Road-Segmentation-Based Curb Detection Method for Self-Driving via a 3D-LiDAR Sensor

Yihuan Zhang, Jun Wang, *Senior Member, IEEE*, Xiaonian Wang, and John M. Dolan, *Senior Member, IEEE*

***Abstract*—The effective detection of curbs is fundamental and crucial for the navigation of a self-driving car. This paper presents a real-time curb detection method that automatically segments the road and detects its curbs using a 3D-LiDAR sensor. The point cloud data of the sensor are first processed to distinguish on-road and off-road areas. A sliding-beam method is then proposed to segment the road by using the off-road data. A curb-detection method is finally applied to obtain the position of curbs for each road segments. The proposed method is tested on the data sets acquired from the self-driving car of laboratory of VeCaN at Tongji University. Off-line experiments demonstrate the accuracy and robustness of the proposed method, i.e., the average recall, precision and their harmonic mean are all over 80%. Online experiments demonstrate the real-time capability for autonomous driving as the average processing time for each frame is only around 12 ms.**

***Index Terms*—Self-driving, 3D-LiDAR sensor, sliding-beam model, road segmentation, curb detection.**

I. Introduction

Autonomous driving technology is growing rapidly to meet the needs of road safety and transportation efficiency. Self-driving cars can be used in many applications where it may be inconvenient, dangerous, or impossible to have a human driver on site. Environment perception is essential for vehicle autonomy, and road boundary detection is a fundamental problem.

In most urban driving scenarios, the road boundary is defined by the position of curbs on both sides. Since curbs are an essential feature distinguishing driving corridors and restricted areas, they are significant for the safety of self-driving. Curbs usually are on both sides of the road and are continuous along the road. However, they are segmented at intersections, which makes the curb detection complicated. In order to obtain curb information on various kinds of roads, it is necessary to develop a method capable of detecting curbs based on intersection recognition.

A. Related Work on Curb Detection

The problem of curb detection has been a popular research topic for decades. A variety of methods have been proposed to detect curbs on urban roads by using their spatial features.

Camera-only methods have been extensively studied as cameras are relatively inexpensive and intuitive for humans. In [1], a digital elevation map was established by a stereo vision system. Based on the height difference between the curbs and the road surface, the Hough transformation method was applied to extract the curb segments. In [2], a digital elevation map was used and a conditional random field method was proposed to propagate the curb position. A polynomial curve-fitting method was then applied to represent the curbs. Recently, a Naive Bayes framework was proposed in [3] to detect the road boundary using stereo vision and the method of support vector regression was used to fit curb curves. However, it is hard for camera-based methods to accurately detect the position of curbs due to the slight difference in color between the road surfaces and curbs. Moreover, these methods are usually sensitive to lighting and weather conditions.

Unlike passive sensors such as cameras, the millimeter wave radars (MMWRs) are not susceptible to ambient light. The MMWRs even have a longer detection range of over 100 meters. In [4], an MMWR was used to obtain depth information and a model-matching method was proposed to obtain the road curve information. In [5], the image of a camera was fused with the information of an MMWR by using a Bayesian method, which improved the accuracy of road curb positions. However, the accurate information of road curbs was hard to obtain due to the low resolution and narrow field-of-view of the MMWRs.

Laser scanners usually have a much higher resolution of curb positions and therefore are widely used in environmental perception. In [6], the point cloud data of a 2D-LiDAR sensor were projected onto a plane vertical to the ground in order to extract the elevation feature of curbs. In [7], a 2D-LiDAR sensor was used to detect and track the road boundary and an extended Kalman filter (EKF) was used to enhance its efficiency and accuracy. The road boundary detection method using 2D-LiDAR sensors was time-efficient, but the true positive rate of boundary detection remained low due to the limited sensing data. In comparison with the 2D-LiDAR sensors,

Manuscript received October 12, 2016; revised May 3, 2017, October 24, 2017, and December 29, 2017; accepted January 1, 2018. This work was supported by the National Natural Science Foundation of China under Grant 61473209 and Grant 61773291. This paper was presented in part at the 18th IEEE International Conference on Intelligent Transportation Systems in 2015 and the IEEE International Conference on Control Applications in 2015. The Associate Editor for this paper was J. Li. *(Corresponding author: Jun Wang.)*

Y. Zhang, J. Wang, and X. Wang are with the Department of Control Science and Engineering, Tongji University, Shanghai 201804, China (e-mail: junwang@tongji.edu.cn).

J. M. Dolan is with the School of Computer Science, Robotic Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2018.2789462

3D-LiDAR sensors are able to provide a large amount of sensing data with a 360° coverage. Thus, the 3D-LiDAR sensors have been increasingly employed to percept the environment. In the previous DARPA Challenge, many teams were equipped with the 3D-LiDARs for environmental perception [8], [9]. In [10], a 3D-LiDAR sensor was used to obtain a large amount of data about surrounding environment. Off-line experiments showed that the proposed method was accurate and robust in most scenarios. However, it is essential to design an *online* method for autonomous driving. In our previous work [11], a real-time method was proposed and tested for curb detection and tracking. The experimental results showed that the method was time-efficient and accurate, but could not deal with intersections. More recent work on curb detection via a 3D-LiDAR sensor also illustrated the advantages of the sensor and their robust performances [12], [13].

The problem of curb detection has also been studied in the remote sensing domain. The methods of road boundary detection were proposed using 3D point-cloud data of high-resolution [14]–[16] or aerial images [17]. In addition, the mobile laser scanning (MLS) system has recently been generally used in the areas of transportation, navigation and autonomous driving [18]. The dense point cloud of the MLS can provide high resolution information of environment, which makes it perfect for building a digital map. Many researchers focused on road information extraction via the MLS systems. In [19]–[21], road markings were extracted and classified automatically by using the point cloud of MLSs. High completeness and correctness were achieved in these papers. Road boundary extraction or curb detection were also studied using the MLS systems. In [22], a moving window operator was used to filter out non-ground points and a feature-based search method was then applied to extract the curb points. The experiments were carried out via a MLS system, and provided promising results of curb extraction. In their recent work [18], a supervoxel generation and road boundary extraction method was proposed. The detection results were accurate with whole context information. However, the sensing range was limited for self-driving, and the sensing data in one frame may not contain enough features. More importantly, the computation cost was too high to provide real-time information processing.

The aforementioned methods all assumed that curbs were continuous and can be fitted as a curve. However, the curbs at intersections are segmented and discontinuous *in sight*, which cannot be represented as a curve. Instead, the road should first be segmented, and then the continuity can be assumed for each segment.

B. Related Work on Intersection Recognition

Many researchers have been working on the problem of intersection recognition. In [23], a monocular vision system was applied to obtain the image, and a learning method was proposed to recognize the shape of intersections. In [24], a threshold-based method was proposed to detect the shape of intersections by using a monocular vision system. The method was time-efficient but relatively less accurate. In [25], a feature-based method was proposed to detect the edges of roads with intersections by using a 2D-LiDAR sensor. In [26], a single-beam model was proposed to classify the types of intersections by using a 3D-LiDAR sensor. In [26] and [27], some trial-and-error rules were used to recognize the types of intersections. However, it is difficult to identify the parameters of the rules for various road scenarios. In [28] and [29], machine learning methods were employed to classify the road shape by using beam models. In [30], multiple LiDAR sensors were fused to identify geometric features of roads, and a fast convolution method was proposed to evaluate the fitness of the given shapes of roads. In [31], the data of 2D-LiDAR sensors and 3D-LiDAR sensors were fused to establish the visibility map and the road was divided into several zones. A zone-checking method was proposed to recognize the shapes of intersections and to extract the boundaries of each zone. In our previous work [32], a double-layer beam model was built to recognize the type of the intersections, based on which the road boundary was detected. The drawback of our previous method is that the approximate position of intersections should be available from maps before recognizing intersection shapes.

In this paper, a 3D-LiDAR sensor is utilized to sense the surrounding environment. Sensor calibration is carried out and a flat filtering method is applied to distinguish the on-road and off-road point clouds. Based on off-road data, a sliding-beam method is used to segment the road and recognize road shapes, and the curbs are then extracted. The framework of this paper is shown in Fig. 1. This paper makes the following contributions:

- The proposed method is able to recognize the road shape and extract the curbs in each segment, allowing it to handle the curb detection problem at intersections.
- We propose an improved sliding-beam method for road segmentation which is robust to various road shapes (e.g., +-shape, T-shape, Y-shape).
- We propose a search-based method to extract the curbs in each road segment based on the spatial features of the point cloud data.
- The proposed method ensures real-time performance for self-driving.

The remainder of this paper is organized as follows. Section II describes the sensor calibration method and the plain-based filtering method. Section III details the method of road segmentation and curb detection. Section IV evaluates the proposed method through comprehensive experiments. Section V summarizes the contribution of the paper and maps out the directions for future research.

II. Data Pre-Processing

This section proposes the sensor calibration method and a plane-based filtering method to extract the on-road and off-road data.

A. Sensor Calibration

The HDL-32E LiDAR sensor by *Velodyne* features up to 32 lasers vertically aligned from +10° to −30°, and its rotating head delivers a 360° horizontal field of view. It generates a point cloud of 700,000 points per second with a range of 70 m and typical accuracy of 2 cm [33].

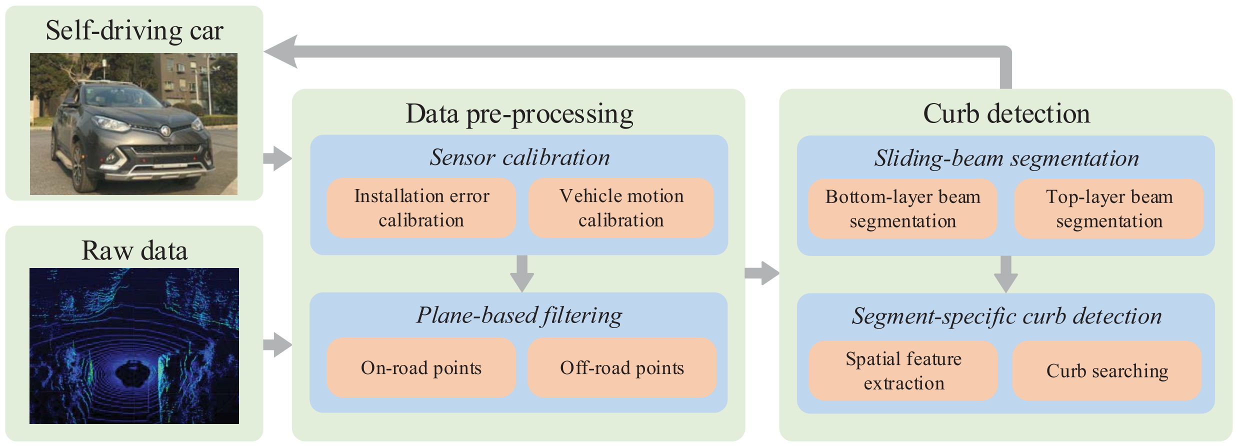

Self-driving car
Raw data
Data pre-processing
Sensor calibration
Installation error calibration
Vehicle motion calibration
Plane-based filtering
On-road points
Off-road points
Curb detection
Sliding-beam segmentation
Bottom-layer beam segmentation
Top-layer beam segmentation
Segment-specific curb detection
Spatial feature extraction
Curb searching

Fig. 1. The flowchart of the proposed method.

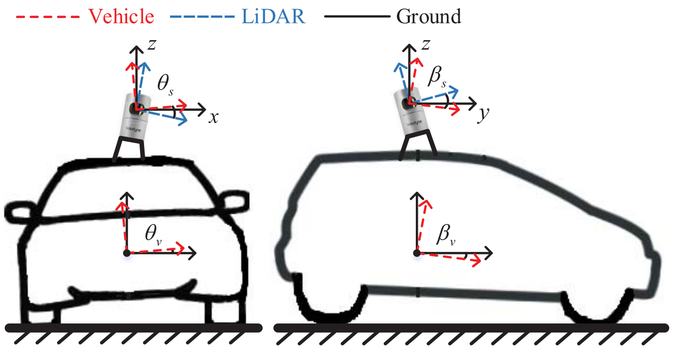


Fig. 2. Description of the coordinate systems. Left: front view. Right: side view.

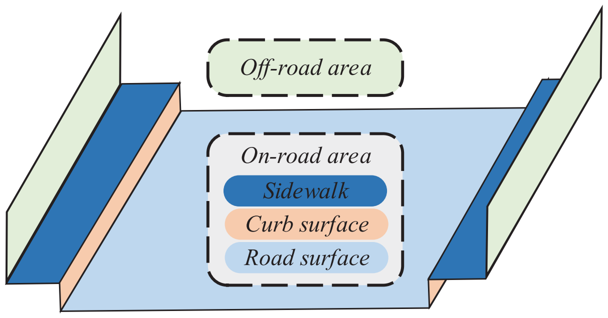


Fig. 3. Road geometry description.

TABLE I
RELATIONSHIP BETWEEN l AND θ_F [33]

l	$\theta_f(°)$	l	$\theta_f(°)$	l	$\theta_f(°)$	l	$\theta_f(°)$
1	−30.67	9	−25.33	17	−20.00	25	−14.67
2	−9.33	10	−4.00	18	1.33	26	6.67
3	−29.33	11	−24.00	19	−18.67	27	−13.33
4	−8.00	12	−2.67	20	2.67	28	8.00
5	−28.00	13	−22.67	21	−17.33	29	−12.00
6	−6.66	14	−1.33	22	4.00	30	9.33
7	−26.66	15	−21.33	23	−16.00	31	−10.67
8	−5.33	16	0.00	24	5.33	32	10.67

In this paper, the 3D-LiDAR sensor is mounted on the top of a self-driving car of VeCaN Laboratory at Tongji University and the raw data of the sensor are in a 3D polar coordinate. The coordinates can be converted into the sensor coordinate as shown in Fig. 2. Each point contains a Cartesian coordinate (x, y, z) and the corresponding laser line $l \in \{1, 2, 3, \ldots, 32\}$. The relationship between the laser line l and the vertical pointing angle θ_f is shown in Table I.

The method of sensor calibration is detailed in [34]. First, the points in the sensor coordinate are transformed into the vehicle coordinate based on the installation deviation angles (θ_s and β_s). In the experimental car, the height of the sensor is 1.5 m, $\theta_s = 0.4°$ and $\beta_s = -2.5°$ with respect to the vehicle coordinate. Then, a differential global positioning system (DGPS) and an inertial navigation system (INS) are utilized to obtain the information of vehicle motion. The typical positioning accuracy is 5 cm and the accuracy of the acquired point cloud is about 2 cm. The average number of points in each frame is about 60,000. Based on the roll angle θ_v and the pitch angle β_v of the host vehicle, the points are transformed into the ground coordinate and the origin is in the center of the sensor. Let $\mathbf{P}$ denote the calibrated dataset and each point be $\mathbf{p}_{i,l} = [x_{i,l}, y_{i,l}, z_{i,l}]$ where l is the number of the corresponding laser line.

B. Plane-Based Filtering Method

Fig. 3 shows the road geometry. The on-road area consists of sidewalks, curb surfaces and road surfaces. The off-road area usually consists of trees, buildings and other objects. After sensor calibration, relatively accurate data are obtained. Here we propose a plane-based filtering method to distinguish on-road and off-road areas.

The plane is defined by a linear equation and a simple least-square method is then applied to plane fitting. First, all points in $\mathbf{P}$ with negative vertical pointing angle are selected. Then a random sample-consensus method is implemented to estimate the parameters of the plane [35]. The extraction of on-road areas does not require high precision because the curb detection method will be further applied. Thus, based on the average elevation of curbs, we set the threshold to 0.2 m, which

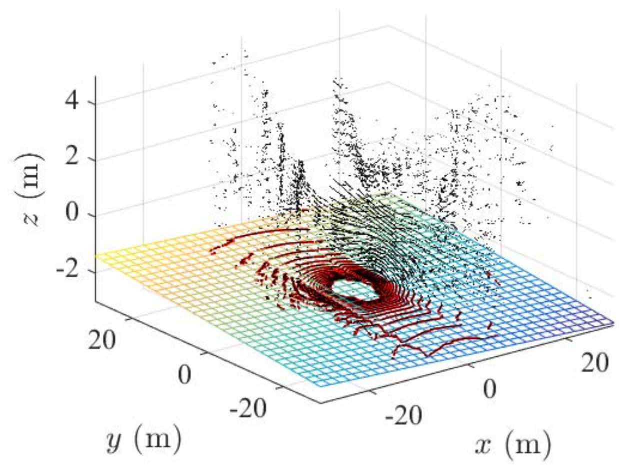


Fig. 4. An example of the flat filtering method.

determines whether a point is on-road or off-road. Note that the plane cannot accurately extract the road surface or sidewalk, neither of which is necessary for the further process. In [18], the road surface was represented by a set of facets which may accurately extract the road surface area. However, the method was time-consuming because it traverses all the points to find their nearest k-neighbors in order to calculate the smoothness and normal vector for each small plane. It then traversed all the points which meet the criteria of the road surface feature and generate several large facets to represent the road surface. In our method, the "rough" on-road data is processed in the next section to extract the curbs for which the single-plane fitting method is able to provide enough data and reduce most of the outliers simultaneously.

One sample of the plane-based filtering method is shown in Fig. 4, where on-road points shown in red are used to extract curbs and off-road points are used to segment the road and recognize its shape. Let $\mathbf{P}_{\text{on}}$ denote the on-road data and $\mathbf{P}_{\text{off}}$ the off-road data.

III. Curb Detection

This section proposes the curb detection method. A sliding-beam method is presented, and a search-based method is applied to detect the curbs in each frame from the on-road point cloud.

A. Sliding-Beam Segmentation

It is necessary for self-driving cars to recognize the road shape, especially the shape of intersections and to find the drivable corridors. The beam model approximates the physical model of range finders and has been widely used in robotics [36]. It is a sequence of beams from a launching point where the range finders are originated and evenly spaced with a given angle resolution θ_{r}. In on our previous work [32], we set a beam model at a fixed distance to detect the intersections. However, it is hard to determine a fixed distance which is robust to various kinds of intersections. Therefore we now propose a sliding-beam model which does not rely on a determined distance to launch the beam model. The beam

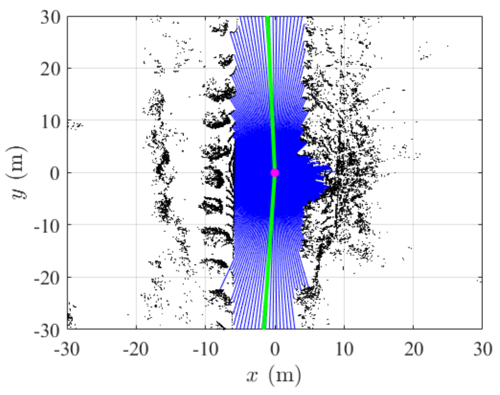


Fig. 5. An example of the bottom-layer beam model.

model contains two parts, i.e. a bottom-layer model and a top-layer one.

1) Bottom-Layer Beam Model: The bottom-layer beam model determines the beam angles for the road segmentation. The angles represent the road direction with respect to the current location of the vehicle. By setting the vehicle position as the launching point, the beam angles are determined by a peak-finding method.

Based on the feature of the LiDAR sensor and the installation position, the data of point cloud are sparse beyond 30 meters. Thus the region of interest is defined by $\{(x, y)| -30 \leq x \leq 30, -30 \leq y \leq 30\}$. In order to determine an accurate beam angle, the angle resolution θ_{r} is set to 1° in this paper. Furthermore, the beam length in the classic model is calculated by the launching point and the nearest point along the beam angle. We expand the beam into a beam zone $\mathbf{Z}_k$ which defined by

$$\mathbf{Z}_k = \left\{(x, y)\left|\frac{(k-1)\cdot\pi}{360} < \arctan\left(\frac{y-y_b}{x-x_b}\right) \leq \frac{k\cdot\pi}{360}\right.\right\} \quad (1)$$

where (x_b, y_b) is the launching point and $k \in \{1, 2, \ldots, 360\}$. The beam length in a zone is defined by the shortest distance of the points in the zone to the launching point, i.e.,

$$d_k = \min_i \sqrt{(x_i - x_b)^2 + (y_i - y_b)^2} \quad (2)$$

where $(x_i, y_i) \in (\mathbf{Z}_k \cap \mathbf{P}_{\text{off}})$. Then a rule-based peak-finding method in [37] is applied to find the segmenting beam angles on road. The beam length in each zone is normalized by

$$\bar{d}_k = \frac{d_k}{\max_{x,y\in\mathbf{Z}_k} \sqrt{(x - x_b)^2 + (y - y_b)^2}} \quad (3)$$

Note that, $\bar{d}_k = 1$ means that there are no off-road points in the kth zone and there might be a branch of the road. The following metrics are defined to extract segmenting angles on the road:

- The threshold of the peak-beam length is set to 1. If the beam length reaches the threshold, i.e. $\bar{d}_k = 1$, no off-road obstacles are in the kth zone.
- The distance of two adjacent non-peak beams is defined as the distance from the endpoint of the shorter beam to the longer beam. The distance threshold is set to D_{b}.

One example of the bottom-layer beam model is illustrated in Fig. 5. The off-road points are shown in black, the purple point is the launching point and the blue lines are in the middle of each beam zone. The green line represents the angle of road segment. In this case, we assume that the road is segmented into two parts, which can also be represented as no intersection at the current position of the host vehicle. By applying the bottom-layer beam model, we are able to segment the road according to the current position of the host vehicle, but we cannot determine whether there is an intersection ahead. Thus, the top-layer beam model is applied next to find the intersections ahead of the host vehicle.

2) Top-Layer Beam Model: The top-layer beam model is built to segment the road ahead of the host vehicle, i.e. to recognize the road shape ahead and especially intersections. A sliding-beam method is then applied. By assuming the initial angle is θ_b, the set of launching points of the top-layer beam model is given by

$$\begin{cases} x_{t,i} = d_b \cdot i \cdot \cos\theta_b \\ y_{t,i} = d_b \cdot i \cdot \sin\theta_b, \end{cases} \quad i \in \{1, 2, \ldots, n_t\}$$

where $(x_{t,i}, y_{t,i})$ is the launching point of the top-layer beam model, d_b is the distance interval between two launching points, and n_t is the number of beam models. Note that $x_{t,i}$ and $y_{t,i}$ should be in the region of interest. Another important parameter is the beam angle resolution θ_r, which also affects the computation time and the resolution of road segmentation.

After launching n_t beam models, the peak-finding method is applied again to obtain the segmenting beam angles in each beam model. The output of each beam model is denoted by $\mathbf{B}_i = (N_i, \theta_{t,i}, x_{t,i}, y_{t,i})$, where $\theta_{t,i}$ are the segmenting beam angles in the ith beam model, N_i is the number of $\theta_{t,i}$, and $(x_{t,i}, y_{t,i})$ is the launching point. Then a voting method given in Algorithm 1 is employed to determine the final segmenting beam angles and the launching point.

An example of the top-layer beam model is shown in Fig. 6, where $d_b = 1$, $D_b = 6$ and $n_t = 28$. The maximal N_i is 4 and the corresponding indices are from 7 to 15 which is larger than 6, so the final road segmentation can be decided from the beam model with $i = 11$.

Algorithm 1 The Voting Method

```
Input: A set B of top-layer beam models $\mathbf{B}_i$ where $i \in n_v$
Output: Final road segmentation $\{x_o, y_o, \theta_o\}$
Sort $\mathbf{B}_i$ by $N_i$ and count the number of same elements in $N_i$;
  for all different elements in $N_i$, starting from the maximum element in $N_i$ do
    if number of elements is larger than $D_b/d_b$ then
      Calculate the mean index value in B with the current element, and find the nearest index $k$ to the mean value;
      $x_o = x_{t,k}$, $y_o = y_{t,k}$, $\theta_o = \theta_{t,k}$ and break loop;
    end
end
```

B. Segment-Specific Curb Detection

Based on the road segmentation, the curb detection method is proposed in the following. First, the spatial features of curbs are defined and extracted. Then the feature thresholds

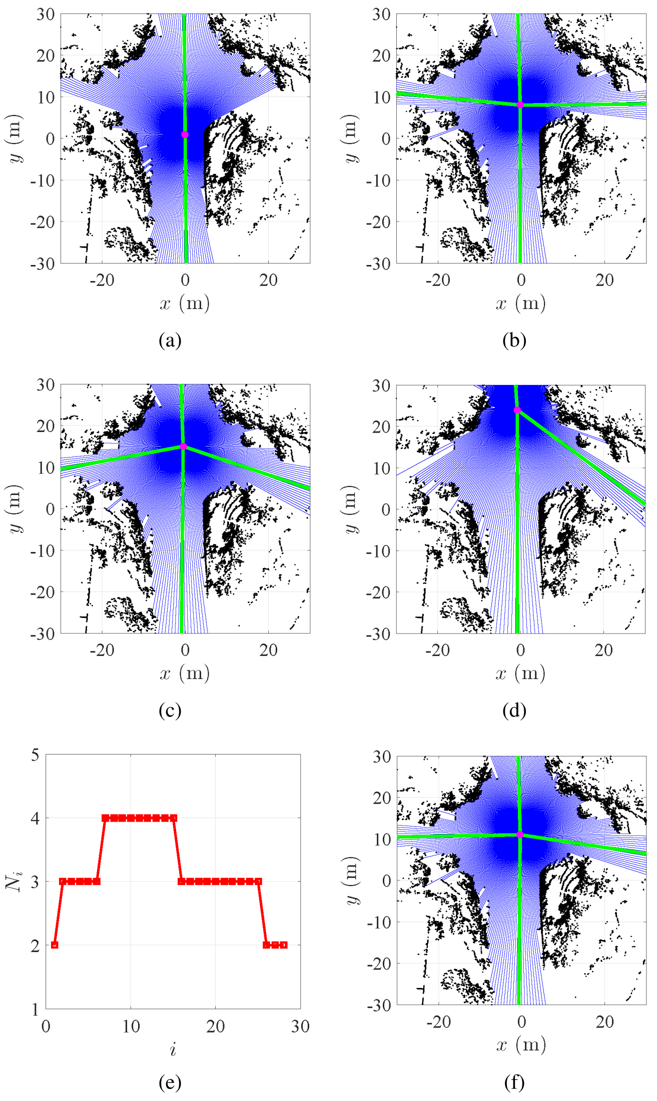


Fig. 6. An example of the top-layer beam model. The sliding beam transfered from the current position of the vehicle to the edge of the region to detect the segments on road. (a) $i = 1$. (b) $i = 8$. (c) $i = 15$. (d) $i = 24$. (e) N_i in sliding beam models. (f) $i = 11$.

are determined and a curb searching method is used to detect the curbs in each segment.

1) Spatial Feature Extraction: Curbs have many spatial features different from road surfaces. The curb height is generally 10 cm to 15 cm, and the elevation changes sharply. In addition, the road surface points are smooth and continuous, and the curb usually appears on the sides of the road. As shown in Fig. 7, H_c represents the curb height, which is set to 0.15 m in this paper. Based on these features, three indices are defined to extract the curbs in each frame.

- $\delta_{xy,l}$ represents the distance between two adjacent points of a road surface in the lth laser line. It sets the horizontal threshold for determining whether the point is on the road surface. It is defined by

$$\delta_{xy,l} = H_s \cdot \cot\theta_{f,l} \cdot \frac{\pi\theta_a}{180} \tag{4}$$

where H_s is the sensor height, $\theta_{f,l}$ is the vertical pointing angle of the lth laser line and θ_a is the angular resolution

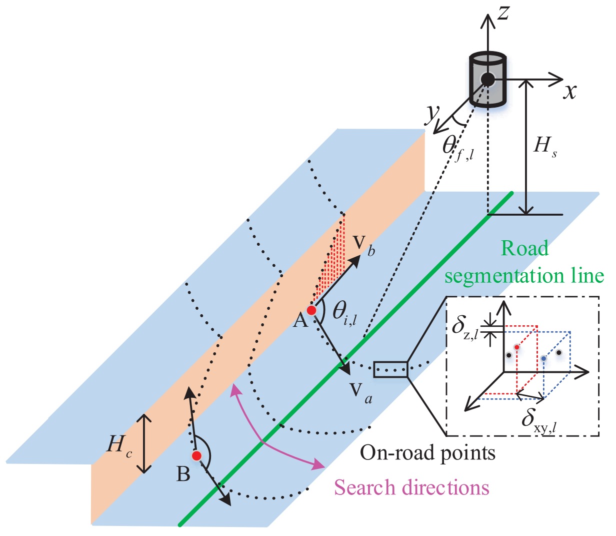


Fig. 7. Curb scenario description.

of the sensor. The parameter θ_a is usually in the range of $[0.1°, 0.4°]$ and is set to 0.4° in this paper.

- $\delta_{z,l}$ is the vertical distance between two adjacent road surface points in the same laser line, and is used to set the vertical threshold to determine the road surface points. It is defined as

$$\delta_{z,l} = \delta_{xy,l} \cdot \sin \theta_{f,l} \tag{5}$$

We assume that the points on the curb surface are uniformly spaced and $\delta_{z,l}$ is the average distance between two points projected to the curb surface.

- $\theta_{i,l}$ is the angle between two vectors originating from the same point $\mathbf{p}_{i,l}$, where $\mathbf{p}_{i,l}$ represents the ith point in lth laser line. The angle $\theta_{i,l}$ is defined as

$$\begin{aligned}
\theta_{i,l} &= \cos^{-1} \frac{\mathbf{v}_a \cdot \mathbf{v}_b}{|\mathbf{v}_a| \cdot |\mathbf{v}_b|} \\
\mathbf{v}_a &= \frac{1}{n_v}\left[\sum_{k=1}^{n_v}(x_{i-k,l} - x_{i,l}), \sum_{k=1}^{n_v}(y_{i-k,l} - y_{i,l})\right] \\
\mathbf{v}_b &= \frac{1}{n_v}\left[\sum_{k=1}^{n_v}(x_{i+k,l} - x_{i,l}), \sum_{k=1}^{n_v}(y_{i+k,l} - y_{i,l})\right] \\
n_v &= \left\lceil \frac{H_c}{\sin \theta_{f,l} \cdot \delta_{xy,l}} \right\rceil
\end{aligned} \tag{6}$$

where n_v is the theoretical number of laser points on a curb surface, $\mathbf{v}_a$ and $\mathbf{v}_b$ are the vectors along the laser points as shown in Fig. 7. When the point $\mathbf{p}_{i,l}$ is on the road surface and near the curb, $\mathbf{v}_a$ represents an approximate direction on the road surface and $\mathbf{v}_b$ consists of points on the curb surface. In that case, the angle between two vectors is less than the angle between two vectors on a road surface. An example is shown in Fig. 7, where there is an obvious difference between the angles of Point A and the Point B. Based on the road segmentation, the initial searching point in each segment is on road surfaces. Then the curb searching method can be used to extract all the curbs.

TABLE II
KEY PARAMETERS IN ROAD SEGMENTATION

Parameters	Physical meanings	Considered values	Units
D_b	Distance threshold	4, 6, 8	m
d_b	Sliding beam model interval	0.5, 1, 2, 3, 4	m
θ_r	Beam angle resolution	1, 2, 3, 4	°

2) Curb Searching: The points in each segment are sorted anticlockwise for the curb searching process. The curb should be searched in both directions because a given road segment usually have two curbs. The following steps determine the features of curbs. *Step* 1

1) Horizontal continuity: the horizontal distance between two adjacent points is calculated. If the distance is larger than $\delta_{xy,l}$, the point is possibly a curb point.
2) Vertical continuity: the vertical distance between two adjacent points is calculated. If the distance is larger than $\delta_{z,l}$, the point is possibly a curb point.
3) Angle threshold: when a possible curb point occurs, $\theta_{i,l}$ is then calculated. The angles of points on the road surface are nearly 180° like Point B in Fig. 7 and the angles of points on curbs are on average 100° like Point A. Therefore, the angle threshold is set to 150°. It is a relatively loose threshold as there are further filtering steps.
4) Elevation threshold: when $\theta_{i,l}$ of a possible curb point is less than the angle threshold, the elevation distribution is calculated in the points of two vectors. If elevation of points in $\mathbf{v}_b$ is increased and the maximum elevation between $\mathbf{p}_{i+n_v,l}$ and $\mathbf{p}_{i,l}$ is greater than H_c, the point $\mathbf{p}_{i,l}$ is counted as a curb point.

Once all laser line points are checked, the whole set of curb points is obtained. There may be a few false positive points, but the precision and recall are relatively high as demonstrated in the following experiments.

IV. Experimental Results

To evaluate the proposed method, extensive off-line experiments have been carried out. In this section, five testing scenarios are selected to evaluate the proposed method. In addition, an online road test is conducted to evaluate the accuracy of the detection results and the capability of real-time computation for self-driving.

A. Road Segmentation Experiments

As shown in Fig. 8, five different scenarios are extracted, including the straight road, curve road, T-shape intersection, +-shape intersection and Y-shape intersection. The first three scenarios consist of 50 frames each. The +-shape intersection consists of 30 frames and the Y-shape intersection consists of 20 frames. Three key parameters listed in Table II must be determined for the road segmentation.

A combination testing method is used to search for the appropriate parameters. To reduce the complexity, these parameters are confined to a certain range empirically. For each scenario, 60 experiments are conducted to analyze the parameters. The testing results are listed in Table III and each

TABLE III
TSR IN THE PARAMETER COMBINATION EXPERIMENT

Mean ± Std.		$d_b = 0.5$	$d_b = 1$	$d_b = 2$	$d_b = 3$	$d_b = 4$
$D_b = 4$	$\theta_r = 1$	0.8040 ± 0.2233	0.8160 ± 0.2326	0.8840 ± 0.1367	0.9160 ± 0.1315	0.8973 ± 0.1694
	$\theta_r = 2$	0.8840 ± 0.1389	0.9520 ± 0.0672	0.9453 ± 0.0628	0.9387 ± 0.0717	0.8853 ± 0.1099
	$\theta_t = 3$	0.9020 ± 0.1588	0.9540 ± 0.0780	0.9840 ± 0.0358	0.9533 ± 0.0650	0.9427 ± 0.0802
	$\theta_r = 4$	0.8800 ± 0.2168	0.9300 ± 0.1304	0.9580 ± 0.0694	0.9340 ± 0.0760	0.9033 ± 0.1121
$D_b = 6$	$\theta_r = 1$	0.9620 ± 0.0634	0.9800 ± 0.0447	0.9800 ± 0.0447	0.9800 ± 0.0447	0.9633 ± 0.0650
	$\theta_r = 2$	0.9900 ± 0.0224	0.9900 ± 0.0224	0.9867 ± 0.0298	0.9693 ± 0.0580	0.9480 ± 0.0867
	$\theta_r = 3$	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	**1.0000 ± 0.0000**	0.9853 ± 0.0202	0.9680 ± 0.0460
	$\theta_r = 4$	0.9900 ± 0.0224	0.9900 ± 0.0224	0.9833 ± 0.0236	0.9687 ± 0.0301	0.9513 ± 0.0549
$D_b = 8$	$\theta_r = 1$	0.9700 ± 0.0671	0.9800 ± 0.0447	0.9800 ± 0.0447	0.9833 ± 0.0236	0.9533 ± 0.0650
	$\theta_r = 2$	0.9833 ± 0.0236	0.9700 ± 0.0447	0.9533 ± 0.0650	0.9393 ± 0.0987	0.9040 ± 0.1228
	$\theta_r = 3$	0.9767 ± 0.0325	0.9767 ± 0.0325	0.9500 ± 0.0866	0.9087 ± 0.1556	0.8980 ± 0.1689
	$\theta_r = 4$	0.9800 ± 0.0447	0.9600 ± 0.0548	0.9333 ± 0.1027	0.8987 ± 0.1539	0.8827 ± 0.1996

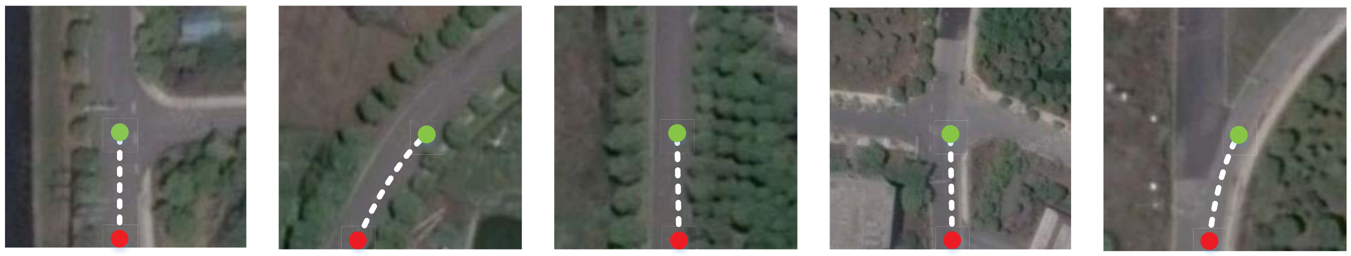


Fig. 8. Bird view of five testing scenarios, red dot is the start point and green dot is the end point. Trajectory is illustrated in white. (a) T-shape intersection. (b) Curve road. (c) Straight road. (d) +-shape intersection. (e) Y-shape intersection.

value is the average true detection rate from five scenarios. The true detection rate is defined as the ratio of correctly detected frames to all frames. The true segmentation result is manually labeled in each scenario. For example, in Fig. 8(a), we labeled all 50 frames as a three-segment intersection because the intersection is in the region of interest from the start to the end of the scenario and should be detected. If the road segmentation number is 3 and the direction is correct, the result is a *true segmentation*. If the road segmentation number is 3 but the final road segment angles are not right, the result is counted as a *wrong segmentation*. In addition, if the number is not 3, it is counted as a *false segmentation*. In this parameter determination experiment, we only use the *true segmentation rate* (TSR) as the evaluation metric.

After the experiment, we can analyze the key parameters. In the testing scenarios, the road width is approximately 8 meters (two lanes) and that is the reason we choose 4 m, 6 m and 8 m as the distance thresholds. When D_b is set to 4 m, some gaps such as the gap between two trees are falsely detected as a road branch. In Fig. 8(d), the right branch of the intersection is a one-lane narrow road which is about 6 m wide. For this case, $D_b = 8$ m causes a missed detection of that branch. Different choices of d_b also lead to different true detection rates. For example, when d_b is set to 4 m in Fig. 8(a), the distance between two sliding beam models is too large to find the branch. The different setting of θ_r almost has the same effect as that of d_b, i.e., a large θ_r may cause the beam to miss some branches. Theoretically, the true detection rate will be higher if d_b and θ_r become smaller, because dense beam models will be instantiated to cover the intersection. However, instantiating a dense beam model is time-consuming.

TABLE IV
COMPARISON RESULTS OF ROAD SEGMENTATION (THE DATA ARE IN THE FORMAT OF MEAN ± STANDARD DEVIATION)

	Zhu [26]	Li [27]	Proposed
TSR	0.5887 ± 0.2545	0.7087 ± 0.2320	**1.0 ± 0.0**
FSR	0.4213 ± 0.2552	0.2733 ± 0.2147	**0.0 ± 0.0**
WSR	0.0300 ± 0.0283	0.0180 ± 0.0249	**0.0 ± 0.0**

By balancing the processing time requirement and the true detection rate, we choose $D_b = 6$ m, $d_b = 2$ m and $\theta_r = 3°$ for the method.

In order to demonstrate the robustness of the proposed method, we compare it with two state-of-the-art methods in [26] and [27]. Two more quantitative metrics are introduced to evaluate these methods: the *wrong segmentation rate* (WSR) and the *false segmentation rate* (FSR). Experimental results are shown in Table IV.

In [26], the beam model was set to a constant distance to detect the intersection, and therefore it can only detect an intersection at a specific location. In [27], the launching point of the beam model was determined by a function of vehicle speed and a parameter. It is difficult to tune the parameter for each scenario. The proposed method in this paper is robust to various kinds of road shapes and the appropriate parameters are determined from experiments. In conclusion, the proposed method demonstrates a promising quality of road segmentation and is robust to various shapes of intersections.

B. Curb Detection Experiments

Five scenarios are used to evaluate the proposed curb detection method. We have manually labeled the curbs in each

TABLE V

CURB DETECTION RESULTS

Mean ± Std.		T-shape intersection	Curve road	Straight road	+-shape intersection	Y-shape intersection
PPV	Proposed	**0.8498 ± 0.0697**	**0.8764 ± 0.0390**	**0.8230 ± 0.0616**	**0.8584 ± 0.0637**	**0.8373 ± 0.0645**
	Zhang [11]	0.4421 ± 0.0643	0.4464 ± 0.0322	0.4721 ± 0.0633	0.3900 ± 0.1160	0.3336 ± 0.0587
	Peterson [30]	0.0434 ± 0.0032	0.0750 ± 0.0097	0.0889 ± 0.0089	0.1005 ± 0.0155	0.0776 ± 0.0068
	Kang [7]	0.2394 ± 0.0271	0.3888 ± 0.0431	0.2337 ± 0.0481	0.3638 ± 0.0691	0.2790 ± 0.0381
	Zai [18]	0.4790 ± 0.0428	0.4525 ± 0.1052	0.6312 ± 0.0728	0.4009 ± 0.0361	0.3609 ± 0.0613
	Yang [22]	0.3240 ± 0.0365	0.4130 ± 0.0324	0.3883 ± 0.0478	0.2850 ± 0.0587	0.3382 ± 0.0207
TPR	Proposed	**0.8928 ± 0.0275**	**0.8227 ± 0.0362**	0.7716 ± 0.0550	**0.7834 ± 0.0562**	**0.8734 ± 0.0308**
	Zhang [11]	0.4724 ± 0.0451	0.4895 ± 0.0376	0.5019 ± 0.0471	0.2985 ± 0.1048	0.3845 ± 0.0738
	Peterson [30]	0.7915 ± 0.0605	0.7052 ± 0.0558	**0.9154 ± 0.0434**	0.6938 ± 0.0806	0.7731 ± 0.0479
	Kang [7]	0.2930 ± 0.0366	0.4726 ± 0.0468	0.2587 ± 0.0534	0.4689 ± 0.0661	0.3827 ± 0.0423
	Zai [18]	0.6801 ± 0.0506	0.6387 ± 0.0505	0.6576 ± 0.0629	0.4821 ± 0.0860	0.5941 ± 0.0452
	Yang [22]	0.4883 ± 0.0422	0.6597 ± 0.0598	0.4892 ± 0.0438	0.5443 ± 0.0924	0.5580 ± 0.0403
F_1	Proposed	**0.8698 ± 0.0460**	**0.8483 ± 0.0327**	**0.7957 ± 0.0520**	**0.8179 ± 0.0501**	**0.8539 ± 0.0422**
	Zhang [11]	0.4562 ± 0.0540	0.4668 ± 0.0335	0.4861 ± 0.0543	0.3351 ± 0.1038	0.3567 ± 0.0640
	Peterson [30]	0.0823 ± 0.0061	0.1355 ± 0.0167	0.1619 ± 0.0147	0.1752 ± 0.0253	0.1410 ± 0.0116
	Kang [7]	0.2623 ± 0.0247	0.4264 ± 0.0437	0.2450 ± 0.0488	0.4091 ± 0.0675	0.3223 ± 0.0390
	Zai [18]	0.5617 ± 0.0440	0.4920 ± 0.0385	0.6436 ± 0.0653	0.4638 ± 0.0890	0.4476 ± 0.0572
	Yang [22]	0.3892 ± 0.0380	0.5076 ± 0.0397	0.4325 ± 0.0454	0.3718 ± 0.0655	0.4209 ± 0.0251

frame of the above scenarios. The data of the labeled curbs are available online[1] for researchers to implement their methods. Five different methods together with the proposed method are tested on the labeled data. Three quantitative metrics are introduced for a comprehensive evaluation. Moreover, the distance threshold of true detection is set to 10 cm according to the positioning error and the labeling error. The distance is calculated by the detected curb and its nearest labeled curb.

- *Precision*, or *positive predictive value* (PPV) is the fraction of the curbs detected correctly out of all the curbs detected in one frame, i.e.

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

where TP is the true positive numbers and FP is the false positive numbers (false alarm).

- *Recall*, or *true positive rate* (TPR), is the fraction of the curbs detected correctly out of all the labeled curbs, i.e.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

where FN means false negative (missed detection).

- F_1 *Score* is the harmonic mean of the precision and the recall, i.e.

$$F_1 = 2 \cdot \frac{\mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}}$$

The results of four examples of curb detection are illustrated in Fig. 9. The green squares are the manually labeled curb data and the red dots represent the detected curbs. In each scenario, the road is segmented by the green lines and on-road data in each segment are denoted by colors. Most of the labeled curbs are detected by the proposed method.

In order to demonstrate the effectiveness of the proposed method, the following state-of-the-art methods are also employed for comparison:

- *Zhang [11]:* Our previous sliding-window method
- *Peterson [30]:* The wavelet transformation method
- *Kang [7]:* The hough transformation method

[1]The data of all the labeled curbs can be found on the website of VeCaN Laboratory: http://vecan.tongji.edu.cn/?research

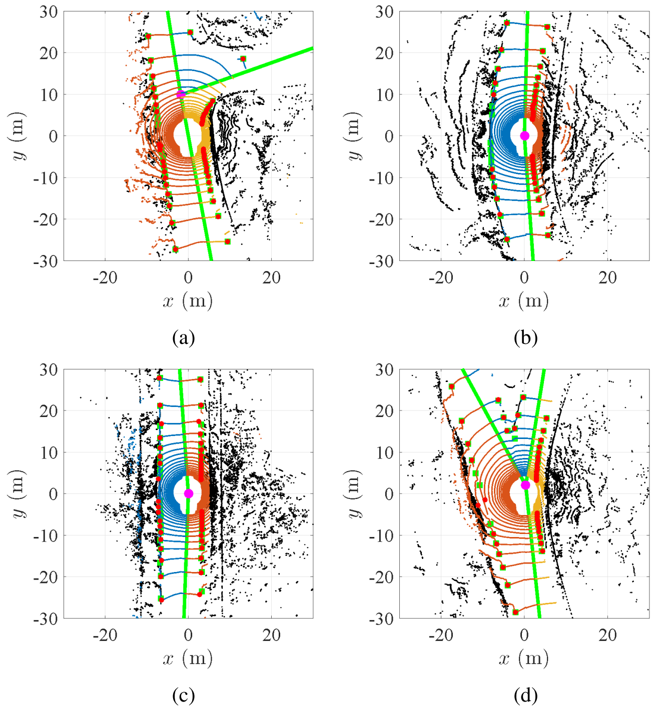


Fig. 9. Examples of the curb detection method. (a) T-shape intersection. (b) Curve road. (c) Straight road. (d) Y-shape intersection.

- *Zai [18]:* The supervoxel generation and road boundary extraction method
- *Yang [22]:* The feature-based search method

The quantitative evaluation and comparison are given in Table V. It is clear that the proposed method is robust and achieves the best performance. In addition, the other five methods achieve relatively poorer performance for the +-shape and Y-shape intersections than the straight road, curve road and T-shape intersections. It indicates that these methods cannot handle the curb detection problem at complex intersections.

In Zhang's method, the performance depends on the parameters that are difficult to tune. Moreover, It cannot deal with the intersection because it assumes that both sides of the road are continuous without intersections. In Perterson's method, the threshold of wavelet transformation is difficult to determine due to the different elevations on the road. In addition, wavelet transformation may produce many false positive points as shown in Table V. Although the recall of this method is higher than that of the proposed method for the straight-road scenario, the precision is too low, i.e. there are many false positive outputs. In Kang's method, the first thing to determine is the distance and angular difference between two adjacent points in the horizontal plane. However, the elevation information is not considered. it is also hard to tune the parameters. In Yang's method, the parameters of the defined rules for elevation jump, point density and slope change are difficult to identify because the vertical pointing angle of each laser in our dataset is different while the angle is constant in their dataset. In Zai's method, the proposed method is applied to extract the road boundaries according to the trajectory data as an input. In our dataset, we do not input the vehicle trajectory information because self-driving cars process the data frame by frame and do not have the future trajectory of the vehicle. In addition, our point cloud is sparse, which makes it hard to reveal the smoothness and normal vector of the supervoxel generated by the K-neighbor method. Furthermore, the processing time of Zai's method is about 60 seconds per frame because the search of K-neighbors and the computing of normal vectors for each point is time consuming.

An example in Fig. 10 illustrates the different performances of these methods. The green squares are the labeled curbs. Results of each method is shown in color dots. Zhang's method cannot deal with all lasers because there is an intersection and the method is not designed to find curbs in each segment. Peterson's method detects most of the labeled curbs, but produces many false positive points. It is thus unable to represent the curbs without further processing. Kang's method assumes a gap on a road surface as a curb point because the elevation information is not considered. Yang's method needs some parameters difficult to tune, therefore some near noise data may be detected as curbs while missing the curbs in distance. Zai's method achieves more accurate detection rate, but it is difficult to obtain an perfect result on such a sparse dataset compared to the dataset used in [18].

The off-line experiments illustrate that the proposed method is accurate and robust for various road scenarios. Besides performing the quantitative evaluation, we also tested the computation time of the proposed method, since self-driving requires the real-time processing ability.

C. Online Experiments

The proposed method is tested on our self-driving car platform, which uses *NI* PXI-8109 as the controller. The core of the controller is an *Intel* i7-620M processor with a base frequency of 2.66 GHz. The proposed method is implemented in Visual C in a Windows-7 Operating System. The online test takes about 6 minutes to process 3600 frames of LiDAR data and the trajectory is about 2.9 km long with an average speed of 28.6 km/h. We have transmitted the curbs in each frame to the data log process by Ethernet and the results are then analyzed.

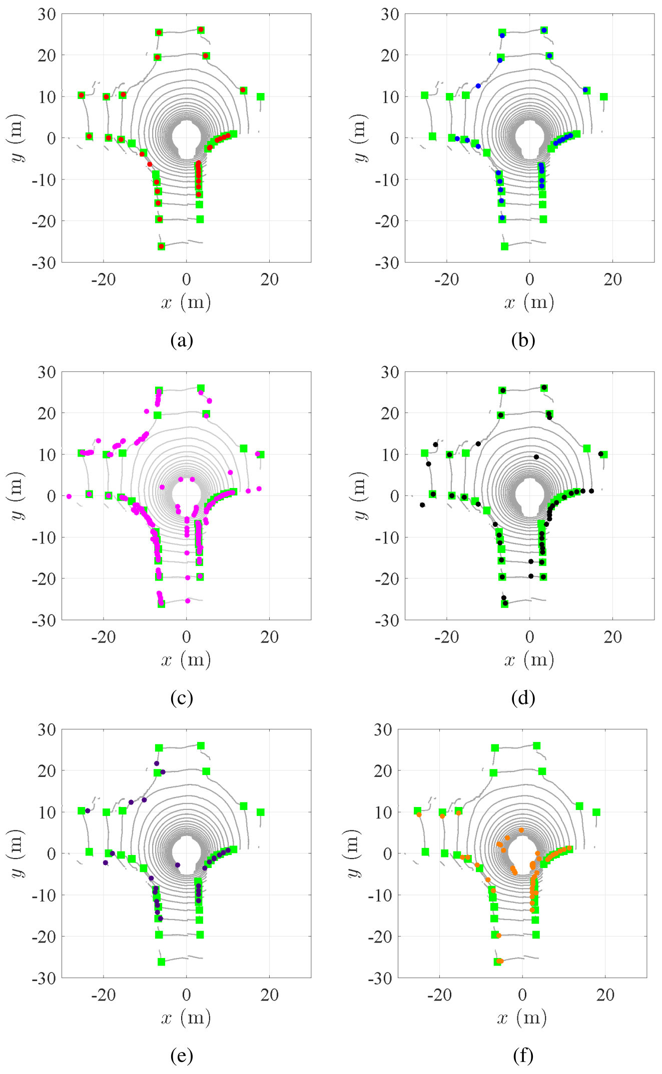


Fig. 10. Comparison results of the curb detection method in Scenario (d). (a) Proposed. (b) Zhang [11]. (c) Peterson [30]. (d) Kang [7]. (e) Zai [18]. (f) Yang [22].

The average processing time is about 12 ms, which is faster than the previous curb detection method [11] because the searching procedure and the feature calculation are optimized. To obtain the ground truth of the curb position, we used a high-resolution digital map as a benchmark, which is created by our team in [34]. The DGPS is applied to obtain an accurate position of the vehicle and all the curbs are projected onto a local map corresponding to the start point. Note that even the DGPS contains about 5 cm localization error. The average curb position error is 32.42 cm, which is not accurate overall. However, over 90% of the curbs are within 10 cm deviation, which means that there are a few false positive detections may have large distance deviations. Moreover, some false detections and miss detections are actually caused by damaged curbs on campus roads.

Experimental results are shown in Fig. 11. The curbs detected by the proposed method are shown in small red dots, the trajectory of the host vehicle is represented in blue. The start position is in black and the end position is in green. The overview of the real-time experiment is shown on the left side

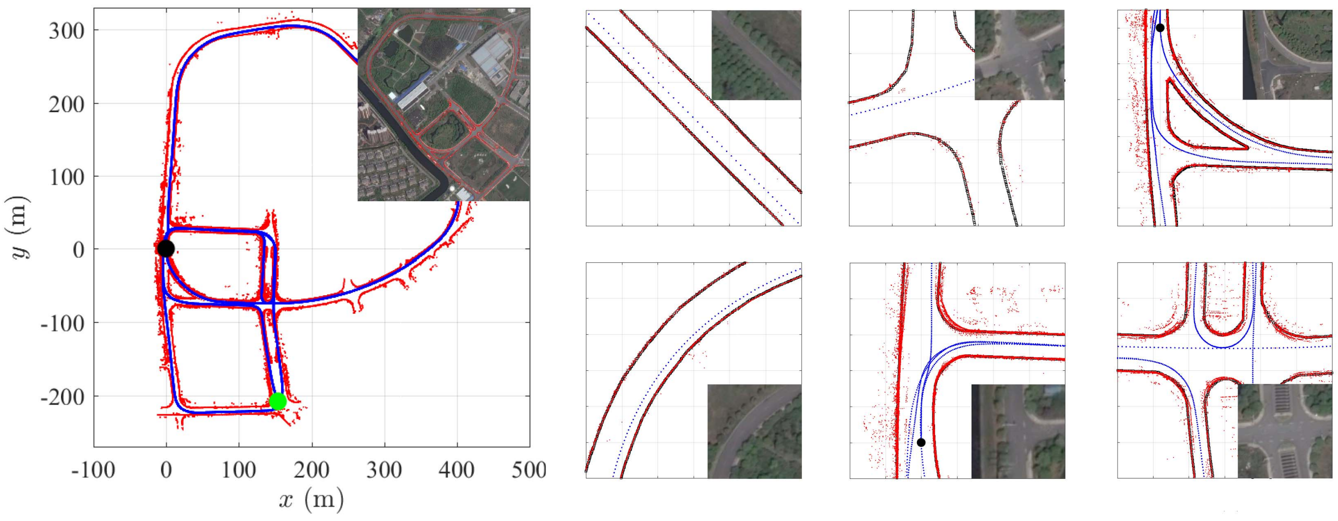


Fig. 11. Results of the real-time experiment.

with a Google Earth view attached. Six scenarios are illustrated on the right side and the small black squares are the reference curbs on the high-resolution map. As shown in the figure, the detected curbs are accurate in the straight and the curve roads while there are some noise in other complex scenarios. One main reason is the localization error from DGPS, and the localization error may produce a slight deviation to the projection of the curbs when the vehicle travels the same intersection in multiple times.

V. Conclusion and Future Work

This paper develops a real-time curb detection method for self-driving based on the road segmentation. The method captures the road curbs in various road scenarios including straight roads, curved roads, T-shape intersections, Y-shape intersections and +-shape intersections. The curb information forms the foundation of decision making and path planning for autonomous driving. Comprehensive off-line and real-time experiments demonstrate that the proposed method achieves high accuracy of curb detection in various scenarios while satisfying the stringent efficiency requirements of autonomous driving. The off-line experiment demonstrates that the curbs can be robustly extracted. The average precision is 84.89%, the recall is 82.87%, and the average F_1 score is 83.73%. Furthermore, the average processing time in the real-time experiments is around 12 ms per frame, which is fast enough for self-driving.

In order to reduce the false positive detections, the context information of the road need to be considered and a filtering method should be adopted to improve the detection results. Furthermore, it is still a challenge to detect the curbs with obstacles such as other vehicles or pedestrians on the road. The detection method will be further modified and extended to enhance its accuracy and robustness under such circumstances. In addition, road context should be taken into consideration to filter the outliers and thereby improve the accuracy of the curb position.

Acknowledgment

The authors would like to thank William Seto in Carnegie Mellon University for his helpful reviews and discussions.

References

[1] F. Oniga, S. Nedevschi, and M. M. Meinecke, "Curb detection based on a multi-frame persistence map for urban driving scenarios," in *Proc. 11th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2008, pp. 67–72.
[2] J. Siegemund, D. Pfeiffer, U. Franke, and W. Forstner, "Curb reconstruction using conditional random fields," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2010, pp. 203–210.
[3] L. Wang, T. Wu, Z. Xiao, L. Xiao, D. Zhao, and J. Han, "Multi-cue road boundary detection using stereo vision," in *Proc. IEEE Int. Conf. Veh. Electron. Safety (ICVES)*, Jul. 2016, pp. 1–6.
[4] M. Nikolova and A. Hero, "Segmentation of a road from a vehicle-mounted radar and accuracy of the estimation," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Oct. 2000, pp. 284–289.
[5] B. Ma, S. Lakshmanan, and A. O. Hero, "Simultaneous detection of lane and pavement boundaries using model-based multisensor fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 3, pp. 135–147, Sep. 2000.
[6] W. Zhang, "LiDAR-based road and road-edge detection," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2010, pp. 845–848.
[7] Y. Kang, C. Roh, S.-B. Suh, and B. Song, "A LiDAR-based decision-making method for road boundary detection using multiple Kalman filters," *IEEE Trans. Ind. Electron.*, vol. 59, no. 11, pp. 4360–4368, Nov. 2012.
[8] W. Yao, Z. Deng, and L. Zhou, "Road curb detection using 3D LiDAR and integral laser points for intelligent vehicles," in *Proc. Joint 6th Int. Conf. Soft Comput. Intell. Syst. (SCIS)*, Nov. 2012, pp. 100–105.
[9] C. Fernández, R. Izquierdo, D. F. Llorca, and M. A. Sotelo, "Road curb and lanes detection for autonomous driving on urban scenarios," in *Proc. IEEE 17th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2014, pp. 1964–1969.
[10] G. Zhao and J. Yuan, "Curb detection and tracking using 3D-LiDAR scanner," in *Proc. 19th IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2012, pp. 437–440.
[11] 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," in *Proc. IEEE Conf. Control Appl. (CCA)*, Sep. 2015, pp. 1020–1025.
[12] R. Huang, J. Chen, J. Liu, L. Liu, B. Yu, and Y. Wu, "A practical point cloud based road curb detection method for autonomous vehicle," *Information*, vol. 8, no. 3, p. 93, 2017.
[13] S. Xu, R. Wang, and H. Zheng, "Road curb extraction from mobile LiDAR point clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 996–1009, Feb. 2017.
[14] H. Guan, J. Li, Y. Yu, M. Chapman, and C. Wang, "Automated road information extraction from mobile laser scanning data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 194–205, Feb. 2015.

[15] H. Wang *et al.*, "Road boundaries detection based on local normal saliency from mobile laser scanning data," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 10, pp. 2085–2089, Oct. 2015.

[16] B. Yang, Y. Liu, Z. Dong, F. Liang, B. Li, and X. Peng, "3D local feature BKD to extract road information from mobile laser scanning point clouds," *ISPRS J. Photogramm. Remote Sens.*, vol. 130, pp. 329–343, Aug. 2017.

[17] G. Máttyus, S. Wang, S. Fidler, and R. Urtasun, "HD maps: Fine-grained road segmentation by parsing ground and aerial images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3611–3619.

[18] D. Zai *et al.*, "3-D road boundary extraction from mobile laser scanning data via supervoxels and graph cuts," *IEEE Trans. Intell. Transp. Syst.*, to be published.

[19] M. Cheng, H. Zhang, C. Wang, and J. Li, "Extraction and classification of road markings using mobile laser scanning point clouds," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 3, pp. 1182–1196, Mar. 2017.

[20] Y. Yu, J. Li, H. Guan, F. Jia, and C. Wang, "Learning hierarchical features for automated extraction of road markings from 3-D mobile LiDAR point clouds," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 2, pp. 709–726, Feb. 2015.

[21] H. Guan, J. Li, Y. Yu, C. Wang, M. Chapman, and B. Yang, "Using mobile laser scanning data for automated extraction of road markings," *ISPRS J. Photogramm. Remote Sens.*, vol. 87, pp. 93–107, Jan. 2014.

[22] B. Yang, L. Fang, and J. Li, "Semi-automated extraction and delineation of 3D roads of street scene from mobile laser scanning point clouds," *ISPRS J. Photogramm. Remote Sens.*, vol. 79, pp. 80–93, May 2013.

[23] C. Rasmussen, "Road shape classification for detecting and negotiating intersections," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2003, pp. 422–427.

[24] M. S. Kurdziel, *A Monocular Color Vision System for Road Intersection Detection*. Ann Arbor, MI, USA: ProQuest, 2008.

[25] K. R. S. Kodagoda, W. S. Wijesoma, and A. P. Balasuriya, "Road curb and intersection detection using a 2D LMS," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, vol. 1. Sep. 2002, pp. 19–24.

[26] Q. Zhu, L. Chen, Q. Li, M. Li, A. Nüchter, and J. Wang, "3D LiDAR point cloud based intersection recognition for autonomous driving," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2012, pp. 456–461.

[27] Q. Li, L. Chen, Q. Zhu, M. Li, Q. Zhang, and S. Ge, "Intersection detection and recognition for autonomous urban driving using a virtual cylindrical scanner," *IET Intell. Transp. Syst.*, vol. 8, no. 3, pp. 244–254, 2014.

[28] T. Chen, B. Dai, D. Liu, and Z. Liu, "LiDAR-based long range road intersection detection," in *Proc. 6th Int. Image Graph. Conf.*, Aug. 2011, pp. 754–759.

[29] S. Tourani, F. Chhaya, and K. M. Krishna, "Linear-chain CRF based intersection recognition," in *Proc. Int. Conf. Veh. Electron. Safety (ICVES)*, 2014, pp. 106–111.

[30] K. Peterson, J. Ziglar, and P. E. Rybski, "Fast feature detection and stochastic parameter estimation of road shape using multiple LiDAR," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2008, pp. 612–619.

[31] Y.-W. Seo and C. Urmson, "A perception mechanism for supporting autonomous intersection handling in urban driving," in *Proc. IEEE/RSJ Int. Conf. Int. Robots Syst. (IROS)*, Sep. 2008, pp. 1830–1835.

[32] Y. Zhang, J. Wang, X. Wang, C. Li, and L. Wang, "3D LiDAR-based intersection recognition and road boundary detection method for unmanned ground vehicle," in *Proc. 18th Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2015, pp. 499–504.

[33] *Velodyne HDL 32-E LIDAR*. Accessed: Jan. 17, 2018. [Online]. Available: http://velodynelidar.com/hdl-32e.html

[34] L. Wang, Y. Zhang, and J. Wang, "Map-based localization method for autonomous vehicles using 3D-LiDAR," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 276–281, 2017.

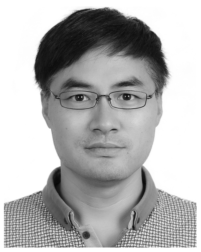
[35] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[36] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2005.

[37] J. Hu, A. Razdan, J. C. Femiani, M. Cui, and P. Wonka, "Road network extraction and intersection detection from aerial images by tracking road footprints," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 12, pp. 4144–4157, Dec. 2007.

Yihuan Zhang received the B.S. degree in automation from Nanjing Normal University in 2013. He is currently pursuing the Ph.D. degree with the Department of Control Science and Engineering, Tongji University. He was a Visiting Scholar at Carnegie Mellon University from 2016 to 2017. His research interests include environment perception, behavior estimation, and decision making in area of autonomous vehicles.

Jun Wang (S'98–M'03–SM'12) received the Ph.D. degree in control engineering from the University of Leeds, U.K., in 2003. He has been a Professor in control engineering with the Department of Control Science and Engineering, Tongji University, since 2010. His research interests are smart sensing and intelligent control in the areas of autonomous vehicles and renewable energy. He is on the Editorial Board of the *International Journal of Driving Science*.



Xiaonian Wang received the B.S. and M.S. degrees in control engineering from the Xi'an University of Technology in 1998 and 2003, respectively, and the Ph.D. degree from Xi'an Jiaotong University in 2007. He is currently a Lecturer with the Department of Control Science and Engineering, Tongji University. His research interests are pattern recognition for autonomous vehicles.

John M. Dolan received the B.S.E. degree from Princeton University, Princeton, NJ, USA, in 1980, and the M.E. and Ph.D. degrees from Carnegie Mellon University (CMU), Pittsburgh, PA, USA, in 1987 and 1991, respectively, all in mechanical engineering. He is a Principal Systems Scientist with the Robotics Institute, CMU. His research interests include autonomous driving, multi-robot cooperation, human–robot interaction, robot reliability, and sensor-based control. He was the Behaviors Leader for the Carnegie Mellon's Tartan Racing Team in the 2007 DARPA Urban Challenge.