.com/lidar/hdlproducts/hdl32e.aspx.
[38] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang and H. Li "PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020) pp. 10529–10538.
[39] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu and M. Bennamoun "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.* (2020). doi: 10.1109/TPAMI.2020.3005434.
[40] Z. Yang, Y. Sun, S. Liu and J. Jia "3DSSD: Point-based 3D Single Stage Object Detector," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020) pp. 10529–10538.
[41] Y. Zhang, J. Wang, X. Wang, C. Li and L. Wang, "A Real-Time Curb Detection and Tracking Method for UGVs by Using a 3D-LIDAR Sensor," *Proceedings of IEEE Conference on Control Applications* (2015) pp. 1020–1025.
[42] P. Besl and N. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(2), 239–256 (1992).
[43] K. Arun, T. Huang and S. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.* **9**(5), 698–700 (1987).
[44] Y. Zhang, L. Wang, J. Wang and J. Dolan "Real-Time Localization Method for Autonomous Vehicle Using 3D-LIDAR," *International Symposium on Dynamics of Vehicles on Roads and Tracks* (2017) pp. 614–619.

Cite this article: Y. Zhang, L. Wang, X. Jiang, Y. Zeng and Y. Dai (2022). "An efficient LiDAR-based localization method for self-driving cars in dynamic environments", Robotica **40**, 38–55. https://doi.org/10.1017/S0263574721000369