Journal of Intelligent & Robotic Systems (2021) 102: 45
https://doi.org/10.1007/s10846-021-01381-7

SHORT PAPER

# A Framework for Drivable Area Detection Via Point Cloud Double Projection on Rough Roads

Fengyu Xu[1,2] · Huawei Liang[1,3,4] · Zhiling Wang[1,3,4] · Linglong Lin[1,3,4]

Received: 15 October 2020 / Accepted: 26 March 2021 / Published online: 25 May 2021

**Abstract**
Drivable area detection is one of the essential functions of autonomous vehicles. However, due to the complexity and diversity of unknown environments, it remains a challenge specifically on rough roads. In this paper, we propose a systematical framework for drivable area detection, including ground segmentation and road labelling. For each scan, the point cloud is projected onto two different planes simultaneously, generating an elevation map and a range map. Different from the existing methods based on mathematical models, we accomplish the ground segmentation using image processing methods. Subsequently, road points will be filtered out from ground points and used to generate the road area with the assistance of a range map. Meanwhile, a two-step search method is utilized to create the reachable area from an elevation map. For the robustness of drivable area detection, Bayesian decision theory is introduced in the final step to fuse the road area and the reachable area. Since we explicitly avoid complex three-dimensional computation, our method, from both empirical and theoretical perspectives, has a high real-time capability, and experimental results also show it has promising detection performance in various traffic situations.

**Keywords** Drivable area detection · Ground estimation · Road labelling · Bayesian decision

## 1 Introduction

In the field of autonomous driving, drivable area detection is a key and fundamental task, especially in complex scenes. The main application of the drivable area is to detect whether a feasible trajectory exists and limit the search of trajectories, which are the critical work for navigating autonomous vehicles across unknown and unstructured environments [1, 2]. For that purpose, the drivable area detection should be accurate and robust. However, the challenge of achieving high detection performance on rough roads is considerably greater than that in structured roads because the former is more complex and irregular. And so far, to the best of our knowledge, camera and LiDAR (Light Detection and Ranging) are the two primary sensors to address these challenges [3–6]. The LiDAR-based drivable area detection has gained an increasing amount of attention, but most of the related works focus on ground segmentation. Meanwhile, researchers generally regard drivable area detection as a road detection task in camera-based methods. However, considering their limiting on trajectory search, the drivable area is significantly different from the ground surface or the road area. Moreover, the noise caused by a bump on rough roads also jeopardizes the performance, especially for camera-based methods. Compared with the camera, 3D LiDAR can provide accurate measurements of the surrounding environment and has less sensitivity to illumination. For the above reasons, an autonomous vehicle could make safe decisions and achieve reliable navigation with the drivable area generated by LiDAR-based methods.

✉ Zhiling Wang
zlwang@hfcas.ac.cn

✉ Linglong Lin
linll@iim.ac.cn

Fengyu Xu
xfy0032@mail.ustc.edu.cn

Huawei Liang
hwliang@iim.ac.cn

[1] Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230088, China

[2] University of Science and Technology of China, Hefei 230026, China

[3] Anhui Engineering Laboratory for Intelligent Driving Technology, Hefei 230088, China

[4] Application and Innovation Research Institute of Robotics and Intelligent Manufacturing C. A. S, Hefei 230088, China

Therefore, we prefer a LiDAR-based solution to dress the challenge of drivable area detection on rough roads.

## 1.1 Previous Work

Over the past decade, extensive research about drivable area detection has been carried out with various sensors such as monocular camera, stereo camera, and 3D LiDAR, leading to varying levels of success. In the following content, we will summarize these methods and analyze their characteristics and applicable scenarios.

### 1.1.1 Methods Based on HD Map

High-Definition (HD) map provides strong priors, for example, drivable area, lane direction, and traffic light location. Yang et al. [7] introduce a single-stage detection algorithm that extracts semantic and geometric features from traffic HD maps, which can be used to detect the drivable area. Chang et al. [8] present a dataset designed to support autonomous vehicle perception tasks, namely Argoverse. Argoverse includes binary drivable area maps at 1-m grid resolution, covering the region where vehicles can drive. Therefore, some perception missions such as target recognition and tracking can be restricted in the drivable area specified by the HD map. Although HD maps can provide accurate drivable area, a fatal defect is that the cost of constructing HD maps is very high, which leads the HD map might not be available everywhere.

### 1.1.2 Methods Based on Camera

For drivable area detection, it is generally regarded as road detection or curb detection in purely camera-based methods, and these methods mainly differ from each other based on the type of cameras such as stereo camera, binocular camera, and monocular camera [3]. Moreover, most of these methods rely on vision conceptions or traffic signs [9] like colour cue, lane lines, etc., but they are scarce on rural and unstructured roads. With the development of artificial intelligence technology, some image segmentation methods using the deep convolutional neural network such as FCN and DeepLab have significantly improved the performance of road detection [10]. However, due to the lack of accurate distance information, it is difficult for camera-based methods to describe the drivable area accurately, whether traditional methods or deep learning-based methods. Besides, we also find some drivable area detection methods based on the combination of camera and LiDAR [9, 11]. Still, they depend enormously on the precise spatial and temporal calibration between sensors, which makes these methods susceptible to the vehicle vibration and road texture.

### 1.1.3 Methods Based on LiDAR

In terms of autonomous vehicles, LiDAR has shown significant results and plays a vital role in environment perception since it offers the ability to acquire massive scale 3D geometrical information. However, most LiDAR-based methods consider the drivable area detection as a road segmentation or ground estimation task. Traditional point cloud segmentation methods summarized in the literature [12] tend to segment point clouds directly in 3D space. For example, the method in [13] fits the points whose vertical height meets certain conditions into a plane, which will be regarded as the ground. Meanwhile, more researchers focus on modelling the ground with complex mathematical models like Spatio-temporal conditional random field [14], Gaussian process in polar grid map [15], and Markov random field [16]. However, due to the complexity of mathematical methods, such as the use of belief propagation for inference on MRF [17], their real-time capability cannot be guaranteed, which is a disaster for autonomous vehicles. Similar to camera-based methods, various methods using deep learning have emerged in the field of point cloud processing, such as a specific deep learning framework for off-road drivable area extraction [18], a fully semantic segmentation network whose output category includes ground [19] and a lightweight CNN model for road-object semantic segmentation [20]. Nevertheless, most methods using deep learning belong to the supervised approach, so training data is critical for the final performance, and their application scenes need to be similar to the training scenes, which limits the general applicability of these methods.

In addition to the above methods, the point cloud processing methods based on projection transform have also been widely used in ground segmentation and obstacle detection. Depending on the projection plane, there are two main projection approaches, see Fig. 1. One is to project point clouds onto an assumed or estimated ground plane, which produces an elevation map [21]. The other is to project point clouds onto a virtual cylinder whose axis is the rotational axis of the scanner [22], and its result is called depth image or range map. Considering the processing strategy of points in the grid cell, the elevation map, also known as a grid-based map, can be divided into classical Max-Min elevation map [23], mean elevation map, multi-level elevation map, etc. [24] Since the point cloud in the distance is sparse, these approaches naturally suffer from an under-segmentation problem. To handle this problem, Bogoslavskyi et al. [25] focus on the spatial neighbourhood relation in the range map. Although methods based on the range map have excellent real-time performance and allow to segment even sparse point clouds, they are deficient in good robustness on rough roads and cannot tell whether a plane belongs to the ground or the smooth part on obstacles [26].

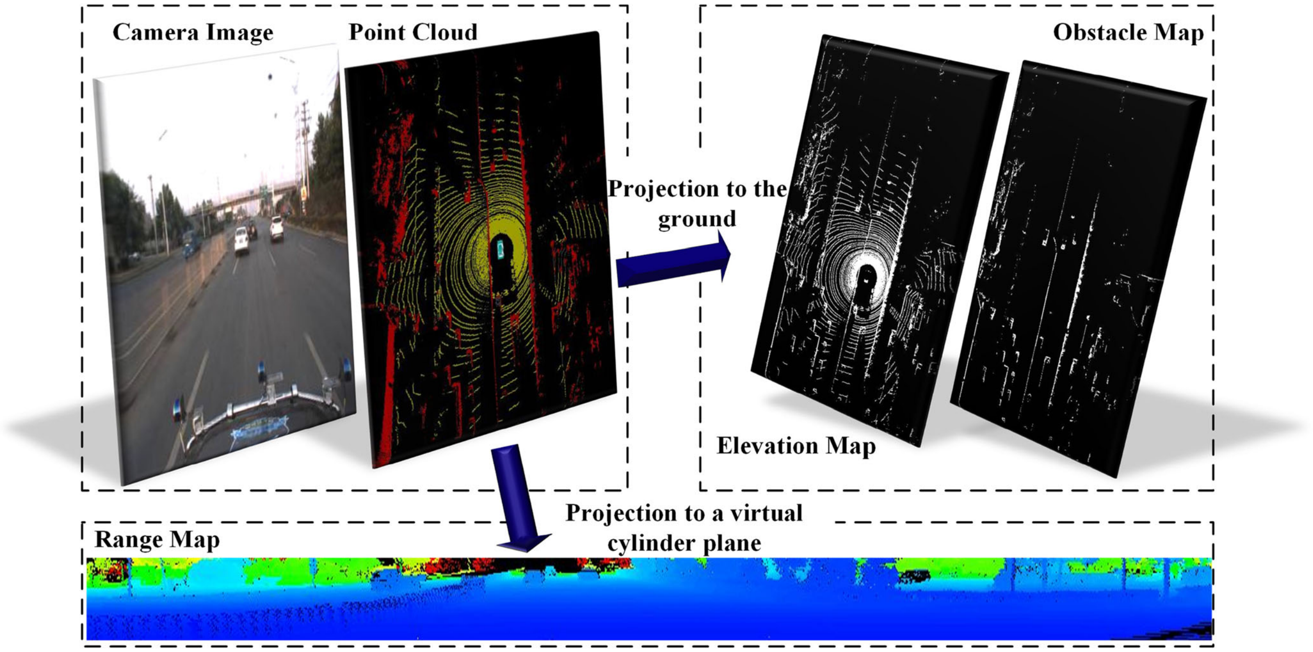


**Fig. 1** Types of point cloud projection: projection to the ground and projection to a virtual cylinder plane

## 1.2 Overview of our Method

Before introducing our detection framework, it is necessary to clarify the relationship between drivable area detection and ground segmentation or road labelling. As we mentioned above, the drivable area has perhaps not been well defined in previous literature, and it is commonly defined as the geometric constraints of where the vehicle can traverse [27]. In this paper, we first clarify the difference and connection among the drivable area, the road area and the ground. Let $A_g \subset \mathbb{R}^2$ refer to the ground and $A_r \subset \mathbb{R}^2$ refer to the road area around a vehicle, and let the drivable area be represented by $A_d \subset \mathbb{R}^2$, so the restriction exists that $A_d \subseteq A_r \subset A_g$. In most time, the drivable area is approximately equal to the road area except for some exceptional cases where there are obstacles on the road. Consequently, we define the reachable area $R_r \subset \mathbb{R}^2$ as a region with the obstacle and geometric constraints, so there is another restriction that $A_d = A_r \cap R_r$, as shown in Fig. 2. Note that the ground area includes the obstacle area, but the ground points are strictly distinguished from obstacle points in the point cloud.

Considering the limitation of the existing methods, we propose a double projection based method. For each scan, the point cloud is projected onto two different planes simultaneously. As a result, we get an elevation map and a range map with the projection planes being the ground and a virtual cylinder plane, respectively. Ground estimation and obstacle detection are performed directly on the elevation map. After that, the reachable area will be generated from the obstacle map, and the ground model built by ground estimation will be regarded as a reference for ground segmentation. Subsequently, road points will be filtered out from ground points via extracting features on the range map. Finally, the drivable area will be obtained from a pixel-level Bayesian fusion method. The complete process is illustrated in Fig. 3, and the main contributions of our work are: (1) It clarifies the definition of drivable area and proposes a systematical framework for drivable area detection including ground segmentation and road labelling; (2) It uses new morphological operations to solve

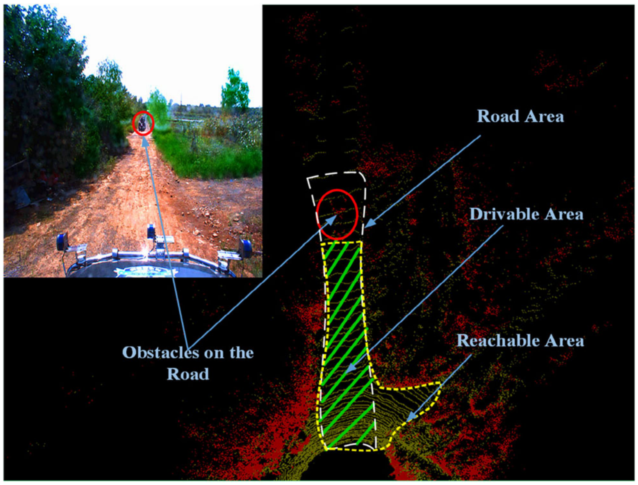


**Fig. 2** Schematic of the restriction: $A_d = A_r \cap R_r$. The road area and reachable area are indicated with white and yellow dashed lines, respectively. And the current drivable area is represented by green lines. Moreover, red points represent obstacles, and the ground points are shown in yellow

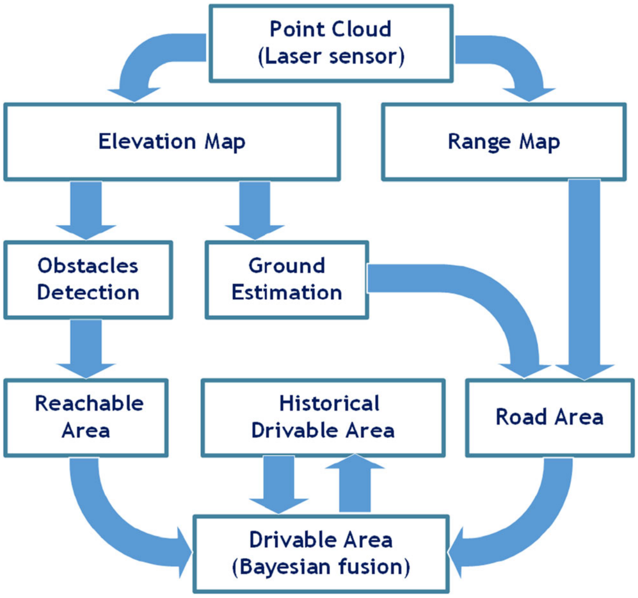


**Fig. 3** Framework of the drivable area detection based on point cloud double projection

the challenge of ground segmentation. And for road labelling, it develops a range map based method by incorporating the ground model and richer features; (3) It introduces a two-step reachable area search method including a fast obstacle closing algorithm and a forward flood fill algorithm; (4) To improve the performance of drivable area detection, it proposes a Bayesian fusion method whose input includes reachable area, road area, and previous drivable area after the ego-motion compensation.

The rest of this paper is organized as follows: Section 2 discusses the point cloud projection method. Section 3, 4, and 5 introduce our algorithm in detail. And some experiment results in real traffic scenes are presented in Section 6. Finally, Section 7 concludes this paper and gives an outlook in this field.

## 2 Point Cloud Projection

The raw 3D point cloud obtained from a laser scanner is a set of points representing distance measurements from the surface of objects. Our primary sensor is a Velodyne HDL64 S2, which has 64 laser beams and can provide a full 360° azimuthal field of view with a 0.17° of rotation angle resolution. Let define a point as follows:

$$p_{k|l} = \left(x_{k|l}, y_{k|l}, z_{k|l}\right)^T \qquad k = 1, 2, \cdots, K \tag{1}$$

where $k$ denotes the firing position whose total number per turn is $K$, and $l$ denotes the laser beam number from 0 to 63 (No.0 means the innermost beam). $x_{k\,|\,l}$, $y_{k\,|\,l}$, and $z_{k\,|\,l}$ are Cartesian coordinates in centimeters.

### 2.1 Elevation Map

The elevation map is usually generated by a grid-based method in which 3D measurements are projected onto a 2.5D occupancy grid. In our work, a lower-left corner coordinate system is adopted for the elevation map $E$ whose size is 500×750 with 20 cm resolution, and the vehicle is located at the grid cell $E_{(250,250)}$. For a point $p_{k\,|\,l}$, it will be mapped into $E_{(i,j)}$ if $p_{k\,|\,l}$ satisfies the following constraints:

$$\begin{cases} 20i \le x_{k|l} < 20(i+1) & 0 \le i < 500 \\ 20j \le y_{k|l} < 20(j+1) & 0 \le j < 750 \end{cases} \tag{2}$$

And $S_{(i,j)}$ is the set of points mapped into $E_{(i,j)}$. In this paper, a Max-Min elevation map is used to distinguish obstacle grid cells from other cells, so if the absolute vertical displacement in a grid cell is greater than 15 cm, this cell will be labelled as an obstacle grid cell. Finally, all obstacle grid cells form the obstacle map. For a non-obstacle grid cell $E_{\left(i^{'},j^{'}\right)}$, if $S_{\left(i^{'},j^{'}\right)} \neq \varnothing$, then this cell will be labelled as a ground cell, otherwise, it will be labelled as an unknown grid cell.

### 2.2 Range Map

Most laser ranging scanners provide raw data with a timestamp and the orientation of beams. Therefore, a 2.5D range map can be naturally created from the incoming LiDAR measurements. In a range map, the total number of rows is equal to that of laser beams in the vertical direction, and the total number of columns is related to the horizontal angle resolution of the scanner. Consequentially, all points of one scan can be arranged within a range map $R$. If $R_{(m,\,n)}$, which represents the pixel value at the coordinate $(m,n)$ in $R$, corresponds to the point $p_{k\,|\,l}$, then

$$\begin{cases} m = l \\ n = f_{\text{mod}}(k + \delta_l, K) \\ R_{(m,n)} = \sqrt{x_{k|l}^2 + y_{k|l}^2 + z_{k|l}^2} \end{cases} \tag{3}$$

where $f_{\text{mod}}$ is a modulo operation, and $\delta_l$ is the zero deviation of rotation angle for laser beam $l$. Note that for invalid measurements, their pixel values are set to 0 in a range map.

## 3 Elevation Map Processing Strategy

Reachable area extraction and ground estimation are two main tasks related to the elevation map. To improve real-time performance, we avoid complex mathematical methods in our framework, and the fact that the elevation map essentially is a 2.5D image makes it possible to accomplish these tasks with morphological operations.

### 3.1 Reachable Area Extraction

Considering the traversability, we need to exclude the region obstructed by surrounding obstacles from the drivable area. Therefore, reachable area extraction is to find out where the vehicle can reach from its current position, and this process illustrated in Fig. 4 can be divided into three steps as follows:

Step 1: Generating a mask used to limit the searching area. The mask comes into being by drawing a line between each ground grid cell and the vehicle cell $E_{(250,250)}$. For connectivity, it needs to be further dilated and eroded, and the region beyond mask is identified as the unknown region.

Step 2: Obstacle clustering and closing. In order to speed up computations, the distance-based obstacle clustering and closing, as shown in Fig. 5, are performed on a down-sampled image with 250×375 pixels. In practice, we set the distance threshold $d_{\text{th}}$ as 1.6 m. For a point $p_n$, if $p_p$ is a point in the cluster $C_{obs}$ and $| p_n p_p | < d_{\text{th}}$, then $p_n$ also belongs to $C_{obs}$, and $p_p$ becomes the parent point of $p_n$. Finally, each obstacle cluster will be closed by drawing lines between its obstacle points and their parent points, see Appendix 1.

Step 3: Searching for the reachable area. Take $E_{(250,250)}$ as the start point, then search for non-obstacle areas under the mask. Since we focus on the area in front of vehicles, our search algorithm named forward flood fill is directional, see Appendix 2.

Due to limitations of the Max-Min elevation map, some special obstacles such as negative obstacles may be ignored. Therefore, the obstacle information in our obstacle map is not complete, resulting in that the range of reachable area is more extensive than the actual. But this issue will be addressed in the data fusion section. Moreover, reachable area extraction is performed in a down-sampled image, so we need to up-sample the search result.

### 3.2 Ground Estimation

Ground estimation is a basic pre-processing for ground segmentation, and its purpose is to establish a ground reference model or a digital terrain model $M_g(x, y)$. If a point$(x_t, y_t, z_t)$ satisfies the following condition:

$$|M_g(x_t, y_t) - z_t| < Z_{\text{th}} \tag{4}$$

where $Z_{\text{th}}$ is a vertical height threshold and is set to 15 cm in our work, then this point could be classified as a ground point. Digital terrain modelling methods based on morphological operations have been developed in some previous studies [28, 29], but they are mostly used to process the airborne LiDAR point cloud, which is significantly different from the point cloud generated by the vehicle equipment. In our work, online ground estimation is performed with the assistance of an elevation map. Therefore, our ground model is a 2D reference matrix from which the reference height of the ground in each grid cell can be inferred.

Firstly, the ground reference matrix $G$ will be initialized using all grid cells marked as ground:

$$G_{(i,j)} = \frac{255}{N_{(i,j)}H}\left(\sum_{L_{(i,j)}} z + 0.5N_{(\mathrm{i},j)}H\right) \tag{5}$$

where $L_{(i,j)} = \{z | [z - \min(z_{(i,j)})] < vh_{\text{th}}, z \in z_{(i,j)}\}$, $z_{(i,j)}$ is the set of the Z-coordinate of points in $S_{(i,j)}$, and $N_{(i,j)}$ is the size of $L_{(i,j)}$. The parameter $H$ named the normalization factor represents the limitation of ground height, and it is used to map the vertical height in $[-0.5H, 0.5H]$ to a pixel value in $(0,255]$. The threshold $vh_{\text{th}}$ has a value smaller than $Z_{\text{th}}$. In our work, the parameters $H$ and $vh_{\text{th}}$ are set to 5 m and 5 cm, respectively. For unknown grid cells, they will be set as 0, which means the current point cloud cannot provide information about these cells. After initialization, the ground reference matrix $G$ can be considered as a grey image. Therefore, the following image morphological operations are introduced for ground modelling:

(1) Image Dilation: Due to the existence of unknown grid cells, image dilation is equivalent to the reasoning on unknown cells, making the ground model satisfy the continuity constraint. Unlike traditional dilation algorithm, our dilation algorithm only changes the unknown cells in the image:

$$G^{\text{dst}}_{(i,j)} = \begin{cases} \min\limits_{(i',j')\in\mathbb{N}_+} G^{\text{src}}_{(i',j')} & G^{\text{src}}_{(i,j)} = 0 \\ G^{\text{src}}_{(i,j)} & G^{\text{src}}_{(i,j)} \neq 0 \end{cases} \tag{6}$$

where $\mathbb{N}_+$ is the set of coordinates with a positive pixel value in the 24-neighbourhood of $G_{(i,j)}$.

(2) Image Filtering: Since LiDAR has wrong measurements and Eq. (5) cannot deal with overhanging and sunken obstacles, the ground reference matrix $G$ after dilation needs to remove its abnormal value. And for this purpose, our filtering method only targets local extremum points, making the ground model satisfy the smoothness constraint. For each grid cell, if $G_{(i,j)}$ is the maximum or the minimum in its cross neighbourhood $\mathbb{C}$, $G(i,j)$ will be regarded as an abnormal value and its new value will be given by:

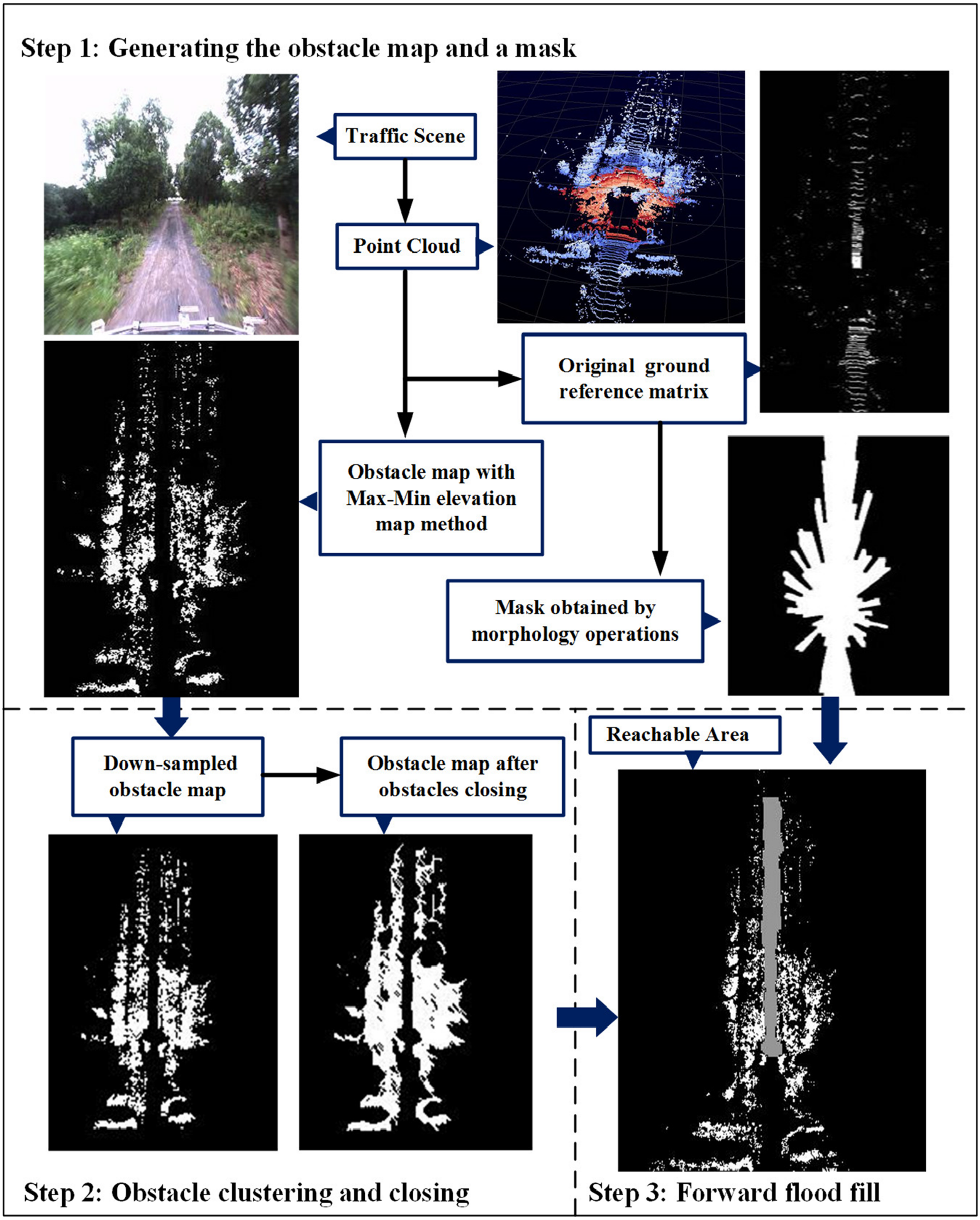

Step 1: Generating the obstacle map and a mask
Traffic Scene
Point Cloud
Original ground reference matrix
Obstacle map with Max-Min elevation map method
Mask obtained by morphology operations
Down-sampled obstacle map
Obstacle map after obstacles closing
Reachable Area
Step 2: Obstacle clustering and closing
Step 3: Forward flood fill

**Fig. 4** Process of reachable area extraction: (1) Generating the obstacle map and a mask; (2) Obstacle clustering and closing; (3) Searching for the reachable area with forward flood fill method

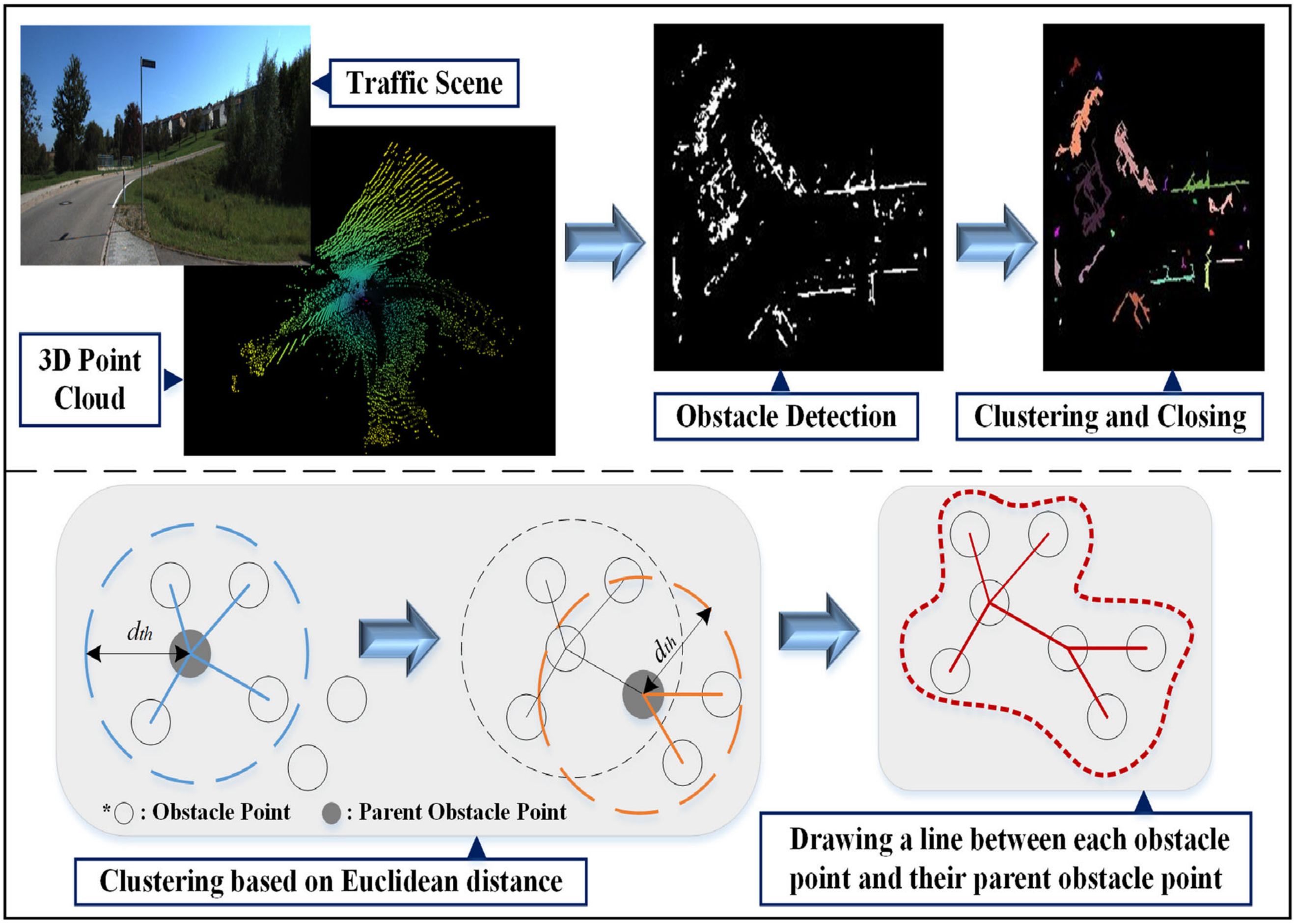

Traffic Scene
3D Point Cloud
Obstacle Detection
Clustering and Closing
dth
dth
* ○ : Obstacle Point
● : Parent Obstacle Point
Clustering based on Euclidean distance
Drawing a line between each obstacle point and their parent obstacle point

**Fig. 5** Sketch of obstacle clustering and closing. In practice, the parameter $\boldsymbol{d}_{\mathbf{th}}$ is slightly smaller than the width of a car to pass through narrow rural roads

$$
\begin{aligned}
v_{\mathrm{ave}} &= \frac{1}{n}\sum_{(i',j')\in\mathbb{C}} G^{\mathrm{src}}_{(i',j')} \\
v_{\mathrm{sum}} &= \sum_{(i',j')\in\mathbb{C}} \left(|G^{\mathrm{src}}_{(i',j')} - v_{\mathrm{ave}}| + \xi\right)^{-2} \\
G^{\mathrm{dst}}_{(i,j)} &= \sum_{(i',j')\in\mathbb{C}} \frac{G^{\mathrm{src}}_{(i',j')}}{\left(|G^{\mathrm{src}}_{(i',j')} - v_{\mathrm{ave}}| + \xi\right)^{2} v_{\mathrm{sum}}}
\end{aligned}
\tag{7}
$$

where $\xi$ is a tiny positive number, $\mathbb{C} = \{(i, j + c), (i + c, j) | c = \pm 3, \pm 7\}$, and $n$ $(n = 8)$ means the total number of elements in $\mathbb{C}$.

After the above morphological operations, the ground model $G$ has been constructed in the form of a reference matrix, and its pixel value can be mapped to the actual ground height through Eq. (5). Therefore, we can efficiently implement ground segmentation with the help of model $G$ in real-time, marking each point in 3D point clouds as a ground point or an obstacle point. Some segmentation results in different scenarios are illustrated in Fig. 6.

## 4 Road Labelling Based on Range Map

Given the mapping relationship established by Eqs. (2) and (3), the result of ground segmentation can be utilized to find all ground points on the range map. After that, we need to filter the road points from the ground points. Unlike the existing works [25, 30], the geometric relationship between adjacent laser beams is not critical in our method, on the contrary, the smoothness of adjacent points in the same beam is designed as a filter. For a road point $p_{v\,|\,e}$, it should first meet the following conditions:

(1) Referring to the form of Eq.(4), the ground model constraint condition is given as follows:

$$
|\frac{G_{(i'',j'')}H}{255} - 0.5H - z_{v|e}| < Z_{\mathrm{th}} \tag{8}
$$

where $G_{(i'',j'')}$ is the associated grid cell of point $p_{v\,|\,e}$. Moreover, all the points in the 4-neighbourhood corresponding to $p_{v\,|\,e}$ on the range map, namely $\{p_{v-1\,|\,e}, p_{v+1\,|\,e}, p_{v\,|\,e+1}, p_{v\,|\,e-1}\}$, also need to satisfy the ground model constructed by ground estimation.

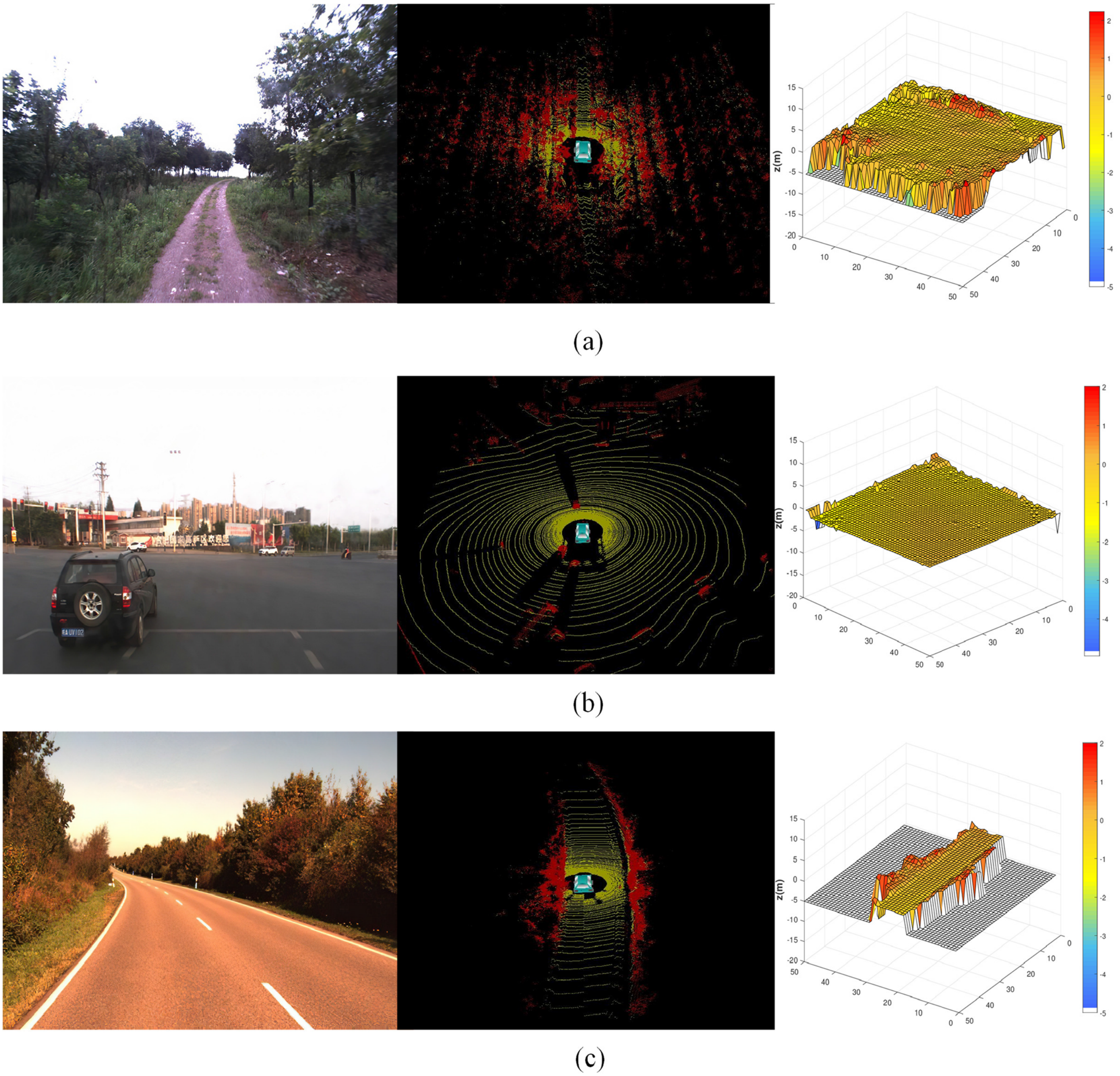


**Fig. 6** Results of ground estimation in different scenarios. Left column: camera image. Middle column: the result of point cloud segmentation. Red points represent obstacles, and the ground points are shown in yellow. Right column: ground model. To facilitate illustration, we select a 500×500 area on the elevation map for display and adjust the grid resolution to 2 m. For unknown grid cells, their height is set to -5 m

(2) Neighbourhood smoothness constraint condition. This condition consists of the distance smoothness $s_d < S_{d\text{th}}$ and the angle smoothness $s_a < S_{a\text{th}}$. The former is related to the variance of the distance between adjacent points, and it needs to be calculated separately on the left and right sides of $p_{v\,|\,e}$. For the latter, it can be represented by the mean of the angle between two vectors that start from $p_{v\,|\,e}$ and end on different sides of $p_{v\,|\,e}$.

The detailed definitions of $s_d$ and $s_a$ are given by:

$$\begin{aligned}
s_d^l &= \frac{1}{n_l-1}\sum_{k=1}^{n_l-1}\left(|L_kL_{k+1}|-\frac{1}{n_l-1}\sum_{k=1}^{n_l-1}|L_kL_{k+1}|\right)^2\\
s_d^r &= \frac{1}{n_r-1}\sum_{k=1}^{n_r-1}\left(|R_kR_{k+1}|-\frac{1}{n_r-1}\sum_{k=1}^{n_r-1}|R_kR_{k+1}|\right)^2\\
s_d &= \min\{s_d^l, s_d^r\} \qquad n_s = \min\{n_l, n_r\}\\
s_a &= \pi-\frac{1}{n_s}\sum_{n=1}^{n_s} f_{\text{acos}}\left(\frac{\overrightarrow{p_{v|e}L_n}\cdot\overrightarrow{p_{v|e}R_n}}{|\overrightarrow{p_{v|e}L_n}\|\overrightarrow{p_{v|e}R_n}|}\right)
\end{aligned} \tag{9}$$

where $L$ and $R$ are the set of ground points selected from $\{p_{v-10 \mid e},\ldots,p_{v-1 \mid e}\}$ and $\{p_{v+1 \mid e},\ldots,p_{v+10 \mid e}\}$. And $n_l$, $n_r$ are the total number of points in $L$ and $R$, respectively. The operation $f_{\text{acos}}$ is an inverse cosine function. There are two thresholds for neighbourhood smoothness: $S_{a\text{th}}$ and $S_{d\text{th}}$, and the former is set to 0.75 in our experiments, however, for the latter, different beams correspond to different values, here we recommend setting them to 1.5–2 times the distance between two adjacent measurements on the flat ground.

However, in most cases, the points only satisfy the above conditions may contain non-road targets due to the existence of isolated areas, as shown in Fig. 7(b). An obvious fact is that laser scan lines on the road surface are discrete on the elevation map, but they are continuous on the range map. Therefore, we extract all connected components on the range map and take connected components with a large area as road areas, see Fig. 7(c). Then, all road points are mapped into the vehicle Cartesian coordinate and connected end to end to generate the road area $A_r$, see Fig. 7(e).

## 5 Drivable Area Detection Based on Bayesian Fusion

As mentioned above, the robustness and accuracy of drivable area detection are pivotal but challenging, especially on rough roads. We address these challenges through a Bayesian fusion framework, whose input includes the previous drivable area after an ego-motion compensation ($M_1$), the current road area($M_2$), the reachable area ($M_3$) and the intersection area of the road area and the reachable area ($M_4$), as shown in Fig. 8. Moreover, all input matrices have been binarized to 0 or 1, and the candidate region $M$ is the union of the above areas, $M = M_1 \cup M_2 \cup M_3$.

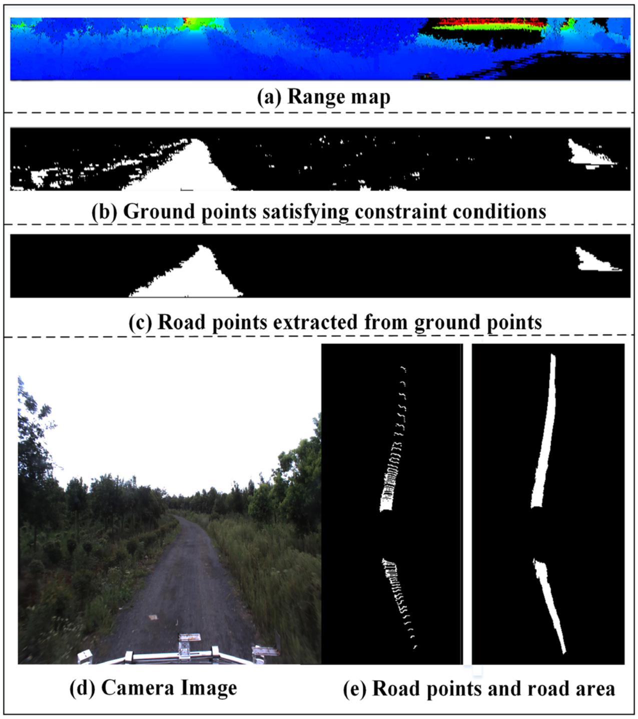

(a) Range map
(b) Ground points satisfying constraint conditions
(c) Road points extracted from ground points
(d) Camera Image
(e) Road points and road area

Fig. 7 Process of road area detection. Since there is a blind area around the vehicle, a circular road area will be artificially added around the vehicle

The Bayesian theory has been widely applied in the field of the drivable area or traversability [31–33]. For the Bayesian approach to unsupervised classification, the goal is to find the most probable class description given the data and prior expectations. In our work, each point in $M$ has two possible states, belonging to the drivable area ($\psi_1$) or not belonging to the drivable area ($\psi_2$). Therefore, there are two elements in the class set $\Psi = (\psi_1, \psi_2)$. The matrices $M_i, i \in I$ and $I = \{1, 2, 3, 4\}$, will be regarded as classifiers, and let's denote $x_i$, $x_i \in \Psi$, as the classification result of the classifier $M_i$ for a point $p(x, y)$ in $M$:

$$x_i(p) = \begin{cases} \psi_1 & M_i(x,y) = 1 \\ \psi_2 & M_i(x,y) = 0 \end{cases} \tag{10}$$

The class prior probability is represented as $P(\psi)$, class conditional probabilities are denoted by $P(x_i| \psi)$, and $P(x_1, x_2, x_3| \psi)$ is the joint probability. According to the Bayesian minimum error rate theory [34], if the class of a point in $M$ is $\psi_c$, then $\psi_c$ satisfies the following condition:

$$\psi_c = \arg\max_{\psi \in \Psi} f(\psi, x_1, x_2, x_3, x_4) \tag{11}$$

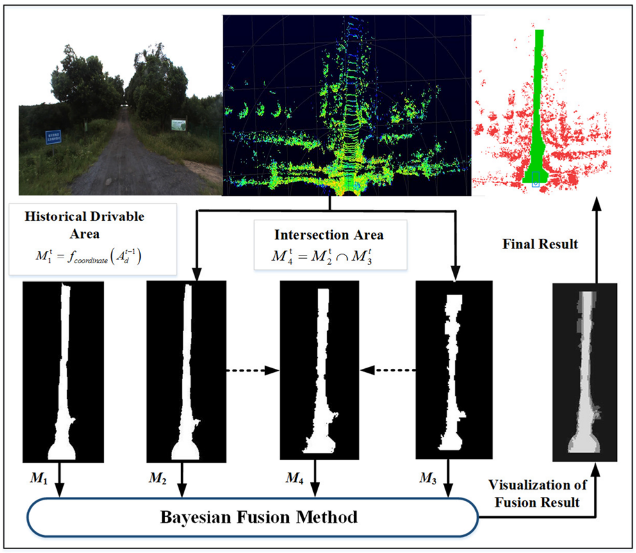

Historical Drivable Area
$M_1^t = f_{coordinate}\left(A_d^{t-1}\right)$
Intersection Area
$M_4^t = M_2^t \cap M_3^t$
Final Result
$M_1$
$M_2$
$M_4$
$M_3$
Visualization of Fusion Result
Bayesian Fusion Method

Fig. 8 Illustration of the Bayesian fusion method for drivable area detection. The current road area based on a range map $M_2$ and the reachable area under obstacle constraints $M_3$ are extracted from point clouds in real-time. $M_1$ is the historical drivable area after the coordinate transform $f_{\text{coordinate}}$, and $M_4$ is the intersection of $M_2$ and $M_3$

Since $M_4$ is directly related to $M_2$ and $M_3$, they are not conditional independent, so we design the target function as follows:

$$f(\psi, x_1, x_2, x_3, x_4) = P(\psi|x_1, x_2, x_3) + \lambda P(\psi|x_4) \tag{12}$$

where $\lambda$ is a value greater than 1, which means that from experience, $M_4$, as the intersection area of $M_2$ and $M_3$, is more reliable. And in our experiments, we set $\lambda$ as 1.2.

The probability $P(\psi| x_1, x_2, x_3)$ can be estimated using Bayesian rule:

$$P(\psi|x_1, x_2, x_3) = \frac{P(x_1, x_2, x_3|\psi)P(\psi)}{P(x_1, x_2, x_3)} \tag{13}$$

Assuming that the classifiers $M_i$, $i \in 1, 2, 3$ are approximately independent of each other, thus we have:

$$P(x_1, x_2, x_3|\psi) = \prod_{i=1,2,3} P(x_i|\psi) \tag{14}$$

$$P(x_1, x_2, x_3) = \prod_{i=1,2,3} P(x_i) \tag{15}$$

Combining Eqs. (13) (14) and (15), we can obtain that:

$$P(\psi|x_1, x_2, x_3) = \frac{P(\psi) \prod_{i=1,2,3} P(x_i|\psi)}{\prod_{i=1,2,3} P(x_i)} \tag{16}$$

After converting the class conditional probabilities to the posterior probabilities, we have

$$P(\psi) \prod_{i=1,2,3} P(x_i|\psi) = P^{-2}(\psi) \prod_{i=1,2,3} P(x_i) \prod_{i=1,2,3} P(\psi|x_i) \tag{17}$$

Combining Eqs. (16) and (17), Eq.(11) can be rewritten as:

$$\psi_c = \arg\max_{\psi \in \Psi} \left[ P^{-2}(\psi) \prod_{i=1,2,3} P(\mathrm{\psi}|x_i) + \lambda P(\psi|x_4) \right] \tag{18}$$

Furthermore, assuming that there is only a slight deviation between the posterior probability and the corresponding prior probability [35]:

$$P\left(\psi_j|x_i\right) = P\left(\psi_j\right)\left(1 + \delta_{ij}\right) \; i = 1, 2, 3 \;\; j = 1, 2 \tag{19}$$

where $\delta_{ij} < < 1$. After some mathematical manipulations, we have

$$P^{-2}\left(\psi_j\right) \prod_{i=1,2,3} P\left(\psi_j|x_i\right) = P\left(\psi_j\right) \prod_{i=1,2,3} \left(1 + \delta_{ij}\right) \tag{20}$$

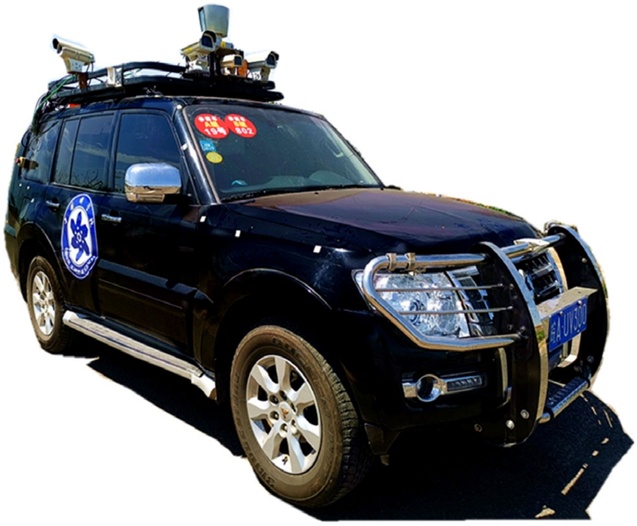


Fig. 9 Our intelligent vehicle used for the experiments. It is equipped with five cameras, a Velodyne HDL-64E S2 LiDAR, and a GPS-aided inertial navigation system

Expand the product on the right of Eq.(20) and ignore higher-order items, then:

$$P\left(\psi_j\right) \prod_{i=1,2,3} \left(1 + \delta_{ij}\right) = P\left(\psi_j\right) + \sum_{i=1,2,3} P\left(\psi_j\right)\delta_{ij} \tag{21}$$

Combining Eqs. (18), (19), and (21), Eq. (18) can be rewritten as:

$$\psi_c = \arg\max_{\psi \in \Psi} \left[ \sum_{i=1,2,3} P(\psi|x_i) + \lambda P(\psi|x_4) - 2P(\psi) \right] \tag{22}$$

In general, the map of an unknown environment is not available, so the prior knowledge is blank in the system. As a result, we assume the prior probabilities of all classes are equal, and then Eq. (22) can be further simplified as follows:

$$\psi_c = \arg\max_{\psi \in \Psi} \left[ \sum_{i=1,2,3} P(\psi|x_i) + \lambda P(\psi|x_4) \right] \tag{23}$$

From the above derivation result, it is evident that posterior probabilities play a decisive role in our Bayesian fusion method. Considering the variability of environments, we estimate the posterior probabilities according to the fusion result in real-time. And this process can be divided into four steps:

Step 1: Calculating the probabilities of each state for all position $p(x, y)$ at time t:

**Table 1** Statistical results of Accuracy on SemKitti dataset (%)

| Approach | Seq00 | Seq01 | Seq02 | Seq03 | Seq04 | Seq05 | Seq06 | Seq07 | Seq08 | Seq09 | Seq10 | Mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MethodGA | 87.44 | 85.32 | 88.79 | 90.40 | 93.89 | 86.06 | 86.81 | 86.74 | 86.61 | 88.00 | 86.11 | 87.83 |
| MethodPF | 97.21 | 67.49 | 90.84 | 92.85 | 97.37 | 96.45 | 95.23 | 96.69 | 96.78 | 89.10 | 89.74 | 91.79 |
| MethodOG | 93.30 | 94.08 | 94.86 | 96.34 | 97.31 | 94.54 | 93.53 | 93.65 | 93.17 | 95.06 | 94.64 | 94.59 |
| MethodVH | 91.15 | 91.98 | 89.96 | 91.48 | 85.64 | 89.72 | 83.89 | 90.82 | 90.87 | 88.47 | 86.28 | 89.11 |
| Ours | 96.94 | 98.65 | 96.90 | 97.32 | 99.03 | 96.25 | 97.02 | 96.42 | 97.33 | 97.11 | 95.86 | 97.16 |

$$p_1 = \sum_{i=1,2,3} P^t(\psi_1|x_i(p)) + \lambda P^t(\psi_1|x_4(p)) \tag{24}$$

$$p_2 = \sum_{i=1,2,3} P^t(\psi_2|x_i(p)) + \lambda P^t(\psi_2|x_4(p))$$

Step 2: Making Bayesian decision:

$$A_d(x,y) = \begin{cases} 1 & p_1 > p_2 \\ 0 & \text{other.} \end{cases} \tag{25}$$

Step 3: Estimating the posterior probabilities:

$$\begin{aligned} P_{\text{temp}}(\psi_1|x_i = \psi_j) &= \frac{\sum\limits_{M_i=\psi_j} M}{\sum A_d} \\ P_{\text{temp}}(\psi_2|x_i = \psi_j) &= \frac{\sum\limits_{M_i=\psi_j} M}{\sum M - \sum A_d + 1} \end{aligned} \tag{26}$$

$$\mathrm{P}^{t+1}(\psi_1|x_i = \psi_j) = f_{\text{nor}}\left(\frac{P_{\text{temp}}(\psi_1|x_i = \psi_j)}{P_{\text{temp}}(\psi_1|x_i = \psi_j) + P_{\text{temp}}(\psi_2|x_i = \psi_j)}\right) \tag{27}$$

$$\mathrm{P}^{t+1}(\psi_2|x_i = \psi_j) = f_{\text{nor}}\left(\frac{P_{\text{temp}}(\psi_2|x_i = \psi_j)}{P_{\text{temp}}(\psi_1|x_i = \psi_j) + P_{\text{temp}}(\psi_2|x_i = \psi_j)}\right) \tag{28}$$

where $i = 1,2,3,4$, and $j = 1,2$. The symbol $\sum A_d$ means the total number of drivable area points in $M$, and $\sum\limits_{M_i=\psi_j} M$ indicates the total number of points in $M$ that meet the constraints $x_i(p) = psi_j$. And the normalization function $f_{\text{nor}}$ is given by:

$$f_{\text{nor}}(r) = \begin{cases} 0.9 & r > 0.9 \\ r & r \in [0.1, 0.9] \\ 0.1 & r < 0.1 \end{cases} \tag{29}$$

Step 4: Updating $t + 1$ posterior probabilities with smooth filtering:

$$P^{t+1}_{\text{new}}(\psi|x) = 0.25 \sum P^T(\psi|x) \qquad T = t-2, \ldots, t+1 \tag{30}$$

## 6 Experiments and Results

For the evaluation, our experimental platform, a sports utility vehicle Pajero, is modified as shown in Fig. 9. It has been equipped with five cameras, a Velodyne HDL64 on the top covering a 360° field of view, and a GPS-aided inertial navigation system that provides vehicle velocity and orientation. To fully demonstrate the performance of our method, our experimental evaluation consists of two parts: the evaluation of ground segmentation and the evaluation of drivable area detection. About the former, we quantitatively test it on the SemKitti dataset [36] and compare our method with some previous methods. Besides, we also make qualitative comparisons on our off-road data. For the latter, we first calculate the time of each step in our method to evaluate real-time

**Table 2** Statistical results of Recall on SemKitti dataset (%)

| Approach | Seq00 | Seq01 | Seq02 | Seq03 | Seq04 | Seq05 | Seq06 | Seq07 | Seq08 | Seq09 | Seq10 | Mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MethodGA | 92.77 | 94.20 | 91.00 | 92.83 | 95.29 | 92.08 | 91.20 | 92.80 | 92.33 | 92.28 | 91.34 | 92.56 |
| MethodPF | 97.54 | 59.72 | 89.14 | 92.23 | 99.67 | 97.69 | 95.68 | 96.80 | 96.56 | 85.61 | 84.92 | 90.51 |
| MethodOG | 93.53 | 97.82 | 94.83 | 96.83 | 97.83 | 94.60 | 94.68 | 93.38 | 94.31 | 96.20 | 94.39 | 95.31 |
| MethodVH | 91.62 | 92.61 | 90.78 | 92.59 | 86.90 | 91.47 | 79.72 | 92.02 | 89.79 | 87.76 | 83.85 | 89.01 |
| Ours | 99.51 | 99.70 | 99.17 | 99.53 | 99.65 | 99.38 | 97.91 | 99.39 | 99.35 | 99.41 | 99.09 | 99.28 |

**Table 3** Statistical results of Precision on SemKitti dataset (%)

| Approach | Seq00 | Seq01 | Seq02 | Seq03 | Seq04 | Seq05 | Seq06 | Seq07 | Seq08 | Seq09 | Seq10 | Mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MethodGA | 83.55 | 86.97 | 92.92 | 94.13 | 96.64 | 82.58 | 87.54 | 80.71 | 85.78 | 89.58 | 82.55 | 87.54 |
| MethodPF | 96.83 | 94.01 | 97.63 | 98.00 | 96.97 | 95.44 | 96.41 | 95.90 | 97.89 | 97.29 | 93.84 | 96.38 |
| MethodOG | 92.92 | 94.36 | 97.80 | 98.20 | 98.64 | 94.68 | 94.64 | 92.60 | 93.99 | 96.20 | 94.80 | 95.35 |
| MethodVH | 90.54 | 96.37 | 94.72 | 95.78 | 93.80 | 88.72 | 92.63 | 88.16 | 94.28 | 94.11 | 87.97 | 92.46 |
| Ours | 94.56 | 98.49 | 96.52 | 96.93 | 99.08 | 93.65 | 97.20 | 93.14 | 96.18 | 96.26 | 93.02 | 95.91 |

performance. However, since there is no suitable dataset for quantitative evaluation, the qualitative results of drivable area detection are shown in the end.

## 6.1 Ground Segmentation

At first, we test our method on the SemKitti dataset based on the odometry dataset of the KITTI Vision Benchmark [37]. The SemKitti dataset provides labels for sequential point clouds including ground, structure, vehicle, human and so on. Because ground segmentation is a binary classification problem for point clouds, we take ground-related points as the positive class, and the points belonging to other classes, such as vehicle, human, object, and structure, are labelled as the negative class in the test. Note that since the nature class includes both ground vegetation and non-ground vegetation like shrubs, this class is excluded in comparison experiments.

We adopt four previous ground segmentation methods as comparison tests. For a point $p_{v\mid e}$, its vertical height may be the simplest feature for ground segmentation, and this method (denoted as MethodVH) is generally used as a simple preprocessing step. In this method, we set the ground condition as $|z_{v\mid e}|$<30 cm. Paper [13] presents a deterministic iterative multiple plane fitting technique (denoted as MethodPF), and in our contrast test, some critical parameters of this method are set as follows: the number of iterations (3), the number of points used to estimate the lowest point representative (20), the threshold for points to be considered initial seeds (120), and threshold distance from the plane (15 cm). For Thrun's method [23] (denoted as MethodOG), all the points in an occupied grid cell whose vertical distance is less than 15 cm are regarded as ground points. Moreover, the geometric angle between the line $p_{v\mid e}p_{v\mid e+1}$ and the horizontal plane serves as a decisive segmentation condition in paper [25] (denoted as MethodGA), and we set the angle threshold to $30^{\circ}$ in this comparison test.

The experiment for ground segmentation produces four outcomes: true positive, true negative, false positive, and false negative, therefore, we adopt accuracy (ACC), recall (TPR), and precision (PPV) as evaluation metrics, and experiment results are tabulated in Tables 1, 2, 3, respectively. The results show that our method possesses

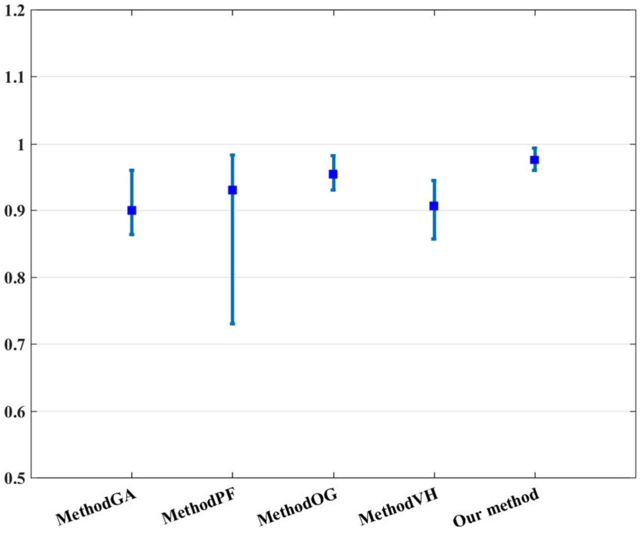


**Fig. 10** Results of $F_{\text{score}}$ average and its range on all sequences

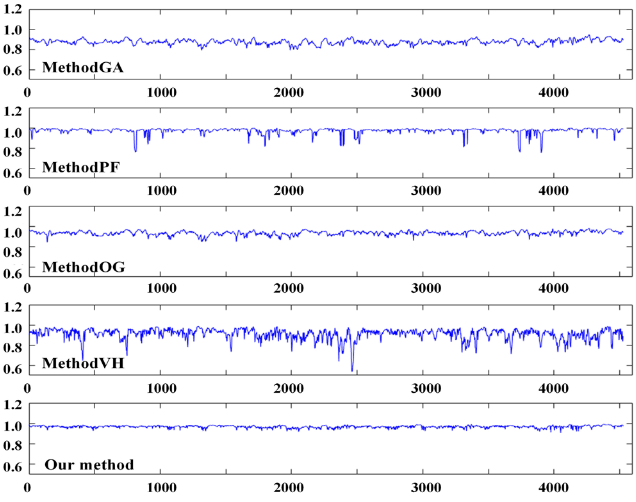


**Fig. 11** Accuracy of our ground segmentation method compared to that of other methods on 4540 continuous scans

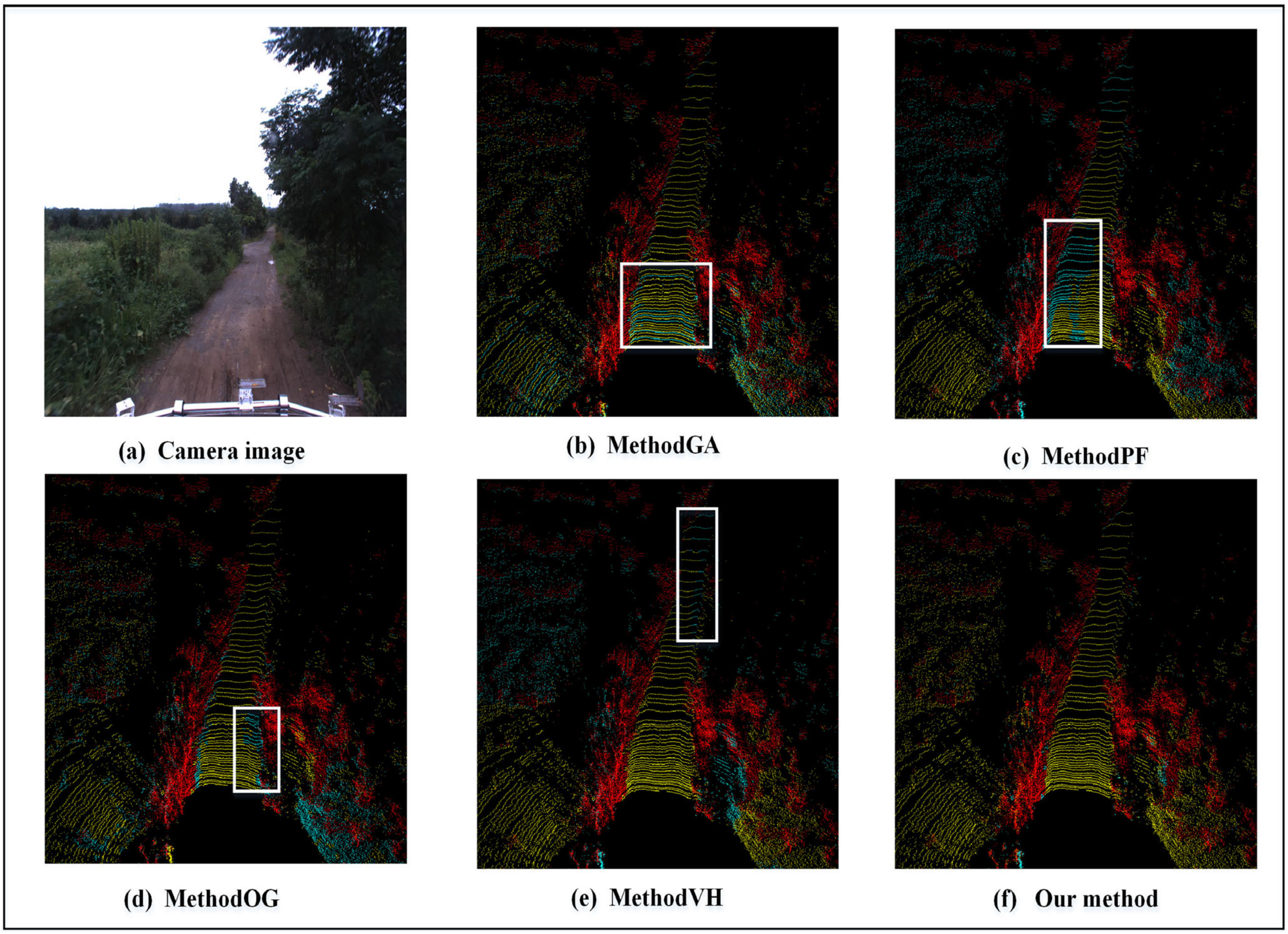


**Fig. 12** Results of ground segmentation with different methods on a rough road. Red and yellow points indicate obstacles and the ground, respectively. And flaws are marked by white boxes in which the blue points should be labelled as ground points

competitive performance metrics to the relevant methods in structured environments. Although the precision of our method is not the best, its accuracy is the highest. Since the road point is a subset of the ground point in our framework, the high accuracy can avoid the loss of road points caused by misclassification of ground points, which is determinant for the drivable area detection, especially on rough roads.

Besides, $F_{\text{score}}$ is a comprehensive indicator of performance in the statistical analysis of binary classification. And $F_{\text{score}}$ is defined as:

$$F_{\text{score}} = \frac{2PPV \times TPR}{PPV + TPR} \tag{31}$$

The average of $F_{\text{score}}$ and its Max-Min range on all sequences are shown in Fig. 10. And Fig. 11 demonstrates the accuracy of different methods on 4540 continuous scans. As can be seen from these figures, the comprehensive performance of our segmentation result is pretty well, and the accuracy of our method has a small fluctuation in different traffic scenes, indicating that it is better adapted to complex roads. Moreover, we also test these methods on our off-road scenarios since there is no dedicated off-road dataset. The test results are qualitatively shown in Fig. 12, and it is obvious that our method will not cause false-negative points on a massive scale when the vehicle posture changes dramatically. Consequently, the proposed method has optimal robustness compared to other methods, and it can better handle challenges on unstructured and rough roads.

**Table 4** Average computation times (**ms**)

| Obstacle detection | Reachable area | Ground segmentation | Road labelling | Data fusion | Total time |
|---|---|---|---|---|---|
| 12.98 | 5.19 | 32.49 | 19.85 | 6.35 | 76.86 |

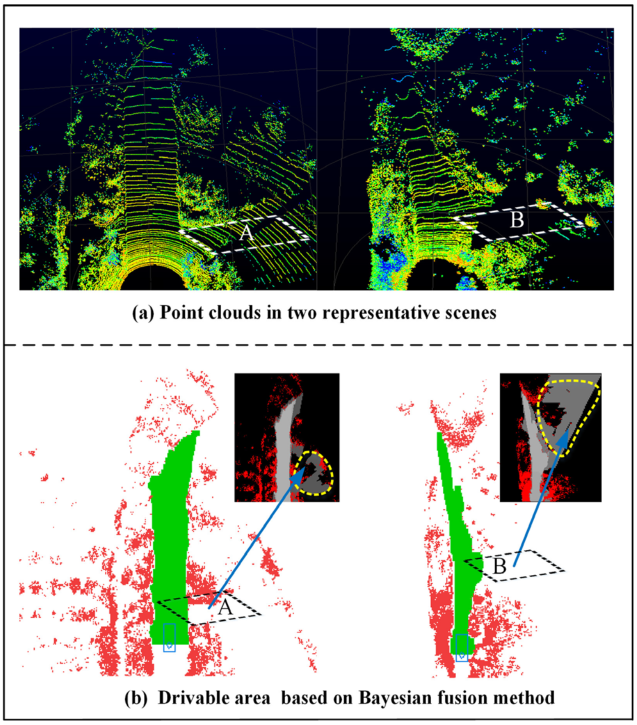

A
B
(a) Point clouds in two representative scenes
A
B
(b) Drivable area based on Bayesian fusion method

**Fig. 13** Results of drivable area detection based on our method and Bayesian fusion maps in two challenging situations

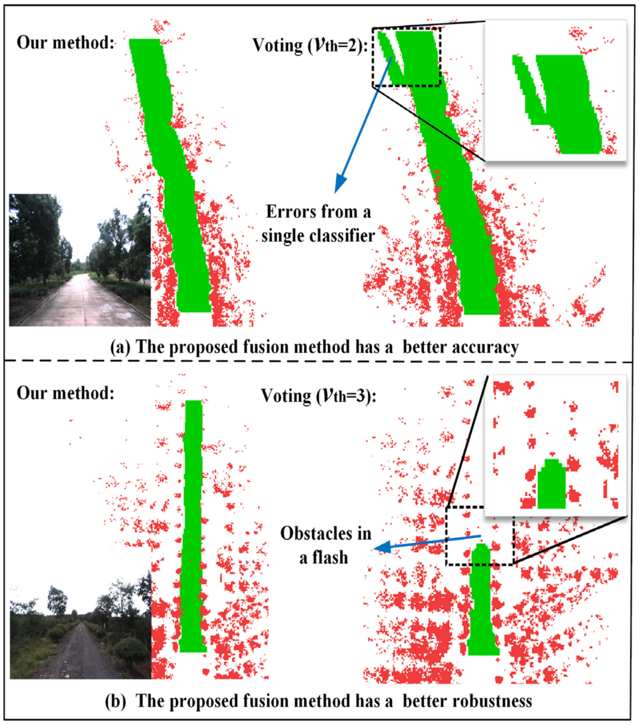

Our method:
Voting ($\nu_{th}$=2):
Errors from a single classifier
(a) The proposed fusion method has a better accuracy
Our method:
Voting ($\nu_{th}$=3):
Obstacles in a flash
(b) The proposed fusion method has a better robustness

**Fig. 14** Comparison of our method to voting with a different voting threshold $\nu_{th}$

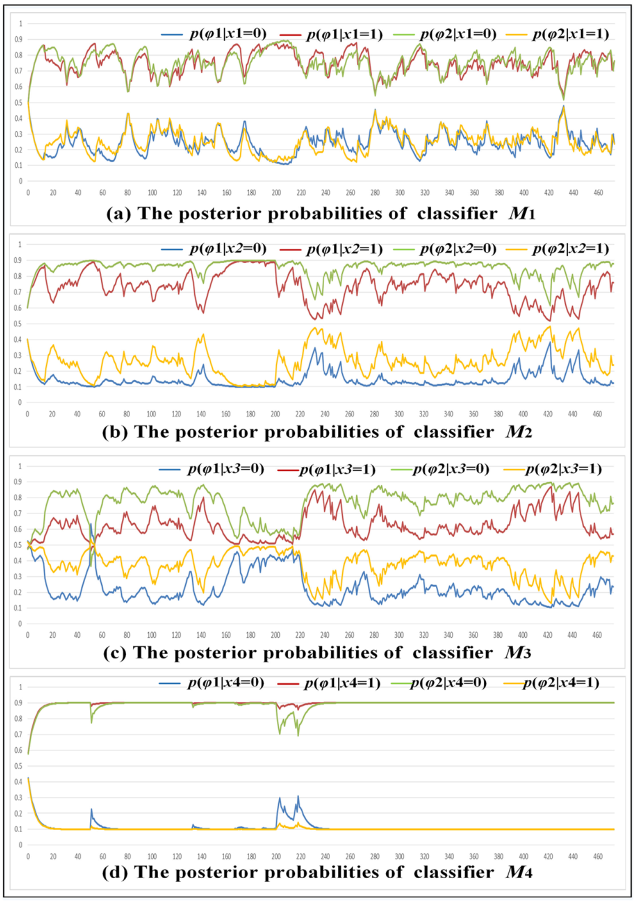


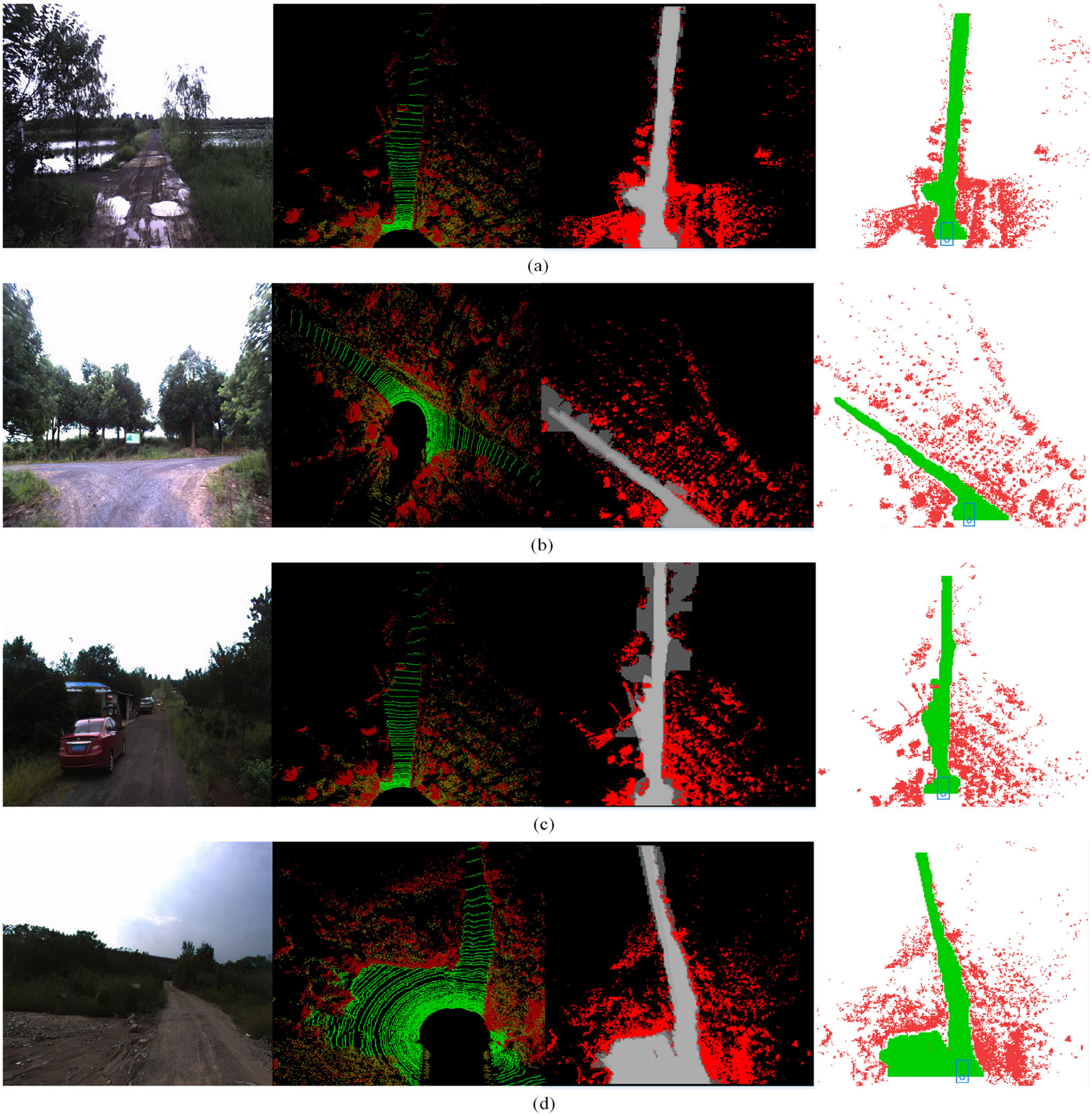
**Fig. 15** Trend of posterior probabilities for each classifier on rough roads

## 6.2 Drivable Area Detection

Without an accurate perception of the environment, autonomous vehicles could not make safe decisions to achieve reliable navigation. In addition, the real-time capability is also crucial for practical applications. Therefore, experimental evaluations of drivable area detection are intended to demonstrate the practicability of our method, including real-time performance and detection accuracy.

### 6.2.1 Runtime Evaluation

We implement the proposed method in C++, and the real-time performance is evaluated on a desktop computer with an Intel i7–7777 3.6GHz CPU and 16 GB memory. The average time for each part is shown in Table 4. Since the rotation frequency of LiDAR is generally 10HZ, our method can support online processing in real-time with the average computation time for each frame being 76.86 ms.


(a)

(b)

(c)

(d)

**Fig. 16** Qualitative results of our method on challenging rough roads. The columns from left to right are camera image, point cloud segmentation result, Bayesian fusion map, and drivable area with obstacles

Besides, the real-time capability can be further improved through the parallelization of data processing.

### 6.2.2 Accuracy Evaluation

Since the off-road environment is complex and diverse, it is difficult to characterize the drivable area with a single feature accurately. In Fig. 13(a), the point cloud in region A has the features of a road surface. However, our vehicle cannot reach there due to the existence of obstacles, so region A belongs to the road area but not the reachable area. In contrast, if only the connectivity on the obstacle map is considered, region B belongs to the reachable area, but it cannot be labelled as the road area since no point cloud exists in region B. For the above situations, only the reachable area or road area is not sufficient to define the distribution of the drivable area. Therefore, we introduce the Bayesian fusion method to handle these complex situations, and we also provide an ablation study to show the advantages of our method.

As shown in Fig. 13(b), there are apparent errors from a single classifier in the Bayesian fusion map, but the proposed method can better solve these challenges and provides reliable drivable areas. In addition, since voting is the most common method for classification problems with multiple classifiers, we compare our approach to it in terms of drivable area detection as an ablation study. The input of the voting method includes the previous drivable area after an ego-motion compensation, the road area and the reachable area, namely $M_1 - M_3$. First, all input matrices are summed up, and then a voting threshold $v_{\text{th}}$ is chosen to binarize the sum matrix. The final detection result is given by:

$$A_d = f_{\text{binarize}}\left(\sum_{i=1,2,3} M_i, v_{\text{th}}\right) \tag{32}$$

where $f_{\text{binarize}}$ is a binarization function, and the voting threshold $v_{\text{th}}$ means that if the number of classifiers supporting a point is not less than $v_{\text{th}}$, this point will be classified into the drivable area. Although voting methods with different thresholds $v_{\text{th}}$ have different effects, our method is better than them in terms of accuracy and robustness, and the results of the ablation study are illustrated in Fig. 14.

As shown in Fig. 14(a), the voting method with $v_{\text{th}}$ = 2 produces some problematic drivable areas, which indicates that it is less effective in suppressing the errors from a single classifier. However, if we set $v_{\text{th}}$ as 3, the obstacles in a flash (for example, a bird in Fig. 14(b)) will cause a sudden change in the drivable area, so the voting method with $v_{\text{th}}$ =3 is too sensitive in abnormal situations to guarantee the robustness of drivable area detection. On the contrary, in this case, the historical information ($M_1$) in our framework works through the Bayesian decision method, increasing the robustness of the detection system.

Figure 15 depicts the trend of posterior probabilities for each classifier on rough roads. Due to the complexity of environments and the uncertainty of each classifier, posterior probabilities are constantly changing over time, and it is continuously revised posterior probabilities that ensure the accuracy of Eq. (23). In addition, as shown in Fig. 15(d), the posterior probabilities of $M_4$ change slightly compared with those of other classifiers, indicating that the intersection of the road area and the reachable area can characterize the drivable area in most cases, which is consistent with our experience, and this is also the reason why we set $\lambda$ as a number greater than 1 in Eq. (12). Further, since there is no drivable area test dataset for quantitative evaluation, we qualitatively show the detection results in several complex off-road scenarios, see Fig. 16. Meanwhile, the results of ground segmentation and road labelling, and Bayesian fusion maps are also illustrated in Fig. 16. From experimental results, it is proved that our method can guarantee the real-time detection of drivable areas in front of the vehicle with promising performance, assisting autonomous vehicles with path planning in rough and unknown environments.

## 7 Conclusion and Future Work

In this paper, we present a framework based on the double projection of point clouds, including ground segmentation, road labelling and drivable area detection. Moreover, our method focuses on the combination of the elevation map and the range map, and some new morphological operations are applied to improve the robustness and real-time performance. More importantly, the decision-making method based on the Bayesian minimum error rate theory is introduced to ensure the stability of the system. And this fusion method takes into account the previous drivable area, the road area and the reachable area, so the final result is conducive to path planning of autonomous vehicles on rough and unstructured roads. Besides, the experiments in different situations verify the robustness and efficiency of our method.

Since only the positive obstacles around the vehicle are considered, the reachable area is more likely to be divergent if the obstacles are sparse. Although this problem can be solved by final data fusion, the richer the obstacle information in obstacle maps, the more accurate the subsequent drivable area detection can be. Therefore, the major work in the future is to expand the types of obstacles such as the negative obstacle and the dynamic obstacle with motion models, making drivable area detection more accurate and reliable.

**Supplementary Information** The online version contains supplementary material available at https://doi.org/10.1007/s10846-021-01381-7.

**Acknowledgements** This work was supported by National Key Research and Development Program of China (2018YFD0700602, 2017YFD0700303, and 2016YFD0701401), Youth Innovation Promotion Association of the Chinese Academy of Sciences (Grant No. 2017488), Independent Research Project of Research Institute of Robotics and Intelligent Manufacturing Innovation, Chinese Academy of Sciences (Grant No. C2018005), Equipment Pre-research Program (Grant No. 301060603), and Technological Innovation Project for New Energy and Intelligent Networked Automobile Industry of Anhui Province.

**Availability of Data and Material** The authors declare that all data and materials support our claims in the manuscript and comply with field standards. The data involved in our research include public dataset (SemKitti) and private dataset. The public dataset can be downloaded from the official website of SemKitti dataset. Our private dataset is currently not available.

**Code Availability** The custom code is currently not available.

**Authors' Contributions** Conceptualization: Fengyu Xu, Zhiling Wang; Methodology: Fengyu Xu, Linglong Lin; Formal analysis and investigation: Fengyu Xu, Linglong Lin; Writing - original draft preparation: Fengyu Xu, Linglong Lin; Writing - review and editing: Fengyu Xu, Linglong Lin, Zhiling Wang; Funding acquisition: Huawei Liang, Zhiling Wang; Resources: Huawei Liang; Supervision: Huawei Liang, Zhiling Wang.

**Funding** This work was supported in part by National Key Research and Development Program of China (2018YFD0700602, 2017YFD0700303, and 2016YFD0701401), Youth Innovation Promotion Association of the Chinese Academy of Sciences (Grant No. 2017488), Independent Research Project of Research Institute of Robotics and Intelligent Manufacturing Innovation, Chinese Academy of Sciences (Grant No. C2018005), Equipment Pre-research Program (Grant No. 301060603), and Technological Innovation Project for New Energy and Intelligent Networked Automobile Industry of Anhui Province.

## Declarations

**Ethics Approval** Not applicable.

**Consent to Participate** Not applicable.

**Consent for Publication** Not applicable.

**Conflicts of Interest/Competing Interests** The authors have no conflicts of interest to declare that are relevant to the content of this article.

## Appendices

This algorithm is used to eliminate the narrow gaps in the obstacle map, while it does not need to traverse all gaps resulting in real-time performance.

**Algorithm 1** Obstacle clustering and closing

```
Input: $d_{\text{th}}$ : distance threshold;
    $AllObsSet$ : all obstacle points in $ObsMap$;
    $ObsMap$ : a down-sample obstacle map (250×375);
Output: $ObsCloseMap$ : result of obstacle closing;
    $AllClusterList$ : result of obstacle clustering;
for $P_{ob} \in AllObsSet$ do
    $OneObsCluster \leftarrow \emptyset$;
    $ObsTemp \leftarrow \emptyset$;
    $NewObsTemp \leftarrow \emptyset$;
    if $ObsMap[P_{ob}.x][P_{ob}.y] > 0$ then
        $ObsTemp$.pushback($P_{ob}$);
        while (1) do
            for $p_p \in ObsTemp$ do
                $NewObsTemp \leftarrow \emptyset$ ;
                for $p_n$ in the $d_{\text{th}}$ neighborhood of $p_p$ do
                    if $ObsMap[p_n.x][p_n.y] > 0$ then
                        $NewObsTemp$.pushback($p_n$);
                        $OneObsCluster$.pushback($p_n$);
                        Line($p_n$,$p_p$) in $ObsCloseMap$;
                        $ObsMap[p_n.x][p_n.y]=0$;
                    end if
                end for
            end for
            if $NewObsTemp == \emptyset$ then
                $ObsClusterList$.pushback($OneObsCluster$);
                $OneObsCluster \leftarrow \emptyset$;
                break;
            else
                $ObsTemp \leftarrow NewObsTemp$;
            end if
        end while
    end if
end for
```

This algorithm is used to find reachable areas in the front of vehicles with the help of ObsCloseMap generated by Algorithm 1. Note that this algorithm is performed in a down-sampled image, so we need to up-sample the search result ReachAreaMap.

**Algorithm 2** Forward flood fill with a mask

```
Input: $Point(x_0, y_0)$ : starting point;
    $ObsCloseMap$ : a obstacle closing map;
    $MaskMap$ : a mask map with 250×375 pixels;
    $X_{\max}, X_{\min}, Y_{\min}, Y_{\max}$ : the search ranges;
Output: $ReachAreaMap$ : reachable area;
Initialize: $SeedSet$.pushback($x_0$)
for $i = Y_{\min}$ to $Y_{\max}$ do
    $SeedSetTemp \leftarrow \emptyset$;
    for $j = 0$ to $SeedSet$.size() do
        for $m = SeedSet[j]$ to $X_{\max}$ do
            IsReachablePoint($m$,$i$)
        end for
        for $n$=$X_{\min}$ to $Setseed[j]$ do
            IsReachablePoint($n$,$i$)
        end for
    end for
    $SeedSet \leftarrow SeedSetTemp$;
end for
Function: IsReachablePoint($x_t$,$y_t$){
if $ObsCloseMap[x_t][y_t]$==0 and $MaskMap[x_t][y_t]$==1 then
    $SeedSetTemp$.pushback($x_t$) ;
    $SeedSet$.delete($x_t$);
    $ReachAreaMap[x_t][y_t]$=1;
else
    break;
end if
}
```

## References

1. Sontges, S., Althoff, M.: Computing the drivable area of autonomous road vehicles in dynamic road scenes. IEEE Trans. Intell. Transp. Syst. **19**(6), 1855–1866 (2018)
2. Haselich, M., et al.: Spline templates for fast path planning in unstructured environments. IEEE/RSJ Int. Conf. Intell. Rob. Syst. pp. 3545–3550 (2011)
3. Liu, Z., Yu, S., Zheng, N.: A co-point mapping-based approach to drivable area detection for self-driving cars. Engineering. **4**(4), 479–490 (2018)
4. Wang, Z., et al.: Real-time Drivable Region Planning Based on 3D LiDAR. Int. Conf. Intell. Syst. Rem. Cont. pp. 563–572 (2019)
5. Wang, H., Sun, Y., Liu, M.: Self-supervised drivable area and road anomaly segmentation using RGB-D data for robotic wheelchairs. IEEE Robot. Autom. Lett. **4**(4), 4386–4393 (2019)
6. Li, Q., et al.: A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios. IEEE Trans. Veh. Technol. **63**(2), 540–555 (2013)
7. Yang, B., Liang, M., Urtasun, R.: Hdnet: Exploiting hd maps for 3d object detection. 2th Conf. Rob. Learn, pp. 146–155 (2018)
8. Chang, M., et al.: Argoverse: 3d tracking and forecasting with rich maps. Proc. IEEE Conf. Comput. Vis. Pattern Recognit. 8748–8757 (2019)
9. Liu, Z., Yu, S., Wang, X., Zheng, N.: Detecting Drivable Area for Self-driving Cars: An Unsupervised Approach. arXiv: 1705.00451 (2017)
10. Minaee, S., Boykov, Y., Porikli, F.: Image Segmentation Using Deep Learning: A Survey. arXiv: 2001.05566 (2020)
11. Manz, M., Himmelsbach, M., Luettel, T.: Detection and tracking of road networks in rural terrain by fusing vision and LIDAR. IEEE/RSJ Int. Conf. Intell. Rob. Syst. pp. 4562–4568 (2011)
12. Grilli, E., Menna, F., Remondino, F.: A review of point clouds segmentation and classification algorithms. Int. Arch. Photogramm. Remote. Sens. Spat. Inf. Sci. XLII-2/W3. 339–344 (2017)
13. Zermas, D., Izzat, I., Papanikolopoulos, N.: Fast Segmentation of 3D Point Clouds: A Paradigm on LiDAR Data for Autonomous Vehicle Applications. Proc. IEEE Int. Conf. Rob. Autom. pp. 5067–5073 (2017)
14. Rummelhard, L., et al.: Ground estimation and point cloud segmentation using SpatioTemporal Conditional Random Field. Proc. IEEE Intell. Veh. Symp. pp. 1105–1110 (2017)
15. Chen, T., Dai, B., Wang, R., Liu, D.: Gaussian-process-based real-time ground segmentation for autonomous land vehicles. J. Intell. Robot. Syst. Theor. Appl. **76**(3), 563–582 (2014)
16. Zhu, Z., Liu, J.: Graph-based ground segmentation of 3D LIDAR in rough area. IEEE Conf. Technol. Prac. Robot Appl. pp. 1–5 (2014)
17. Byun, J., et al.: Drivable Road Detection with 3D Point Clouds Based on the MRF for Intelligent Vehicle. Springer Tracts. Adv. Rob. pp. 49–60 (2015)
18. Gao, B., et al.: Off-Road Drivable Area Extraction Using 3D LiDAR Data. Proc. IEEE Intell. Veh. Symp. pp. 1505–1511 (2019)

19. Roynard, X., Deschaud, J., Goulette, F.: Classification of Point Cloud Scenes with Multiscale Voxel Deep Network. arXiv: 1804.03583 (2018)
20. Zhang, W., Zhou, C., Yang J., Huang K.: LiSeg: Lightweight Road-object Semantic Segmentation In 3D LiDAR Scans for Autonomous Driving. Proc. IEEE Intell. Veh. Symp. pp. 1021–1026 (2018)
21. Martínez, J., Morán, M., Morales, J., Reina, A., Zafra, M.: Field navigation using fuzzy elevation maps built with local 3D laser scans. Appl. Sci. **8**(3), 397 (2018)
22. Moosmann, F., Pink, O., Stiller, C.: Segmentation of 3d lidar data in non-flat urban environments using a local convexity criterion. Proc. IEEE Intell. Veh. Symp. pp. 215–220 (2009)
23. Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., Mahoney, P.: Stanley: the robot that won the darpa grand challenge. J. Field. Rob. **23**(9), 661–692 (2006)
24. Douillard, B., et al.: Hybrid elevation maps: 3D surface models for segmentation. IEEE/RSJ Int. Conf. Intelligent Rob. Syst. pp. 1532–1538 (2010)
25. Bogoslavskyi, I., Stachniss, C., Bonn: Efficient Online Segmentation for Sparse 3D Laser Scans. PFG-J. Photogramm. Remote Sens. Geoinf. Sci. **85**(1), 41–52 (2017)
26. Burger, P., Wuensche, H.: Fast Multi-Pass 3D Point Segmentation Based on a Structured Mesh Graph for Ground Vehicles. Proc. IEEE Intell. Veh. Symp. pp. 2150–2156 (2018)
27. Hughes, C., et al.: Drivespace: Towards context-aware drivable area detection. Intl. Symposium Electronic Imaging Science Technol. 2019(15), 42–1–42-9 (2019)
28. Guiotte, F., et al.: Semantic segmentation of LiDAR points clouds: Rasterisation beyond Digital Elevation Models. IEEE Geosci. Remote Sens. Lett. 1–4 (2020)
29. Meng, X., Lin, Y., Yan, L., Gao, X., Yao, Y., Wang, C., Luo, S.: Airborne LiDAR point cloud filtering by a multilevel adaptive filter based on morphological reconstruction and thin plate spline interpolation. Electronics. **8**(10), 1153 (2019)
30. Bogoslavskyi, I., Stachniss, C.: Fast range image-based segmentation of sparse 3D laser scans for online operation. IEEE Int. Conf. Intell. Rob. Syst. pp. 163–169 (2016)
31. Luo, Z., Mohrenschildt, M., Habibi, S.: A probability occupancy grid based approach for real-time lidar ground segmentation. IEEE Trans. Intell. Transp. Syst. **21**(3), 998–1010 (2019)
32. Shan, T., Wang, J., Englot, B.: Bayesian generalized kernel inference for terrain traversability mapping. 2th Conf. Rob. Learn. pp. 829–838 (2018)
33. Schreier, M., Willert, V., Adamy, J.: Compact representation of dynamic driving environments for ADAS by parametric free space and dynamic object maps. IEEE Trans. Intell. Transp. Syst. **17**(2), 367–384 (2016)
34. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification. John Wiley & Sons, Hoboken (2012)
35. Challa, S., Morelande, M.R., Musicki, D., Evans, R.J.: Fundamentals of object tracking. Cambridge University Press, Cambridge (2011)
36. Behley, J., et al.: SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. Proc IEEE Int. Conf. Comput. Vision. pp. 9297–9307 (2019)
37. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. Proc IEEE Comput Soc. Conf. Comput. Vision Pattern Recognit. pp. 3354–3361 (2012)

**Fengyu Xu** received his B.S. degree in measurement and control technology and instrumentation program from the school of mechanical engineering, Sichuan University in 2016. He is currently working toward a Ph.D. degree in detection technology and automation from the University of Science and Technology of China, and his laboratory is at Hefei Institutes of Physical Science, Chinese Academy of Sciences. His research interests involve machine learning, multiple object detection and tracking and environment perception in intelligent transportation systems.

**Huawei Liang** received a Ph.D. degree in detection technology and automation from the University of Science and Technology of China in 2007. He was a visiting scholar at the Chinese University of Hong Kong from December 2006 to December 2007. Now, He is a principal investigator and the deputy director of the Institute of Intelligent Machines, Hefei Institutes of Physical Science Chinese Academy of Science. In recent years, he has been responsible for and participated in more than ten scientific research projects from the National High Technology Research and Development Program and the National Fund Committee. He has been engaged in robotics, intelligent vehicle technology and systems, detection technology and automation device, pattern recognition and intelligent system, control theory and control engineering.

**Zhiling Wang** received a Ph.D. degree in control science and engineering from the University of Science and Technology of China in 2008. Now, he is a master tutor at the University of Science and Technology of China, and he is also an executive deputy director of the Intelligent Vehicle Technology Research Center, Institute of Intelligent Machines, Hefei Institutes of Physical Science, Chinese Academy of Science. He has also participated in research work from Anhui Engineering Laboratory for Intelligent Driving Technology and Application and Innovation Research Institute of Robotics and Intelligent Manufacturing Chinese Academy of Sciences. He has been mainly engaged in the research of driverless vehicles, environmental perception and understanding, machine vision and machine learning.

**Linglong Lin** received the B.S. degree in computer science and technology from Anhui University of Technology in 2010, the Ph.D. degree in nuclear science and engineering from the University of Chinese Academy of Sciences in 2016. He is currently an associate professor at the Institute of Intelligent Machines, Hefei Institutes of Physical Science, C.A.S, Hefei, China. Since 2016, he continuously completed in Intelligent Vehicle Future Challenge of China and Overcome Danger ground unmanned vehicle challenge of China. His research has focused on self-driving vehicles, such as LiDAR point cloud data processing, information fusion from multiple sensors system, object recognition and tracking, and deep learning.