IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON URBAN COMPUTING AND INTELLIGENCE

Received March 13, 2019, accepted April 9, 2019, date of publication April 22, 2019, date of current version May 3, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2912581

A Decision Tree Based Road Recognition Approach Using Roadside Fixed 3D LiDAR Sensors

JIANYING ZHENG[1], SIYUAN YANG[1], XIANG WANG[1], XIAOFANG XIA[2,3], YANG XIAO[4,3], AND TIESHAN LI[4]

[1]School of Rail Transportation, Soochow University, Suzhou 215131, China
[2]School of Computer Science and Technology, Xidian University, Xi'an 710071, China
[3]Department of Computer Science, The University of Alabama, Tuscaloosa, AL 35487, USA
[4]Navigation College, Dalian Maritime University, Dalian 116026, China

Corresponding author: Yang Xiao (yangxiao@ieee.org)

This work was supported in part by the Natural Science Foundation of Jiangsu Province under Grant BK20160324, in part by the Natural Science Foundation of Jiangsu Colleges and Universities under Grant 16KJB580009, in part by the Science and Technology Innovation Funds of Dalian under Grant 2018J11CY022, in part by the Key Research and Development Plan of Jiangxi Province under Grant 20181ACE50029, and in part by the National Natural Science Foundation of China under Grant 61673288.

ABSTRACT As one of the most important elements in the intelligent transportation system (ITS), the road traffic monitoring system (RTMS) needs to be functioned with a road recognition mechanism. Current works on road recognition mainly target at the field of automatic driving and cannot be directly used in the RTMS. In this paper, we propose a decision tree-based road recognition algorithm using roadside fixed light detection and ranging (LiDAR) sensors in the RTMS. These LiDAR sensors have a low vertical resolution, which implies that we cannot get a clear far boundary and obvious features of roads from the point cloud data. Point cloud data obtained by the roadside LiDAR sensors are projected onto a plane rasterized to grids of points. Using a decision tree, these grids are first classified into background grids and road grids. For reducing misclassification, these grids are further reclassified using a mean filtering algorithm. Finally, a minimum circumscribed rectangle algorithm is employed to obtain accurate road boundaries. The experiment results show that compared to existing road recognition algorithms, the proposed approach has advantages of being completely automatic, requiring shorter recognition time and having a wider detection range.

INDEX TERMS Road recognition, 3D LiDAR sensors, decision tree, mean filter algorithm, minimum circumscribed rectangle algorithm.

I. INTRODUCTION

Nowadays, roads in almost every country are crowded with vehicles so that we are flooded with news about traffic congestion and even accidents almost every day. It is reported that road traffic injury causes millions of deaths worldwide every year [1]. To alleviate this situation, many researchers are trying their best to develop an Intelligent Transportation System (ITS) [2], [3] which smartly takes advantage of multiple leading-edge information and communication technologies to improve safety, efficiency, and sustainability of transportation networks [4]. Road Traffic Monitoring System (RTMS) is one of the most important elements within the ITS domain [5]. Through analyzing data collected by detectors such as inductive loop sensors and video image processors and then distributing information concerning vehicle flow and road conditions to relevant stakeholders, RTMS aims at optimizing vehicle traffic flow and improving the road safety. To achieve this goal, RTMS needs to be functioned with a road recognition mechanism which responds for localizing and tracking road boundaries [6].

Currently, researchers have conducted extensive works on road recognition, which are mainly based on vision sensors and/or active light detection and ranging (LiDAR) sensors [7]. For vision-based lane recognition algorithms, the basic idea is to employ different types of cameras such as visible light and night-vision cameras to capture images of roads, which are then processed by different image

The associate editor coordinating the review of this manuscript and approving it for publication was Maode Ma.

processing methods to extract meaningful features such as road color, texture, edges, and brightness to segment lanes. The most commonly used image processing methods mainly include Sobel operator, Hough transform, random sample consensus (RANSAC), etc. For example, in paper [8], noisy lane edge features are detected using the Sobel operator and road images captured by cameras are divided into multiple subregions along the vertical direction. In paper [9], lane markings on the road are detected by dividing region of interest (ROI) into two subregions and applying the Hough transform in each subregion independently. In paper [10], a linear lane model and RANSAC are jointly used for detecting lanes, and a Kalman filter [11] is then used to refine the noisy output. However, this category of road recognition algorithms suffer from the drawback that it can be easily affected by changes of external environment. Specifically, these algorithms show much less robustness and much lower accuracy under poor visibility conditions such as shadow, illumination changes, and dramatically curved lanes where quality of the pictures captured by cameras degrades a lot.

In contrast, LiDAR sensors are not susceptible to external environment. Also, they have the capability to scan the 360° three-dimensional (3D) surrounding objects [12] with high resolution and precision. Thus, researchers are currently shifting their focuses on developing LiDAR-based road recognition techniques. For example, in paper [13], the authors propose a lane marker detection method which first projects point cloud data of 3D LiDAR sensors into a 2D grid map and then segment lanes from ground by comparing height differences between grids in a grid map. In paper [14], the authors present a real-time ground segmentation approach based on Gaussian process regression in a polar grid map for an autonomous land vehicle equipped with a LiDAR system. In paper [15], the authors develop a real-time intersection recognition and road boundary detection algorithm using a 3D LiDAR sensor on unmanned ground vehicles. Note that the above road recognition algorithms target at the field of automatic driving and serve for Advanced Driving Assistance System (ADAS) in ITS. LiDAR sensors used in these algorithms are equipped on vehicles and are usually 64-channel or at least 32-channel for achieving a real-time road recognition.

In this paper, we focus on road recognition in a RTMS where LiDAR sensors are fixed at a roadside. LiDAR sensors with more channels have higher accuracy but are correspondingly much more expensive. In the RTMS, for reaching a balance between government's limited budget and large demand for LiDAR sensors, we choose to deploy 16-channel LiDAR sensors at the roadside. Compared with 64-channel or 32-channel LiDAR sensors, the vertical resolution of 16-channel LiDAR sensors is much lower. This results in an unclear far boundary and a lack of obvious features for road recognition. Thus, we claim that the LiDAR-based road recognition algorithms already designed for/used in the ADAS cannot be directly applied in the RTMS for road recognition. For example, with 64-channel or 32-channel LiDAR sensors, we can use height information (i.e., average and variance) to accurately discriminate road surface and obstacles [6], [16], [17]. However, this cannot be done with 16-channel LiDAR sensors due to its low vertical resolution.

Currently, there are few researches on road recognition for RTMS which use LiDAR sensors with low vertical resolution fixed at roadside for traffic information collection. To the best of our knowledge, paper [18] is the only related work in this regard. In paper [18], the authors apply the DBSCAN algorithm to cluster vehicle trajectories for lane detection. Nevertheless, this method requires to calibrate the road area manually. In this paper, to address this limitation, we propose an automatic road recognition algorithm which consists of the following steps: first, point cloud data collected by the 3D LiDAR sensors are projected into an XOY plane which is then rasterized into grids of points; second, by analyzing how vehicles influence the distribution of points in grids, we extract five features, three of which are then selected to train a decision tree that is employed to classify the grids into background grids and road grids for the purpose of road recognition; last, we apply a mean filtering algorithm to filter the mistakenly classified grids and a minimum bounding rectangle algorithm to obtain the boundary of the road space. Contributions of this paper are summarized as follows:

- We propose an automatic road recognition approach for the RTMS which uses roadside fixed LiDAR sensors with low vertical resolution;
- The proposed approach applies subsequently a decision tree and a mean filtering algorithm to obtain a complete road region, which is further selected as the region of interest using a minimum bounding rectangle algorithm;
- We conduct experiments to choose appropriate parameters for the proposed approach;
- Compared with existing works, the proposed approach has the advantages of being completely automatic, requiring shorter recognition time, and having a wider detection range.

The remainder of this paper is organized as follows. In Section II, we explain how the proposed approach works. In Section III, we report some experimental results. We conclude this paper in Section IV.

II. THE PROPOSED APPROACH

In this section, we demonstrate the working strategy of the proposed approach. For better understanding, we show the working flow in Fig. 1. As shown in the figure, we first project the point cloud data of 3D LiDAR sensors onto an XOY plane which is a Cartesian coordinate system which has two perpendicular lines (the x-axis and y-axis) in a plane. This XOY plane is immediately further rasterized into grids of points (shown in Section II-A). Afterwards, for the purpose of road recognition, feature extraction and feature analysis are conducted for selecting appropriate features which are then used to train the decision tree (we will explain feature extraction and feature analysis in Section II-B and Section II-C, respectively; and demonstrate the training of the

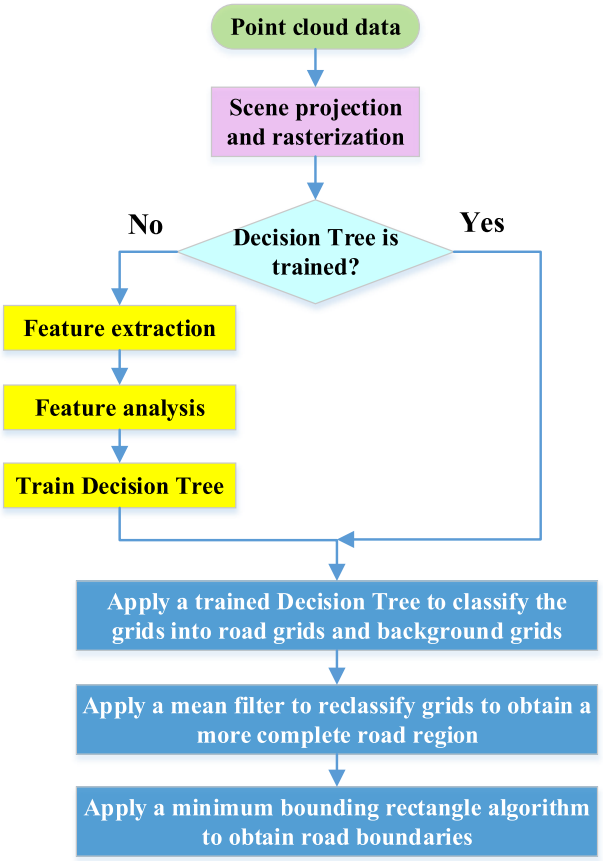

FIGURE 1. The working flow of the proposed approach.

decision tree in Section II-D). We first employ the decision tree to classify the grids into road grids and background grids, and then apply a mean filter algorithm to reclassify the grids for reducing misclassification and obtaining a more complete road space (shown in Section II-E). We finally utilize a minimum bounding rectangle algorithm to obtain accurate road boundaries (explained in Section II-F).

A. SCENE PROJECTION AND RASTERIZATION

In real applications, the data collected by 3D LiDAR sensors fixed at the roadside are stored in the form of 3D point clouds. We show a frame of the 3D point cloud data in Fig. 2, from which we can observe blue semi-circular curves. These curves actually represent the points on a flat and even road on which the LiDAR sensors are reflected. Disordered points at the upper left corner and the bottom right corner of Fig. 2 represent trees and bushes at both sides of the road. Different colors correspond to different reflection intensities, as indicated by the legend located at the right side of Fig. 2. The 3D point clouds are so complicate that we cannot directly use them to conduct feature extraction and analysis, which is a prerequisite for road recognition. To make the data more understandable, we project all point cloud data onto an XOY plane. LiDAR sensors collect 3D point cloud data by rotating the reflection source which is driven by its own motor.

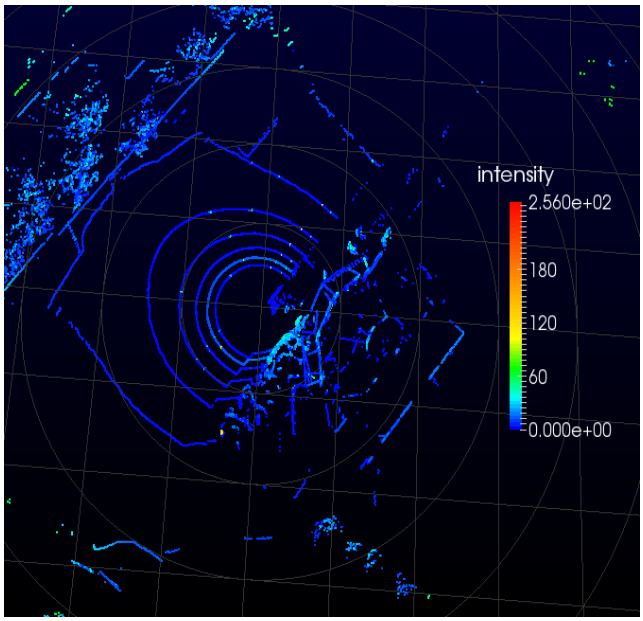

FIGURE 2. A frame of 3D point cloud data collected by a LiDAR sensor.

When the motor works, a certain amount of jitter is unavoidable. Such jitter is amplified with the increase of the distance. Besides, when wind goes up, trees and shrubs at the roadside tremble significantly. The jittering of LiDAR sensors' motor and the trembling of trees and shrubs can cause errors. To be more specific, 3D point cloud data in different frames for the same location may contain different information. To remove such kind of errors, we further rasterize the XOY plane into grids of points.

B. FEATURE EXTRACTION

The main difference between road space and background space lies in that there are vehicles and pedestrians moving in the road, whereas there are not in the background space. Thus, in this paper, we select the feature of the dynamic movement of vehicles and pedestrians as the basis to do the road recognition.

When vehicles pass through, the density of points in the road changes significantly. This is because when vehicles enter LiDAR sensors' detection area, they reflect some light from distant trees and roads to these LiDAR sensors, which makes the points on the road space denser. This inspires us to extract the following feature:

Feature 1: Variance of Point Density (VPD): On the XOY plane where point cloud data is projected, the number of points in a certain grid is called the point density (PD) of the grid. Let n denote the number of frames during a certain period of time. Let $X_{PD}(i)$ denote the point density of a certain grid at the i-th frame, where $i \in \{1, 2, \ldots, n\}$. Let μ_{PD} denote the average of point density. Then, we have

$$\mu_{PD} = \frac{1}{n}\sum_{i=1}^{n} X_{PD}(i).$$

Let σ_{PD} denote the VPD. Then, we have

$$\sigma_{PD} = \frac{1}{n}\sum_{i=1}^{n}(X_{PD}(i) - \mu_{PD})^2.$$

Also, the distance and height of vehicles change the information of the point cloud data greatly. Thus, we extract the following two features:

Feature 2: Variance of Average of Point Distance (VAPD): Let k denote the number of points in a certain grid. Let $X_{Dis}(i,j)$ denote the distance from point j in a certain grid to a LiDAR sensor at the i-th frame, where $i \in \{1,2,\ldots,n\}$, $j \in \{1,2,\ldots,k\}$. Let $\mu_{PD}(i)$ denote the average of point distance in a certain grid at the i-th frame. Then, we mathematically have

$$\mu_{PD}(i) = \frac{1}{k}\sum_{j=1}^{k}X_{Dis}(i,j).$$

Let $\bar{\mu}_{PD}$ denote the average of point distance in a certain grid during a certain period time. Then, we mathematically have

$$\bar{\mu}_{PD} = \frac{1}{n}\sum_{i=1}^{n}\mu_{PD}(i).$$

Let σ_{APD} denote the VAPD. Then, we have

$$\sigma_{APD} = \frac{1}{n}\sum_{i=1}^{n}(\mu_{PD}(i) - \bar{\mu}_{PD})^2.$$

Feature 3: Variance of Average of Point Height (VAPH): Let $X_{Ht}(i,j)$ denote the height of point j in a certain grid at the i-th frame, where $i \in \{1,2,\ldots,n\}, j \in \{1,2,\ldots,k\}$. Let $\mu_{PH}(i)$ denote the average of point heights in a certain grid at the i-th frame. Then, we have

$$\mu_{PH}(i) = \frac{1}{k}\sum_{j=1}^{k}X_{Ht}(i,j).$$

Let $\bar{\mu}_{PH}$ denote the average of point heights in a certain grid during a certain period of time. Then, we have

$$\bar{\mu}_{PH} = \frac{1}{n}\sum_{i=1}^{n}\mu_{PH}(i).$$

Let σ_{APH} denote the VAPH. Then, we have

$$\sigma_{APH} = \frac{1}{n}\sum_{i=1}^{n}(\mu_{PH}(i) - \bar{\mu}_{PH})^2.$$

In the scene, the background is fixed and cannot move, and this implies that they appear in the same grid in every rasterized grid map (which means the rasterized XOY plane for a certain frame of 3D point cloud data). In contrast, the vehicles keep moving, which implies that the grids where they appear in different grid maps keep varying. Based upon the above information, we extract the following feature:

Feature 4: Average of Point Frequency (APF): Let $X_{PF}(i)$ denote the point frequency which indicates whether there are points in a certain grid at the i-th frame, where $i \in \{1,2,\ldots,n\}$. Specifically, if there are no points in the grid, we let $X_{PF}(i)= 0$; otherwise, we let $X_{PF}(i)= 1$. Technically, we have

$$X_{PF}(i)= \begin{cases} 0, & X_{PD}(i) = 0 \\ 1, & X_{PD}(i) > 0 \end{cases}.$$

Let μ_{PF} denote the average of point frequency during a certain period of time. Then, we have

$$\mu_{PF} = \frac{1}{n}\sum_{i=1}^{n}X_{PF}(i).$$

Since materials of vehicles are different from that of roads, buildings and trees, LiDAR sensors' reflection intensity varies with the entering of vehicles. Thus we extract the following feature:

Feature 5: Variance of Average of Point Intensity (VAPI): Let $X_{It}(i,j)$ denote the intensity of point j at the i-th frame, where $i \in \{1,2,\ldots,n\}, j \in \{1,2,\ldots,k\}$. Let $\mu_{PI}(i)$ denote the average of point density in a certain grid at the i-th frame. Then, we have

$$\mu_{PI}(i) = \frac{1}{k}\sum_{j=1}^{k}X_{It}(i,j).$$

Let $\mu_{PI}(i)$ denote the average of point density in a certain grid for a certain period of time. Then, we have

$$\bar{\mu}_{PI} = \frac{1}{n}\sum_{i=1}^{n}\mu_{PI}(i).$$

Let σ_{API} denote the VAPI. Then, we mathematically have

$$\sigma_{API} = \frac{1}{n}\sum_{i=1}^{n}(\mu_{PI}(i) - \bar{\mu}_{PI})^2.$$

C. FEATURE ANALYSIS

In this section, we conduct feature analysis to screen out less important features such that dimension disaster can be avoided during the training process of the decision tree.

In order to screen out the unimportant features, we need to conduct feature subset selection. Feature subset selection requires two parts: a search strategy to select candidate subsets, and an objective function to evaluate these candidates [19]. There are several algorithms for the search strategy such as exponential algorithm, sequential algorithm, and randomized algorithm. The exponential algorithm evaluates a number of subsets that grows exponentially with the dimensionality of the search space. The sequential algorithm add or remove features sequentially, but has a tendency to become trapped in local minima. The randomized algorithm incorporates randomness into their search procedure to escape local minima. There are three approaches for the objective function in the feature subset selection, namely

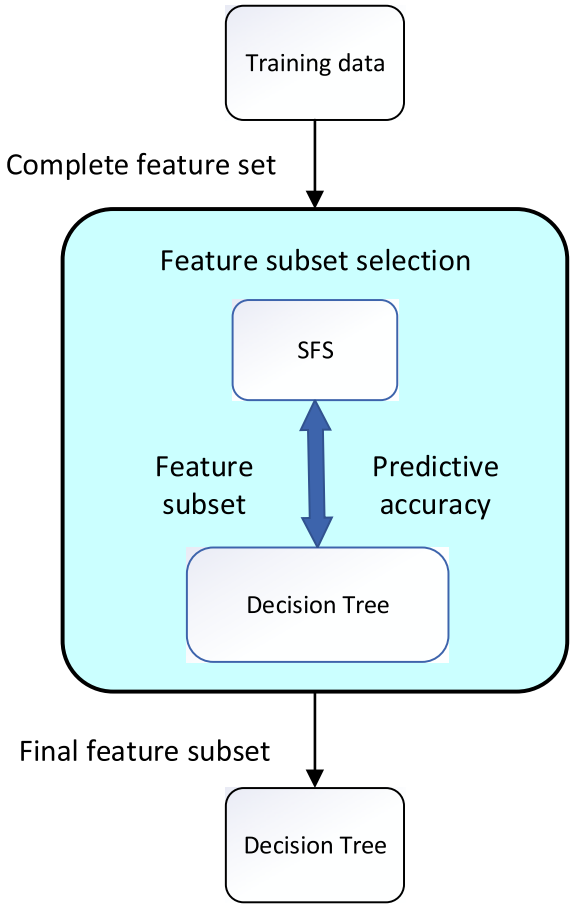

FIGURE 3. The flow chart of feature subset selection.

filter, wrapper, and embedded [20]. The filter approach evaluates subsets by their information content, e.g., interclass distance, statistical dependence, or information-theoretic measures. The wrapper approach uses a classifier to evaluate subsets by their predictive accuracy (on test data) by statistical re-sampling or cross-validation. The embedded approach picks out the features that are important to the training of the model in the process of confirmation model. The filter and embedded methods are faster than the wrapper method, but the wrapper method is more accurate than the other methods because the wrapper approach uses the classifier itself to evaluate the subset [21]. Fig. 3 is the flow chart of the feature subset selection in this paper. We apply the Sequential Forward Selection (SFS) as the search strategy due to its simplicity and high speed. Moreover, we use the wrapper approach as the objective function due to its accuracy. The SFS is used to search the optimal feature subset. The predictive accuracy of the decision tree is the feedback of the objective function. Until the feature subset with maximum accuracy is selected, feature subset selection is finished.

Feature analysis is conducted as follows: first, we apply subset search to choose a subset of features that yield the minimum classification error; then, we calculate the information gain of each feature to validate that the features are appropriately selected.

1) SUBSET SEARCH

The Sequential Forward Selection (SFS) are applied as the subset search procedure due to its simplicity and high speed [22]. As one of the simplest greedy search algorithms, SFS starts from an empty set, and then sequentially adds features selected by an objective function which evaluates the performance of the feature subsets. Note that in this paper, we use the decision tree to evaluate the candidate subset.

TABLE 1. Confusion matrix.

Actual class	Predicted class: Positive	Predicted class: Negative
Positive(P)	TP (true positive)	FN (false negative)
Negative(N)	FP (False positive)	TN (true negative)

The feature subset is used as the input feature of the decision tree, and the accuracy of the decision tree is used to evaluate the quality of the feature subset. In order to calculate the accuracy, we build a sample set, which contains road grids and background grids. Moreover, the number of the road grids is similar to the number of background grids, and this aims to balance the distribution of road and background grids and to improve the performance of the classifier. Then, we randomly select part of the grids as the training set and the remaining grids are set as the test set. We use the features and the training set to train the decision tree. Comparing the test set labels and the predicted labels, we obtain the accuracy of Table 2, Table 3, Table 4, Table 5, Table 6. The confusion matrix shown in Table 1 explains the accuracy computation of decision tree:

1) 'positive' means road grids and 'negative' means background grids;

2) actual class denotes the real class label;

3) predicted class denotes the result of classification by the decision tree;

4) True positive (TP) is the number of the correct classified road grids;

5) True negative (TN) is the number of correct classified background grids;

6) False positive (FP) is the number of road grids misclassified as background grids;

7) False negative (FN) is the number of background grids misclassified as road grids;

8) P is the number of road grids in the test set;

9) N is the number of background grids in the test set.

Therefore, we have:

$$accuracy = \frac{TP + TN}{P + N}. \tag{1}$$

For example, the sample set has 1266 elements, among which there are 610 road grids and 656 background grids. 80% of the sample set is selected as the training set and 20% of the sample set is the test set. We have $P = 127$, $N = 126$,

$TP = 127$, $TN = 121$, $FN = 0$, and $FP = 5$. When we plug these values into equation 1, we obtain $accuracy = 98.02\%$.

The new extended feature set should produce a higher classification accuracy compared with any other feature set [22]. The detailed procedure of SFS subset search is presented as follows:

In the first round, each feature is used as a candidate subset to train the decision tree to classify the calibrated samples. Accuracy rates are recorded in Table 2. When APF is chosen as the candidate feature subset, the decision tree shows the highest accuracy. Thus, in this round, we choose the subset of APF as the best feature subset.

TABLE 2. Classification accuracy of single feature decision tree.

Single feature	Accuracy
VPD	94.47
VAPH	74.54
APF	96.84
VAPD	64.43
VAPI	78.66

In the second round, features other than APF are respectively added to the subset of APF to train the decision tree. After the test samples are classified with the trained decision tree, we calculate the accuracy, which is shown in Table 3. As can be seen, when the subset of APF and VPD is chosen, the decision tree has the highest accuracy.

TABLE 3. Classification accuracy of double feature decision tree.

Double features	Accuracy
APF, VPD	97.23
APF, VAPH	96.84
APF, VAPD	96.84
APF, VAPI	96.84

TABLE 4. Classification accuracy of three feature decision tree.

Three features	Accuracy
VPD, APF, VAPH	97.63
VPD, APF, VAPD	95.65
VPD, APF, VAPI	96.44

In the third round, we use three features to train the decision tree. Two features of them are APF and VPD which are selected in the last round, and the third feature is one feature out of VAPH, VAPD, and VAPI, respectively. As can be seen in Table 4, when the set of APF, VPD, and VAPH is selected, the accuracy of the decision tree is the highest.

In the fourth round, we use four features to train the decision tree. The first three features are APF, VPD, and VAPH, and the fourth feature is VAPD and VAPI, respectively. We record the classification results in TABLE 5. As can be seen in Table 5, the accuracy rate in both cases remains the same with that when the subset features of VPD, APF and VAPH are applied. Hence, we can conclude that the subset of APF, VPD and VAPH is the subset that we are looking for.

TABLE 5. Classification accuracy of four feature decision tree.

Four features	Accuracy
VPD, VAPH, APF, VAPD	97.63
VPD, VAPH, APF, VAPI	97.63

2) SUBSET EVALUATION

Information entropy [23] is the most commonly used index to measure the purity of a sample set. A smaller information entropy implies a higher purity of the sample set. In this section, we employ information gain to evaluate the features to train the decision tree. A feature with greater information gain should be selected with priority.

All the five features extracted in Section II-B are continuous. For calculating the information gain of continuous features, we need to first discretize them by using a dichotomy algorithm which proceeds as follows: first, for a sample set, the values of a specific feature are sorted in an ascending or a descending order; then, the average value of every two adjacent feature values is calculated as the partition point which obviously divides samples into two parts; last, the information gain of the feature is determined as the largest information gain achieved at all the partition points.

To illustrate how the dichotomy algorithm proceeds, we in the following show how to calculate the information gain of the feature of VPD. Assume that we have a set of samples D, each of which has a VPD feature value σ_{PD}. (1) First, we sort the samples in the VPD values' descending order. (2) Afterwards, we get $|D| - 1$ partition points. Let $T(i)$ denote the i-th partition point. Let $\sigma_{PD}(i)$ denote the value of σ_{PD} ranked at the i-th order. Then, we have

$$T(i) = \frac{\sigma_{PD}(i) + \sigma_{PD}(i+1)}{2}, \forall 1 \leq i \leq |D| - 1,$$

where "$|\cdot|$" denotes the cardinality of a set. The partition point $T(i)$ divides the sample set into two subsets which are respectively denoted by $D_+(i)$ and $D_-(i)$. Note that we here use $D_+(i)$ to denote the subset of samples with feature value σ_{PD} larger than $T(i)$ and $D_-(i)$ the subset of samples with feature value σ_{PD} not larger than $T(i)$. (3) Let $H_+(i)$ and $H_-(i)$ denote the information entropy in the sample subset of $D_+(i)$ and $D_-(i)$, respectively. Then, in line with the definition of information entropy [24], we have

$$H_+(i) = -\sum_{k=1}^{2} p_k \log_2 p_k,$$

and

$$H_-(i) = -\sum_{k=1}^{2} p_k \log_2 p_k,$$

respectively. Take $H_{-}(i)$ as an example, where i equals to 1 or 2. p_1 represents the proportion of the road grids in the sample set smaller than $T(i)$. p_2 indicates the proportion of the background grids in the sample set smaller than $T(i)$. Let $Gain(D, \sigma_{PD})$ denote the information gain of the VPD. Then, we have

$$Gain(D, \sigma_{PD}) = \max_{1 \le i \le \lambda_0 - 1} \left\{ H(D) - \left[\frac{|D_+(i)|}{|D|} H_+(i) + \frac{|D_-(i)|}{|D|} H_-(i) \right] \right\}. \quad (2)$$

We present the information gain for evaluating the five features selected in Section II-B in Table 6. As can be seen, the features of VAPD and VAPI has the smallest information gain. This validates that the feature subset of APF, VPD, and VAPH that are filtered in the last sub-section are appropriately selected.

TABLE 6. Features' information gain.

Feature	Information gain
VPD	0.4235
VAPH	0.4042
APF	0.5197
VAPD	0.3246
VAPI	0.2335

D. GRID CLASSIFICATION

With three features of APF, VPD, and VAPH, one can figure out many ways to do the road recognition by classifying background grids and road grids. The simplest way is to choose a certain threshold for each feature. However, this approach is faced with the problem that an appropriate threshold is not easy to be selected and an inappropriate threshold usually implies poor accuracy of classification. In contrast, machine learning techniques do not need to deal with the threshold selection problem [25]. Compared with other machine learning techniques, the decision tree has advantages of low computational complexity, being easy to interpret and requiring fewer data sets. Since Iterative Dichotomiser 3 (ID3) builds the fastest decision tree, in this paper, we apply it to generate the decision tree to classify the background grids and road grids [26]. The central idea of the ID3 algorithm is to measure the selection of features by information gain. The decision tree selects the features with the greatest information gain after splitting. The ID3 algorithm has been widely used because of its simplicity and the good capability of noise resistance.

The specific procedure goes as follows. First, the point cloud data collected by a LiDAR sensor fixed at roadside during a certain period are projected to an XOY plane, based on which the features of PF, VPD, and VAPH are calculated. Then, some grids are calibrated as the sample set, 80% of which are set as the training set and the remaining samples are set as the test set. After many times of training, the classifier with the highest accuracy is chosen as the final decision tree. Note that during the training process, we apply the cross-validation technique. Afterwards, we carry out a pruning process to address the data over-fitting problem. This also helps improve the classification accuracy of the decision tree.

TABLE 7. Accuracy of the decision tree.

	Total	# of road grids	# of background grids
Sample set	1266	610	656
Training set	1013	488	525
Test set	253	127	126
Classification results	253	132	121
Accuracy	98.02%	96.21%	96.03%

We record classification accuracy of the trained decision tree in Table 7. As seen in the table, the sample set contains 1266 elements in total, among which there are 610 road grids and 656 background grids. When we consider road grids and background grids together, the classification accuracy is 98.02%. When we consider just road grids, the classification accuracy is 96.21%. When we consider just background grids, the classification accuracy is 96.03%.

FIGURE 4. Road recognition results by applying only the decision tree.

E. NOISE FILTERING

We show road recognition results by applying only decision tree in Fig. 4, where gray and white grids represent background grids and road grids, respectively, and black grids represent grids with no 3D point cloud data. As shown in Fig. 4, most grids are correctly identified. However, some road regions (especially those far from LiDAR sensors) discontinue. This is mainly due to occlusion of LiDAR sensors

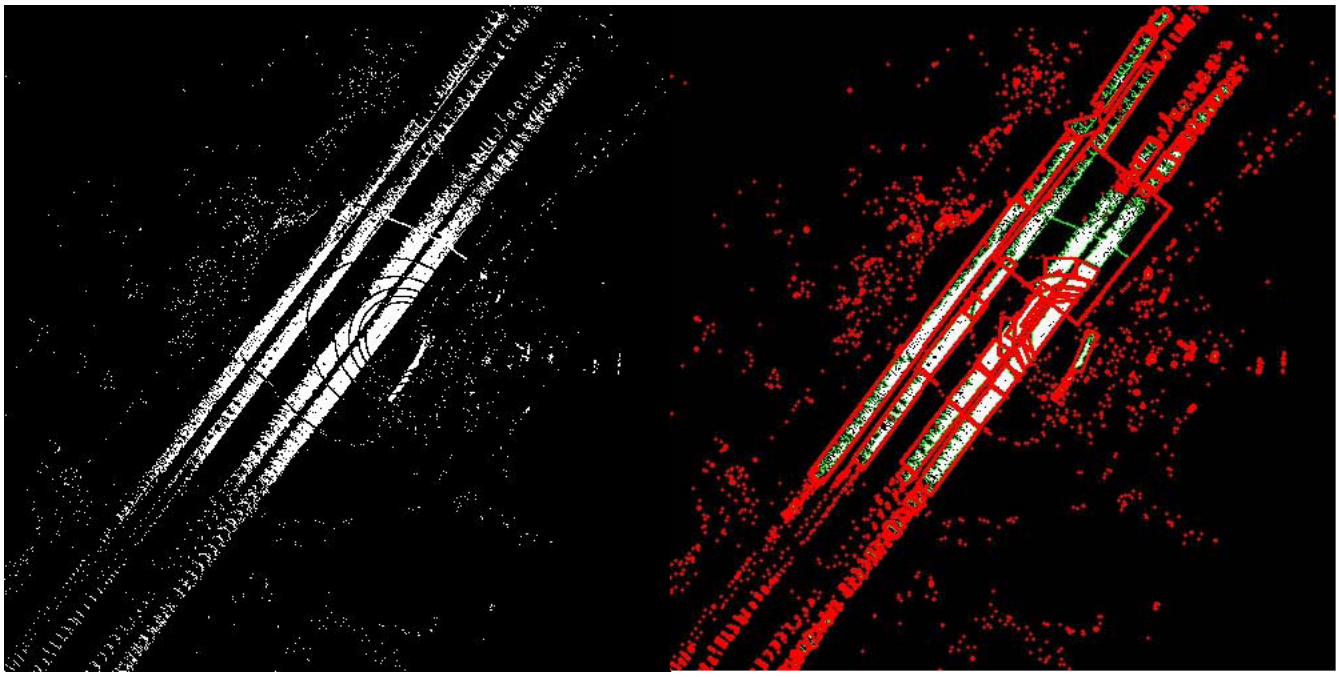

FIGURE 5. The result of ROI selection by white grids.

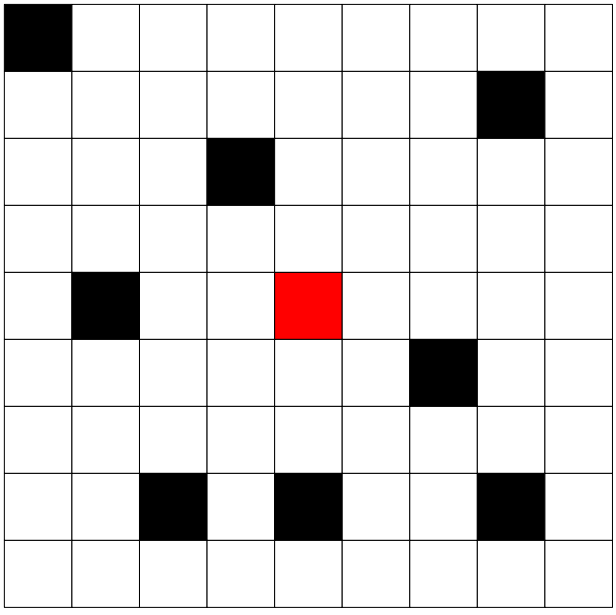

FIGURE 6. An example to illustrate the mean filter algorithm.

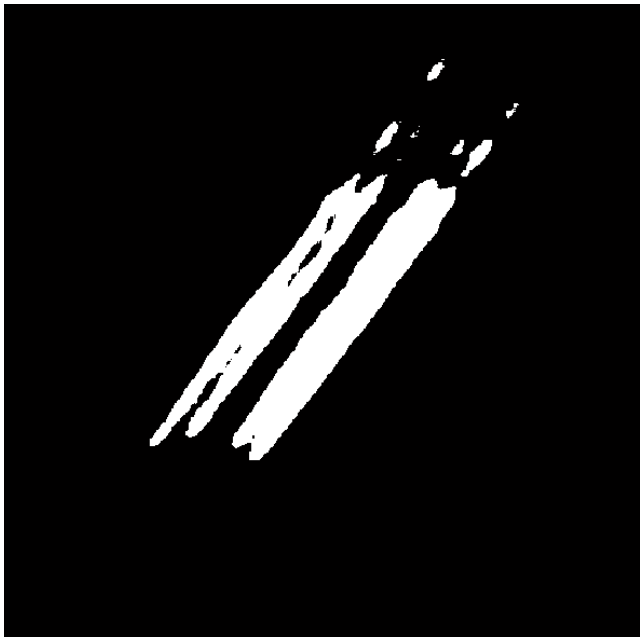

FIGURE 7. Road recognition results after applying mean filtering algorithm to reclassify grids which have first been classified by the decision tree.

which makes points in these road grids too sparse such that road grids are mistakenly classified as background grids.

These grids will bring difficulty for the selection of the region of interest (ROI). For example, In Fig. 5, we use the minimum circumscribed rectangle of the white grids to get the ROI. But the result is not ideal. Because the original white area is discontinuous. In order to get the continuous and regular road area, we use the mean filtering to obtain smooth image edges and effectively filter out noise points. The mean filtering can correct the misclassified grids. We focus on addressing this issue in the following.

As aforementioned, the decision tree classifies grids on the XOY plane into background grids and road grids. Let us set values of background grids and road grids as 0 and 1, respectively. Then, the scene can be regarded as an image

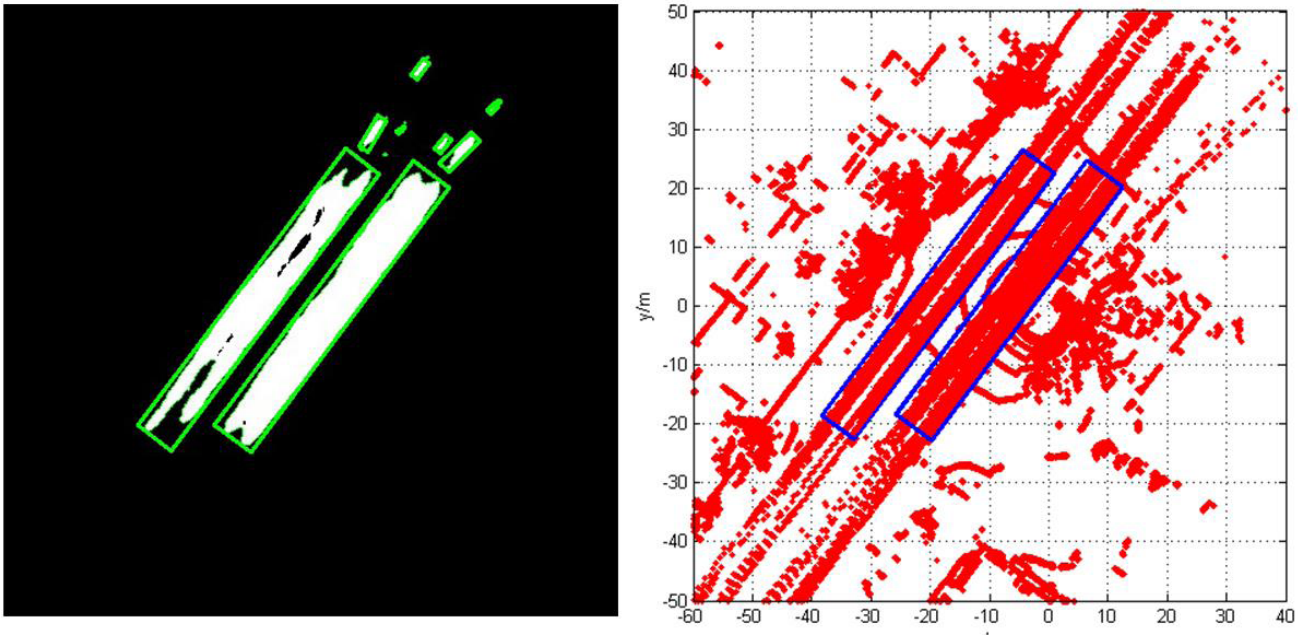

FIGURE 8. Road recognition results after applying a minimum bounding rectangle algorithm.

which is made up of pixels and contains some noise which needs to be removed. As a linear image filtering algorithm, the mean filtering can obtain smooth image edges and effectively filter out noise points. Furthermore, it has advantages of being simple, intuitive, and easy for implementation. Thus, we apply the mean filtering to reduce noise in the image of grids on the XOY plane.

For each target grid, the mean filtering operates as follows: First, a kernel at the center of which the target grid locates is determined. Note that the kernel represents the shape and size of the neighborhood to be sampled when calculating the mean. In Fig. 6, the kernel is chosen as a 9-by-9 square, and black grids and white grids represent background grids and road grids, respectively. The center grid in red is the target grid. Second, the value of center grid is replaced with the mean value of its neighbors, including itself. Note that the mean is between 0 and 1. For example, in Fig. 6, the value of center grid should be update as $72/81$ (note that the target grid is first classified as background grids). Third, the value of center grid is rounded such that one can judge whether the grid is a road grid or a background grid. Since $72/81$ is rounded to 1, we can judge that in Fig. 6 the target grid should be reclassified as a road grid.

It is rationale to apply the mean filtering algorithm here because points nearby should have similar properties and if one point differs from most of surrounding points, it is probably misclassified. The mean filtering has the effect of eliminating points which are unrepresentative of their surroundings. In Fig. 7, we present the road recognition results after applying the mean filtering algorithm to reclassify grids which have been first classified by the decision tree. Note that black grids and white grids represent background grids and road grids, respectively.

This road consists of two left lanes, two right lanes, and a middle lane. The middle lane is a turning lane. Each lane can be distinguished in Fig. 4. However, as the distance between lanes in the same direction is so close that the threshold of the median filtering is too small, the information of lane distinction will be incomplete. Therefore, accurate single-lane information cannot be obtained in Fig. 4. However, it can be got through the median of the information of the two lanes in the same direction. Obviously, compared with the results in Fig. 4, road regions in 7 are much more continuous and regular.

F. ROI SELECTION

Since roads are rectangular, in this section, we apply the minimum bounding rectangle algorithm to select the region of interest (ROI). The process is conducted using functions in the OpenCV library. First, the grid map is transformed into a binary gray image. Then, we use the *findContour*s function to obtain the image contour. Afterwards, we apply the *minAreaRect* function to get the minimum circumscribed rectangle. As shown in the left figure of Fig. 8, at this point, the left and right lanes are distinctly distinguished and an isolation belt locates in between. Note that green lines represent road boundaries. By mapping the rectangles to actual scene, we can see that rectangular boundaries and actual road boundaries are entirely consistent, as shown in the right figure of Fig. 8.

Fig. 9 is the diagram of the experimental scene. The range of the scene in this paper is $[X_3, X_2]$ and $[Y_2, Y_1]$. The grid

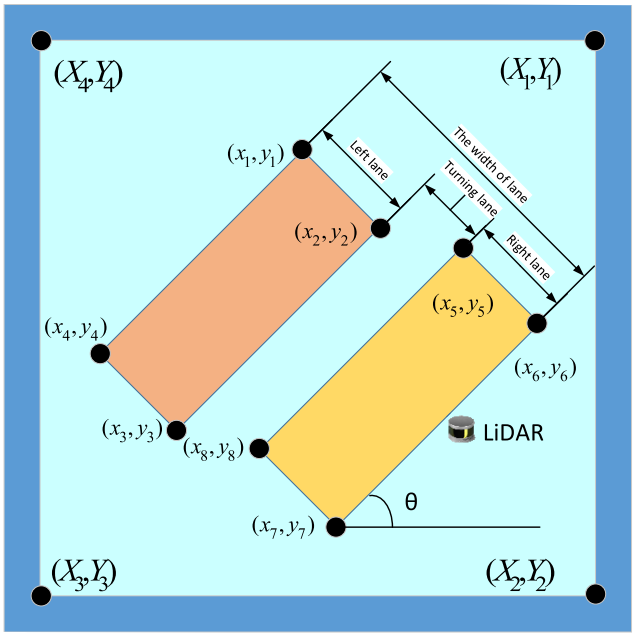

FIGURE 9. The diagram of the experimental scene.

size is G meters. This is a map consisting of $m \times n$ grids.

$$m = \lceil \frac{Y_1 - Y_2}{G} \rceil.$$
$$n = \lceil \frac{X_2 - X_3}{G} \rceil.$$

We convert the grid map into an image consisting of $m \times n$ pixels. Then the pixel position of the p-th row and the r-th column on the image, i.e., the corresponding actual position of the space, is:

$$(x, y) = \begin{cases} x = X_3 + q \times G \\ y = Y_1 - p \times G \end{cases}$$

By the rectangle obtained by the minimum circumscribed rectangle, we can get the boundary pixels of left and right lane. According to the calculating formula, we can get the corresponding actual position coordinates. Because there is an angle between the road and the horizontal angle, it is necessary to calculate the angle θ first. We calculate the road slope angle:

$$\theta = \arctan \frac{y_6 - y_7}{x_6 - x_7}.$$

The left lane width is:

$$w_{left} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

The right lane width is:

$$w_{right} = \sqrt{(x_5 - x_6)^2 + (y_5 - y_6)^2}.$$

The width of a single-lane can be got by the median of the width of a double-lane.

Because there is no the minimum circumscribed rectangle in the turning lane, it is necessary to calculate its width by the coordinates of the other two lanes. Firstly, we turn the coordinates system to horizontal angles. The formula of coordinate rotation is as follows.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

According to the formula, what can be obtained are (x'_1, y'_1), (x'_2, y'_2), (x'_5, y'_5), (x'_6, y'_6).

The turning lane width is:

$$w_{turning} = y'_2 - y'_5.$$

The total lane width is:

$$w_{lane} = y'_1 - y'_6.$$

III. EXPERIMENTAL RESULTS AND DISCUSSIONS

In the experiment, we place a Velodyne's 16-channel LiDAR sensor (called the VLP-16 LiDAR sensor) near a T intersection in the state of Nevada for the purpose of collecting data, as shown in Fig. 10. The VLP-16 LiDAR sensor creates 360° 3D images by using 16 laser/detector pairs mounted in a compact housing. It has an effective scanning radius of 100m, a low power consumption (~ 8 W), a light weight (830 g), a compact footprint (~ Φ 103 mm x 72 mm), a dual return capability and a reasonable price. These features make the VLP-16 LiDAR sensor an ideal equipment to be deployed at the roadside to serve for the RTMS. The scanning frequency of VLP-16 LiDAR sensor is 10Hz, which means that one frame lasts 0.1s. The Google satellite map of the intersection is shown in the Fig. 10. As can be seen, this is a two-way five lane main road with a total width of 19.85m. It consists of two left lanes, two right lanes and a middle lane, with each being 3.65m. There are also a left sidewalk and a right sidewalk, with each being 0.8m.

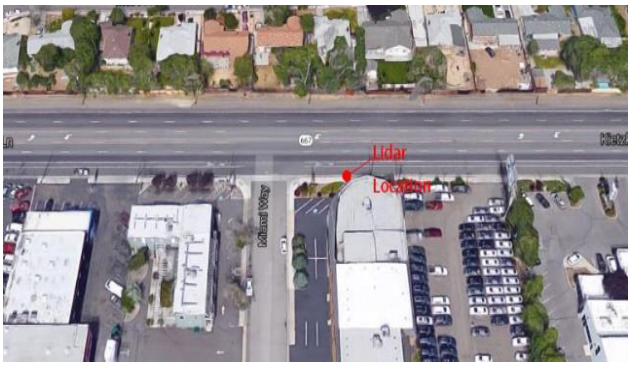

FIGURE 10. Data collection site [18].

A. CHOOSING AN APPROPRIATE GRID SIZE

Since the proposed method does the road recognition by classifying the grids on the rasterized XOY plane into background grids and road grids, we can easily infer that the grid size is critical in extracting an accurate road space. On the whole, too large grids imply blurred road boundaries, and too small

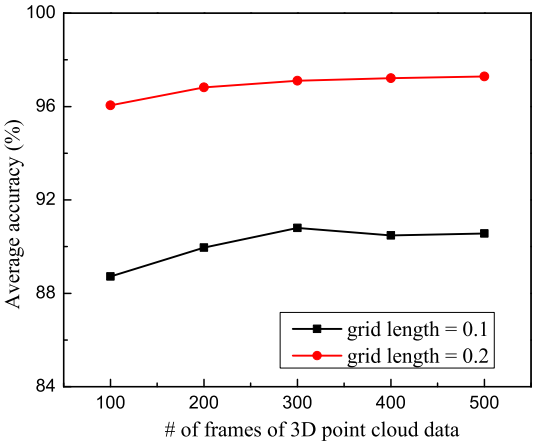

FIGURE 11. Experimental results for choosing an appropriate grid size.

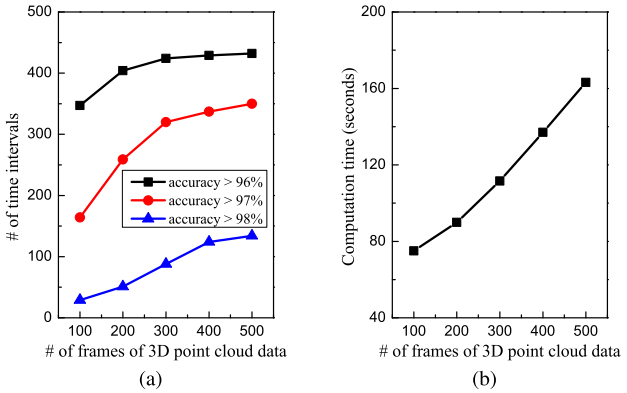

FIGURE 12. Experimental results for choosing an appropriate number of frames.

grids come with misclassification and a huge amount of calculation. An appropriate grid size should meet the following requirements: (1) it guarantees that we can get a precise road boundary; (2) it incurs a small amount of calculation; (3) the grid classification accuracy should be high. From Fig. 11, we can observe that no matter how many frames of 3D point cloud data are used, the average accuracy is obviously lower in the case where the grid length is set as 0.1m than in the case where the grid length is set as 0.2m. In the experiment, we observe that road boundaries get blurred when the grid length is set larger than 0.2m. Thus, in this paper, the appropriate grid length should be 0.2m.

B. CHOOSING AN APPROPRIATE NUMBER OF FRAMES

The road has a speed limit of 40km/h. It takes 10s for a car from appearing to vanishing in the detecting range of the VLP-16 LiDAR sensor. Hence, the minimum length of experiment time is 10s. As aforementioned, for the chosen Velodyne's 16-channel LiDAR sensor, one frame lasts for 0.1s. Thus, the minimum number of frames should be 100. At the T intersection in the state of Nevada where we placed the LiDAR sensor, we choose 460 time intervals for collecting data. For each such time interval, we train the decision tree with 100 frames, 200 frames, 300 frames, 400 frames and 500 frames of data, respectively. We show the classification results in Fig. 12(a), where X-axis represents the number of frames (which ranges from 100 to 500) and the Y-axis represents the number of time intervals whose classification accuracy exceeds 96%, 97%, and 98%, respectively. As can be seen, for a given number of frames, the number of time intervals during which the accuracy of grids classification exceeds 96% is larger than the number of time intervals during which the grid classification accuracy exceeds 97%. The number of time intervals during which the grid classification accuracy exceeds 98% is the smallest. Furthermore, more frames usually imply more time intervals during which the grid classification accuracy exceeds a certain number. However, from Fig. 12(b), we can observe that calculation burden increases with the number of frames. When the experiment time rises from 400 frames to 500 frames, the grid classification accuracy stays almost the same, whereas the computation time still keeps increasing. Thus, we set the number of frames as 400.

C. CHOOSING AN APPROPRIATE KERNEL SIZE FOR MEAN FILTERING

The size of kernel in the mean filtering algorithm impacts the performance of the filter greatly. Specifically speaking, a too large neighborhood produces blurred edges of the road, whereas a too small neighborhood does not have a good filtering effect. In Fig. 13, we aim at finding the appropriate size of neighborhood. The grid size is set as 0.2m, and we set the size of kernel as 11*11, 13*13, 15*15, 17*17 and 19*19, respectively. As can be seen, the road area gets more complete when the size of kernel increases. However, when the size of kernel is set as 17*17 or 19*19, some background grids are mistakenly classified as lane grids. Furthermore, from Table 8, we can see that road recognition results under kernel size of 15*15 are more consistent with actual situation than other kernel sizes. Thus, in the experiment, we set the size of kernel of the mean filtering algorithm as 15*15.

TABLE 8. Road recognition results under different size of kernels of the mean filtering algorithm.

Unit(m)	Left lane	Right lane	Turning lane	Width of the lane
Actual width	7.3	7.3	3.65	18.25
11*11	5.58	6.02	5.24	17.04
13*13	6.36	6.64	4.21	17.21
15*15	7.10	7.40	3.61	18.11
17*17	7.38	8.12	3.02	18.52
19*19	8.00	8.83	2.65	19.48

D. COMPARING WITH RELATED WORKS

As aforementioned, paper [18] is currently the only work on road recognition for RTMS which use LiDAR sensors

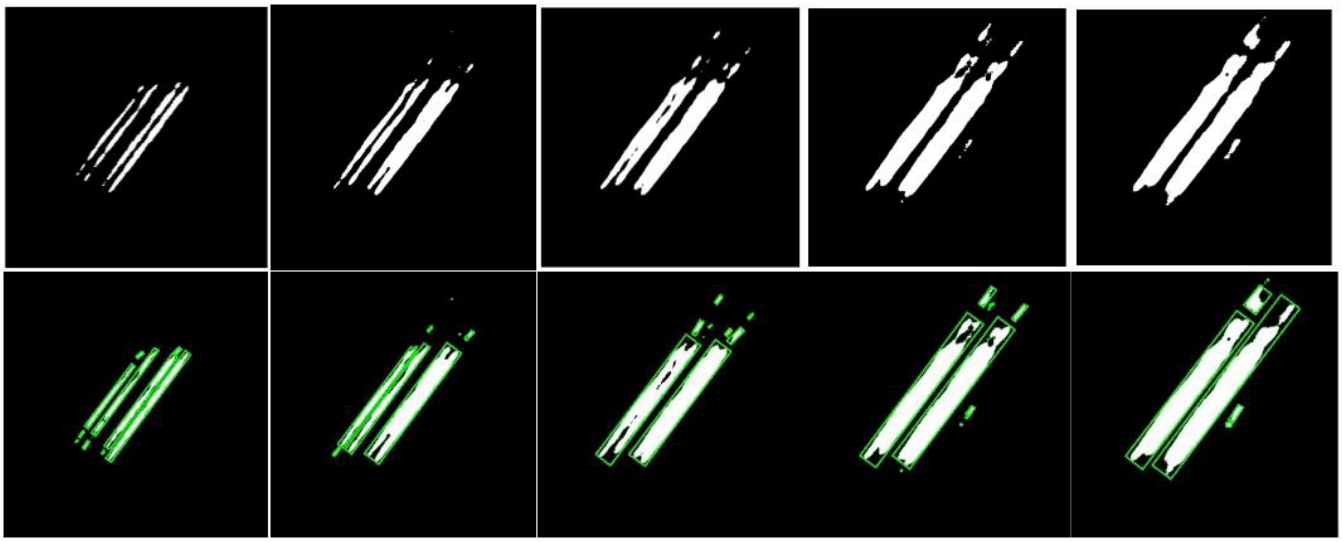

FIGURE 13. Experimental results for choosing an appropriate kernel size for the mean filtering algorithm.

TABLE 9. Comparison with the approach in paper [18].

comparison aspects	approach in paper [18]	The proposed approach
Feature Selection	Single feature	Multiple features
Recognition method	DBSCAN	Decision Tree
Required # of frames	2000 frames	400 frames
Lane identification	Manual	Automatic
Range	45 m	80 m

with low vertical resolution fixed at roadside. In TABLE 8, we compare our proposed approach with the approach in paper [18]. As can be observed, our proposed approach has the advantages of being totally automatic, requiring fewer frames of 3D point cloud data, and having a larger detection range.

IV. CONCLUSION

In this paper, we propose a decision tree based road recognition algorithm using roadside fixed 16-channel LiDAR sensors with low vertical resolution. We first project 3D point cloud data collected by LiDAR sensors onto the XOY plane, which is then rasterized to grids of points. Experiment results show that the size of grids should be set as 0.2m. By analyzing how vehicles impacts the distribution of points on the XOY plane, we extract the following five features: VPD, VAPD, VAPH, APF, and VAPI. The results of feature analysis show that VPD, VAPH, and APF are the most important features. Thus, the above three features are then used to train the ID3 decision tree which classifies the grids into road grids and background grids for the purpose of road recognition. Afterwards, for reducing the misclassification, we apply the mean filtering algorithm to filter the noise points and get a more complete road space. Experiment results show that an appropriate size for the kernel of the filtering algorithm should be 15*15. We finally employ the minimum bounding rectangle algorithm to obtain road boundaries. Experiment results show that compared to existing works, the proposed approach can perform road recognition more accurately and faster.

REFERENCES

[1] World Health Organization. (2018). *Road Traffic Injuries*. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[2] J. Zheng *et al.*, "Non-intrusive traffic data collection with wireless sensor networks for intelligent transportation systems," *Ad Hoc Sensor Wireless Netw.*, vol. 34, no. 1, pp. 41–57, 2016.

[3] B. Xu, J. Zheng, Q. Wang, Y. Xiao, and S. Ozdemir, "An adaptive vehicle detection algorithm based on magnetic sensors in intelligent transportation systems," *Ad Hoc Sensor Wireless Netw.*, vol. 36, nos. 1–4, pp. 211–232, 2017.

[4] E. M. Irandu, "Towards efficient management of public transportation in the city of nairobi through application of intelligent transport systems (ITS)," *World Rev. Intermodal Transp. Res.*, vol. 2, no. 1, pp. 72–83, 2008.

[5] J. Ying, "Highway traffic automatic detection system based on video and image processing," in *Intelligence Computation and Evolutionary Computation*. Berlin, Germany: Springer, 2013, pp. 521–526.

[6] G. Lu, "A lane detection, tracking and recognition system for smart vehicles," Ph.D. dissertation, Univ. Ottawa, Ottawa, ON, Canada, 2015.

[7] J. Zheng, B. Xu, X. Wang, X. Fan, H. Xu, and G. Sun, "A portable roadside vehicle detection system based on multi-sensing fusion," *Int. J. Sensor Netw.*, vol. 29, no. 1, pp. 38–47, 2019.

[8] D.-J. Kang and M.-H. Jung, "Road lane segmentation using dynamic programming for active safety vehicles," *Pattern Recognit. Lett.*, vol. 24, no. 16, pp. 3177–3185, 2003.

[9] V. Gaikwad and S. Lokhande, "Lane departure identification for advanced driver assistance," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 910–918, Apr. 2015.

[10] A. Borkar, M. Hayes, and M. T. Smith, "Robust lane detection and tracking with ransac and Kalman filter," in *Proc. 16th IEEE Int. Conf. Image Process. (ICIP)*, Nov. 2009, pp. 3261–3264.

[11] B. B. Alagoz, M. Erturkler, and C. Yeroglu, "A theoretical investigation on moving average filtering solution for fixed-path map matching of noisy position data," *Int. J. Sensor Netw.*, vol. 29, no. 4, pp. 213–225, 2019.

[12] A. Boualem, Y. Dahmani, C. D. Runz, and M. Ayaida, "Spiderweb strategy: Application for area coverage with mobile sensor nodes in 3D wireless sensor network," *Int. J. Sensor Netw.*, vol. 29, no. 2, pp. 121–133, 2019.

[13] S. Kammel and B. Pitzer, "Lidar-based lane marker detection and mapping," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2008, pp. 1137–1142.

[14] T. Chen, B. Dai, R. Wang, and D. Liu, "Gaussian-process-based real-time ground segmentation for autonomous land vehicles," *J. Intell. Robotic Syst.*, vol. 76, nos. 3–4, pp. 563–582, 2014.
[15] 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 IEEE Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2015, pp. 499–504.
[16] F. Moosmann, O. Pink, and C. Stiller, "Segmentation of 3D Lidar data in non-flat urban environments using a local convexity criterion," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2009, pp. 215–220.
[17] L. Chen, Y. He, J. Chen, Q. Li, and Q. Zou, "Transforming a 3-D LiDAR point cloud into a 2-D dense depth map through a parameter self-adaptive framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 1, pp. 165–176, Jan. 2017.
[18] J. Wu, H. Xu, and J. Zheng, "Automatic background filtering and lane identification with roadside LiDAR data," in *Proc. 20th IEEE Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–6.
[19] R. Bryll, R. Gutierrez-Osuna, and F. Quek, "Attribute bagging: Improving accuracy of classifier ensembles by using random feature subsets," *Pattern Recognit.*, vol. 36, no. 6, pp. 1291–1302, 2003.
[20] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Elect. Eng.*, vol. 40, no. 1, pp. 16–28, 2014.
[21] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artif. Intell.*, vol. 97, nos. 1–2, pp. 273–324, 1997.
[22] A. Marcano-Cedeño, J. Quintanilla-Domínguez, M. Cortina-Januchs, and D. Andina, "Feature selection using sequential forward selection and classification applying artificial metaplasticity neural network," in *Proc. 36th IEEE Annu. Conf. Ind. Electron. Soc. (IECON)*, Nov. 2010, pp. 2845–2850.
[23] N. T. T. Nga, N. K. Khanh, and N. H. Son, "Entropy correlation-based clustering method for representative data aggregation in wireless sensor networks," *Int. J. Sensor Netw.*, vol. 28, no. 4, pp. 270–283, 2018.
[24] R. K. Pathria and P. D. Beale, *Statistical Mechanics*, 3rd ed. New York, NY, USA: Academic, 2011.
[25] E. U. Warriach and K. Tei, "A comparative analysis of machine learning algorithms for faults detection in wireless sensor networks," *Int. J. Sensor Netw.*, vol. 24, no. 1, pp. 1–13, 2018.
[26] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

Authors' photographs and biographies not available at the time of publication.