CURB DETECTION AND TRACKING USING 3D-LIDAR SCANNER

Gangqiang Zhao and Junsong Yuan

School of Electrical and Electronic Engineering,
Nanyang Technological University, Singapore, 639798

ABSTRACT

This paper presents a novel road curb detection method using 3D-LIDAR scanner. To detect the curbs, the ground points are separated from the pointcloud first. Then the candidate curb points are selected using three spatial cues: the elevation difference, gradient value and normal orientation. Afterwards the false curb points caused by obstacles are removed using the short-term memory technique. Next the curbs are fitted using the parabola model. Finally, the particle filter is used to smooth the curb detection result. The proposed approach was evaluated on a dataset collected by an autonomous ground vehicle driving around the Ford Research campus and downtown Dearborn. Our curb detection results are accurate and robust despite variations introduced by moving vehicles and pedestrians, static obstacles, road curvature changes, etc.

***Index Terms*—** Curb detection, 3D-LIDAR, pointcloud, spatial cue, particle filter

1. INTRODUCTION

Worldwide, an estimated 1.2 million people are killed in road crashes each year and as many as 50 million are injured [1]. To reduce this huge number of traffic accidents, it is imperative to improve the automated driving techniques. Obstacle detection and avoidance is one key problem for automated driving. As the delimiters of the drivable area, the curb position information is very important for obstacle localization and categorization.

Several existing approaches address the curb detection problem using video sensors (monocular or stereo), laser scanner, or a mixture of these sensors. Shin et al. [2] extract the curbs by the 1D laser scanner. Oniga et al. [3] use a stereo vision camera and the Canny algorithm to find the curbs directly. Siegemund et al. [4] also find the curbs by using a stereo vision camera. The conditional random field (CRF) is employed to propagate the curb detection result. However, this method relies on the assumption that there are at most two visible curbs. Aufrere et al. [5] detect the curbs using a 1D laser scanner and a camera together. The drawback of all the above methods is that they produce a lot of false positive detection.

In this research, we use the 3D pointcloud data obtained by the Velodyne 3D-LIDAR (Light Detection And Ranging) scanner [6], which can produce accurate but sparse pointcloud. Fig.1 shows the testbed vechicle and one sample keyframe. Curb detection using this senor is a new problem and has not been well studied. The senor is only used to help the vision based lane detection in MIT's driverless car [7]. In this paper, we propose a novel road curb detection method using the Velodyne 3D-LIDAR scanner. To detect the road curbs, we first separate the ground points from the pointcloud. Then we select the curb points using three spatial cues: elevation difference, gradient value and normal orientation. Then the false positives caused by obstacles are removed using the short-term memory technique. Next, the curbs are fitted using the parabola model. Finally, the particle filter is used to smooth the curb detection result. To the best of our knowledge, the proposed approach is the first systematic curb detection work using the Velodyne 3D-LIDAR scanner.

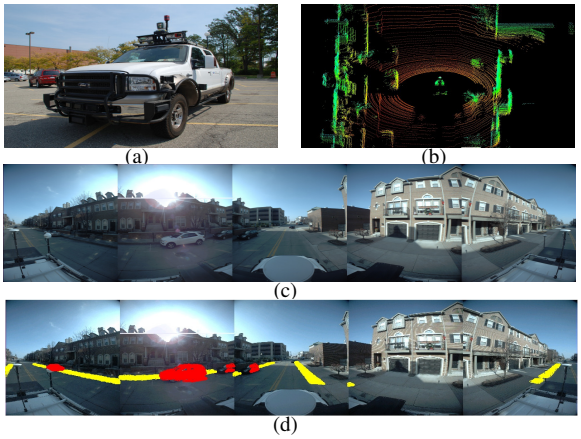
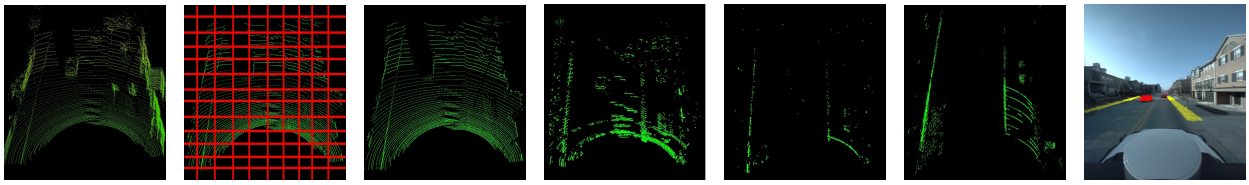


Fig. 1. (a) shows the autonomous vehicle testbed; (b) shows the 3D pointcloud of one keyframe generated by the Velodyne 3D-LIDAR scanner; (c) shows the corresponding stacked images from the five sensors of the omni-directional camera; (d) shows the detected curbs (yellow region) and the possible obstacles on the road (red region).

There are several advantages of our method. First of all, unlike existing methods that rely on the elevation difference or edge detection only, our method considers several kinds of spatial-temporal cues together. It can automatically detect and locate road curbs without knowing their shapes, lengths, or their distances to the car. Moreover, it can handle curb variations introduced by moving vehicles and pedestrians, static obstacles, road curvature changes, etc. Finally, it does not require much *a priori* information but have a real time performance. We test our approach on a dataset collected by an



(a) (b) (c) (d) (e) (f) (g)

Fig. 2. (a) is the 3D pointcloud; (b) is the 2D voxel grid; (c) is the ground points; (d) is the curb points selected using the elevation difference; (e) is the curb points selected using three cues; (f) is the combined curb points; (g) is the detected curbs in the camera image.

autonomous ground vehicle driving around the Ford Research campus and downtown Dearborn. The results validate the robustness and effectiveness of our method.

2. CURB DETECTION

To detect the road curbs, we first separate the ground points from the pointcloud. Then we select the candidate curb points using three spatial cues: the elevation difference, gradient value and normal orientation. Then we reject the curb points caused by obstacles using the short-term memory technique. After that, the ridge points are selected and the curbs are fitted using the parabola model.

2.1. Ground points separation

The real time performance is very important for automated driving. To speed up the process, we first build a 3D cubic voxel grid using the pointcloud. The pointcloud data are stored in cubic voxels for efficient retrieval. By voxelizing, the spatial neighborhood relations of the 3D points are modelled explicitly. A voxel is considered to contain ground data if the voxel is a member of the lowest (in elevation) set of adjacent non-empty voxels in a vertical column (i.e. not part of an overhang). All 3D points stored in that set of voxels are fitted to a plane and the inliers are the ground points. Denote these ground voxels as V, which look like a 2D grid in the horizontal plane if we do not consider the elevation information, as shown in Fig.2(b). One ground voxel $V_{i,j} \in V$ may contain a number of ground points or be an empty voxel, where i denotes the row and j denotes the column in the 2D grid. The width and height of the voxel are same and denoted as S. Fig.2(c) shows the ground points.

All of the voxels that contain data, but are not considered to be part of the ground surface are called the non-ground voxels. These non-ground voxels are then geometrically segmented by partitioning them according to 3D adjacency [8]. All the non-ground segments are the possible obstacles.

2.2. Curb points selection using spatial-temporal cues

The curb points are different from other ground points in many aspects. First, the elevation difference is larger in the curb region. Second, the elevation changes quickly in the direction perpendicular to the curb. Third, the curb surface always points to the road center. Based on these observations, we propose to select the curb points from the ground points using the elevation difference, elevation gradient value and surface normal orientation.

To select the curb points, the first criterion is the elevation difference of each ground voxel. Denote the elevation difference of the ground voxel $V_{i,j}$ as $E_{i,j}$. The ground points of this voxel are selected if the elevation differs within a reasonable range, i.e., $E_{i,j} > 0.05m$ and $E_{i,j} < 0.25m$ in our experiment, where m represents one meter. Fig.2(d) shows the curb points selected using the elevation difference.

We use the elevation gradient to approximate the local elevation change rate. To calculate the gradient, we compute the average elevation of all points inside each voxel and use it to represent the elevation of this voxel. The voxel grid is transformed to a 2D elevation map I. The raw elevation map is convolved with a Gaussian filter first. Then the 3×3 Sobel operator is used to obtain the gradient in the horizontal direction (G_x) and the vertical direction (G_y). Due to the sparsity of the points, it is not appropriate to directly use the Sobel operator. Therefore, we need to check the validation of the neighborhood for the operator. The checking is done for each pair of neighbors in the elevation map. Take calculating the gradient for voxel $I_{i,j}$ as an example, if $I_{i,j-1}$ or $I_{i,j+1}$ has no elevation value, the G_x Sobel operator $\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$ is changed to $\begin{bmatrix} -1 & 0 & +1 \\ 0 & 0 & 0 \\ -1 & 0 & +1 \end{bmatrix}$. In other words, only when one pair of voxels are valid, we will compute the gradient using them. Otherwise, it is not considered. The $Mask$ operator represents the check process when computing the gradient:

$$G_x = I * Mask * \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}. \quad (1)$$

One point is selected if the gradient of its voxel is in a desired interval (equivalent to an elevation variation between 0.05 and 0.25 meter).

The third criterion is the surface normal. To compute the normal for one ground point, the Principal Component Analysis method is used, and all points located in the same voxel are considered as its neighbors. Specifically, for each neighbor point p_i, we assemble the covariance matrix Cov as:

$$Cov = \frac{1}{k} \sum_{i=1}^{k} (p_i - \overline{p}) \cdot (p_i - \overline{p})^T, \quad (2)$$

where k is the number of points located in the same voxel, $\overline{p}$ is the 3D centroid of the neighbors. The normal is then the eigenvector associated with the smallest eigenvalue. One point is selected if its normal is perpendicular to the car. Fig.2(e) shows the curb points which satisfy all three criteria.

After obtaining the candidate curb points for each frame, we can further remove the noise points produced by the obstacles: the presence of a large vertical discontinuity, perhaps caused by the car ahead, does not indicate a road curb. We reject these false positives by using short-term memory technique: if a given voxel has ever been classified as part of the road, then any detections of a vertical discontinuity in that voxel do not mask road curbs when the voxel remains in sensor range. Due to the small ego motion between successive frames, we can combine the curb points of several previous frames with that of current one. The combined curb points are shown in Fig.2(f) and this image is called the curb evidence map. The curb evidence map has the same voxel grid as Fig.2(b).

2.3. Parabola fitting

The obtained curb evidence map represents the possibility of a particular ground region to be a curb point. In each row of the evidence map, we select two voxels with the maximal evidence score as the ridge points. Here the row's direction is perpendicular to the car's forward direction. One ridge point is selected on the left of the car and the other is selected on the right. Denote all the 2D ridge points on the one side as R. Due to the inaccuracy of the pointcloud and the motion of the car, these ridge point detections are prone to false positives. Therefore, we model the curb as a parabolic and use a random sample consensus (RANSAC) algorithm to remove the false positives that do not match the parabolic model. At each RANSAC iteration, three ridge points are randomly selected for a parabola fitting. To determine the set of inliers for a parabola, we first compute its conic coefficient matrix C [7], and define the set of candidate inliers L to contain the ridge points within distance S:

$$L = \{r \in R : r^T C r < S\}, \tag{3}$$

where S is the width and height of each voxel. The parabola is then re-fitted to L using a linear least squares method, and a new set of candidate inliers is computed. After a number of RANSAC iterations, the parabola with the greatest score is selected as a candidate curb model.

3. PARTICLE FILTER BASED CURB TRACKING

After the curb detection in one frame, the particle filter method [9] is used to track curbs and update the parameters of the curb model. Other techniques, such as Kalman filtering, can also be used within this framework. However, we choose the particle-filtering over the Kalman filter to prevent the result from being biased too much by the predicted vehicle motion but to give more weight to the curb evidence.

Instead of tracking the whole curb, we track the ridge points that belong to the curb. Each ridge point is represented by dozens of particles. To use the particle filter algorithm, first we need to initialize the particle set using the result of the previous frame, and then predict the particles in the current frame. Finally we have to resample the particles based on their weights by using the observation model. For each ridge point, we have the following steps:

1) Initialization of the particles: After the curb detection on one frame, the particles are selected based on the inlier ridge points. The state vector can be defined by $X_t = X_t^1, X_t^2, \ldots, X_t^n$, where $X_t^i = [x_t^i, y_t^i]^T$ is the 2D coordinate of particle i, n is the number of particles.

2) State prediction: Assuming the change of curb model for two consecutive frames is small, a normal distribution can be used to model the state transition of particles as:

$$p(X_t|X_{t-1}) = \mathcal{N}(AX_{t-1}, \Sigma), \tag{4}$$

where $\mathcal{N}$ is a normal distribution, the matrix A is the translation matrix obtained by the navigation system, and Σ is the covariance, which handles the difference of curb between two consecutive frames.

3) Observation Model: For the i_{th} particle, the measurement is denoted as z_i. The $p(z_t^i|X_t^i)$ specifies the likelihood of one position being near a curb point. Here, we use the evidence score to approximate this likelihood and the observation model can be derived:

$$p(z_t^i|X_t^i) \propto e^{-M}, \tag{5}$$

where M is the value of X_t^i in the curb evidence map.

4) Resampling: For every particle, we can compute its weight using the observation model Eq.5:

$$w_t^i = \eta\, p(z_t^i|X_t^i), \tag{6}$$

where η is a normalization factor. After resampling, the particle that has the highest weight will be kept, and the others will be removed from the particle set X_t. All the predected ridge points using particle filter will be added to the ridge points of the current frame unless they have left the sensor range.

4. PERFORMANCE EVALUATION

To evaluate our approach, we test it on challenging real-world 3D-LIDAR dataset. In addition, we compare our approach with the state-of-the-art methods [3][7].

4.1. Dataset

In order to evaluate our algorithm in a real-world scenario, we present results using the Ford Campus Vision and LIDAR Data Set [6]. This data set was collected with a Ford autonomous ground vehicle outfitted with a Velodyne 3D-LIDAR scanner, an omni-directional camera and other sensors, as shown in Fig. 1(a). This dataset includes two subsets, the first one corresponds to a loop in the downtown Dearborn Michigan while the second one corresponds to a loop inside the Ford Research campus in Dearborn Michigan.

4.2. Parameters setting

Several parameters should be set first. The width and length S of each voxel is set to be 0.2 meter. To obtain the evidence map, we combine the 5 previous frames with current one. The number of RANSAC iterations is set to be 200. 50 particles are used to track each ridge point and the covariance matrix Σ

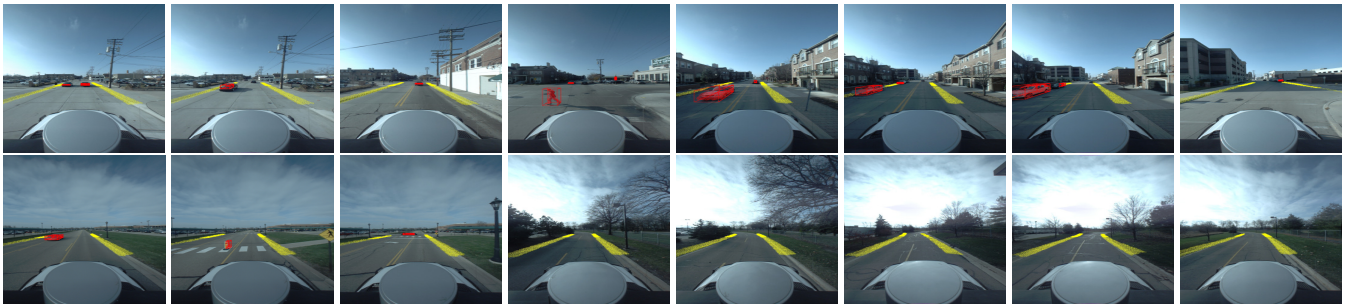


Fig. 3. Sample results of curb detection in the corresponding camera images. The yellow regions show the detected curbs and the red regions show the possible obstacles. The top row shows the results of sub-dataset1 while the bottom row shows the results of sub-dataset2.

is a diagonal matrix whose diagonal entries are set to be 0.1. The experiments are performed on a Xeon 2.67GHz PC and our approach can process 8 frames in one second. To quantify the performance, we manually labelled the ground truth positions of the curbs in about 20% of all keyframes. The performance is measured by the average distance between the ground truth curb points and the parabola curb model.

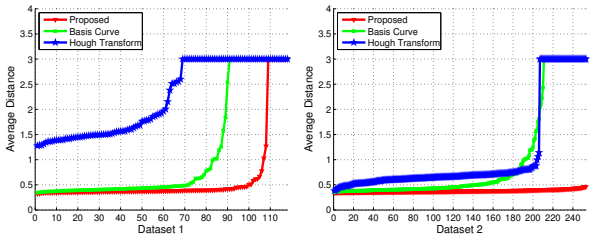
4.3. Curb detection

We evaluate the method on a sequence of 600 frames in the first sub-dataset and a sequence of 1000 frames in the second sub-dataset. Fig.3 shows the result of several keyframes. The yellow regions show the detected curbs and the red regions show the obstacles which are on the road. In the sequences, the road curbs are subject to variations introduced by moving vehicles and pedestrians, static obstacles, road curvature changes, etc. It is possible that some keyframes contain only one curb and some keyframes do not contain any curbs. Table 1 shows the number of total keyframes and the number of correctly detected curbs for each sequence. It can be seen that the proposed method has a detection rate of about 98%. These results show that the proposed approach performs well for detecting curbs from real-world road data.

To further evaluate the propose method, we compare it with two other methods: (1) Basis Curve approach and (2) Hough Transform approach. The Basis Curve approach is the state-of-the-art approach for curb detection using LIDAR dataset [7]. This method uses the elevation difference to select the ridge point and uses the Kalman Filter to smooth the detection result. Hough Transform is a classical method for line detection and it is used for curb detection in [3]. As shown in Fig. 4, our proposed approach outperforms both Basis Curve approach and Hough Transform approach in terms of the average distance. The Basis Curve approach does not consider the spatial cues such as gradient and road surface normal, while the Hough Transform approach has difficulty to handle high curvature roads. These comparisons clearly demonstrate the advantages of the proposed curb detection method.

5. CONCLUSIONS

A novel curb detection algorithm using 3D-LIDAR was presented. It employs several spatial and temporal cues to detect the curb points and uses particle filter to track the curbs.



(a) (b)

Fig. 4. (a)shows the performance comparison of our approach, Basis Curve approach and Hough Transform approach for sub-dataset1, (b) shows the comparison for sub-dataset2. The vertical axis shows the average distance of each keyframe. For each approach, the keyframes are ordered accoring to their average distances.

Table 1. Detection accuracy on two sub-datasets.

Dataset	Labeled Frame No.	Correctly detected No.
Sub-dataset1	118	109
Sub-dataset2	253	253

Experiments on a challenging road dataset demonstrate the accuracy and robustness of the curb detection approach.

6. ACKNOWLEDGMENT

This work was supported in part by the Nanyang Assistant Professorship (SUG M58040015) to Dr. Junsong Yuan, and JSPS-NTU project M4080882.040. We thank Jingjing Meng to help proof read the paper.

References

[1] Margie Peden et al., "World report on road traffic injury prevention: summary," World Health Organization, 2004, pp. 1–66.

[2] Y. Shin et al., "Drivable road region detection using a single laser range finder for outdoor patrol robots," in *Intelligent Vehicles Symposium(IV)*. IEEE, 2010.

[3] F. Oniga et al., "Curb detection based on a multiframe persistence map for urban driving scenarios," in *Intelligent Transportation Systems*. IEEE, 2002.

[4] J. Siegemund, D. Pfeiffer, U. Franke, , and W. Forstne, "Curb reconstruction using conditional random fields," in *Intelligent Vehicles Symposium*. IEEE, 2010.

[5] R. Aufrere et al., "Multiple sensor fusion for detecting location of curbs, walls, and barriers," in *Intelligent Vehicles Symposium*. IEEE, 2003, pp. 203–210.

[6] Gaurav Pandey, James R Mcbride, and Ryan M Eustice, "Ford campus vision and lidar data set," *Int. J. Rob. Res.*, pp. 1543–1552.

[7] Albert Shuyu Huang, "Lane estimation for autonomous vehicles using vision and lidar," in *Thesis of Massachusetts Institute of Technology*, 2010.

[8] Bertrand Douillard et al., "A pipeline for the segmentation and classification of 3d point clouds," in *International Symposium on Experimental Robotics 2010*, 2010.

[9] K. ZuWhan, "Robust lane detection and tracking in challenging scenarios," in *IEEE Transactions on Intelligent Transportation Systems*, 2008, pp. 16–26.