Mathematical Method for Lidar-based Obstacle Detection of Intelligent Vehicle

Binbin Sun, Wentao Li, Huibin Liu, Pengwei Wang, Song Gao and Penghang Feng

Abstract—**Obstacle detection based on 3D laser radar plays an important role on environment perception and safe operation of intelligent vehicle. The obstacle detection method based on a 3D 32-line lidar is studied in this paper. Firstly, the lidar data and the data coordinate system were calibrated. Then, road boundary suspected points were screened based on road height and smooth features. Moreover, RANSAC algorithm was used for data fitting to achieve road boundary extraction. For point clouds within the road boundary, ground and non-ground point clouds were identified by ground plane fitting method. In order to process the non-ground point clouds and realize the recognition and detection of obstacles, an improved DBSCAN algorithm was designed, which can perform clustering processing and generate the bounding boxes representing obstacles. Finally, vehicle test was carried out. Test results show that the obstacle detection method proposed in this paper can extract the road boundary accurately. Furthermore, the non-ground point cloud within road boundary and the obstacles in the non-ground point cloud were identified and distinguished.**

Index Terms—**intelligent vehicle, 3D laser radar, visual obstacle detection, improved DBSCAN algorithm**

I. Introduction

The rapid development of modern intelligent technology has brought new opportunity for the development of intelligent vehicle, which is of great significance for road traffic safety [1,2]. Autonomous driving technology of intelligent vehicle consists of environment perception, path planning, intelligent decision-making and vehicle motion control [3-6]. Environmental perception is the foundation for intelligent vehicle to realize autonomous driving, which includes information acquirement based on various sensors and object identification via data processing. Consequently, methods to identify and distinguish obstacles for intelligent vehicle are the key to environment perception [7].

Currently, obstacle detection technology can be divided into three categories according to sensors used in vehicle, including machine vision, millimeter-wave radar and lidar [8]. Vision-based obstacle detection technology mainly relies on the data collected by the camera. It has the advantages of large detection range and perfect target information, but is unable to obtain depth information [9]. Millimeter wave radar has high resolution and is less influenced by weather, so it can provide real-time feedback to the relative position, relative speed, relative distance and other indicators of obstacles. However, the millimeter wave radar is unable to obtain target contour due to the high noise [10].

Lidar-based obstacle detection is to obtain obstacle information according to the time difference between laser source transmitting and receiving laser beam [11]. As an active ranging system, lidar has centimeter-level accuracy in measuring obstacles (centimeter level) and perfect real-time performance. Lidar plays an important role in such fields as obstacle detection and segmentation, navigable area detection, high-precision map drawing and positioning, and obstacle trajectory prediction [12,13]. Obstacle detection can be realized based on either 2D lidar or 3D lidar. Because of the fast scanning speed and small amount of data, 2D lidar is mostly used for locating vehicles and pedestrians in structured scenes. Owe to rich information and wide scanning range, 3D lidar is widely used in obstacle detection under urban working conditions with high environmental complexity.

Currently, raster-based method is a common obstacle detection method of 3D lidar. First of all, the method projects scanning point cloud into grids. Then, it conducts clustering analysis based on the information in each grid to determine whether the corresponding grid is obstacle or ground. Clustering analysis has a great influence on obstacle detection [14]. Currently, the prototype-based, the hierarchical and density-based clustering methods are commonly used [15]. The prototype-based clustering method assumes that the clustering structure can be characterized by a group of prototypes, and then iteratively updated and solved after initialization. This method is difficult to achieve irregular sample points. The hierarchical clustering method divides the sample points into different layers.

According to sample density, the density-based clustering method is able to determine whether the samples belong to the same class, so as to achieve the clustering results. This

Manuscript received September 22, 2020; revised December 18, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 51805301, the Natural Science Foundation of Shandong under Grant ZR2019BEE043, the Postdoctoral Science Foundation of China and Shandong under Grant 2020M680091 and 202003042, the key R & D projects in Shandong under Grant 2019GHZ016.

Binbin Sun, is a professor in the School of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China (e-mail: sunbin_sdut@126.com).

Wentao Li, the corresponding author, is a graduate student in the School of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China (e-mail: liwentao1213@126.com).

Huibin Liu, is a graduate student in the School of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China (e-mail: 18764623813@163.com).

Pengwei Wang, is a lecturer in the School of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China (e-mail: wpwk16@163.com).

Song Gao, is a professor in the School of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China (e-mail: gaos546@126.com).

Penghang Feng, is a graduate student in the School of Transportation and Vehicle Engineering, Shandong University of Technology, Zibo, China (e-mail: fengpenghang@saicmotor.com).

method can realize arbitrary shapes and high dimensional features. At present, DBSCAN is a commonly used density clustering method to cluster point cloud of arbitrary shape with a large number of noise points [16]. However, the application of this method is limited to two shortcomings. Specially, the distance threshold of the sample neighborhood and the threshold of the sample number in the neighborhood cannot be adaptively modified according to the distance between the detection area and the radar. Furthermore, for the large data set, the method leads to multiple clustering core points, long clustering convergence time and large memory usage.

Consequently, this paper proposes an improved DBSCAN algorithm to detect obstacle based on lidar. The lidar point cloud data is preprocessed in chapter 2. Chapter 3 screens the points of road boundary and fits them by RANSAC algorithm to achieve road boundary extraction. In chapter 4, the point clouds within the road boundary are divided into ground clouds and non-ground clouds based on ground plane fitting. In chapter 5, an improved DBSCAN algorithm is proposed to cluster non-ground point clouds and generate contour of obstacles. Chapter 6 carries out vehicle test and data analysis.

II. Lidar Data Preprocessing

In this paper, a Velodyne hdl-32e lidar is used, which can measure seven hundred thousand points per second. Data preprocessing, data calibration and coordinate system calibration are necessary for obtaining effective data information.

A. Lidar Data Calibration

Ideally, the coordinate origin of lidar is the point where the laser emission source is located inside the radar, that is, the midpoint of the longitudinal array of 32-line laser beams. However, due to the deviation of the position and direction angle of the laser beam source during processing and installation, each set of laser beams has its own set of calibration parameters. Consequently, the return value of lidar is the distance and angle information in polar coordinate system, which needs to be converted to Cartesian coordinate system for calibration. In the Cartesian coordinate system, the coordinate value (P_x,P_y,P_z) of the measured laser point can be expressed as follows.

$$\begin{cases} D_{real} = D_0 + D_{cor} \\ D_{xy} = D_{real}\cos\theta - V_0\sin\theta \\ P_x = D_{xy}\sin\beta - H_0\cos\beta \\ P_y = D_{xy}\cos\beta + H_0\sin\beta \\ P_z = D_{real}\sin\theta + V_0\cos\theta \end{cases} \tag{1}$$

where D_0 is the distance value returned by lidar with noise in polar coordinate system. D_{cor} is the distance correction coefficient, which represents the distance deviation of laser beam. D_{real} is the actual distance from the obstacle to the origin of lidar in the polar coordinate system. θ is the rotation correction angle, representing the angular deviation of the plane laser beam and the laser coordinate Y axis. β is the vertical correction angle, which represents the included angle between the laser beam and the plane X-O-Y. V_0 is the vertical offset, which represents the offset from the launching point of X-O-Z plane laser beam to the origin of the radar coordinate system. D_{xy} is the projection distance of the end point of laser beam on the horizontal plane in Cartesian coordinate system. H_0 is the horizontal offset, representing the offset from the starting point of the horizontal laser beam to the origin of the radar coordinate system.

B. Calibration of Coordinate System

During the installation of lidar, the lidar coordinate system has relationship with the vehicle coordinate system due to pitch, roll and deflection. As the surrounding environment information obtained by the laser radar provides data basis for the subsequent decision-planning and motion control, the lidar coordinate system needs to be unified under the vehicle body coordinate system.

Therefore, the vehicle body coordinate system is defined as the world coordinate system (O_w-$X_wY_wZ_w$). The origin of the coordinate system is the center of mass of the vehicle. Two coordinate systems, namely the reference coordinate system of lidar (O_{lb}-$X_{lb}Y_{lb}Z_{lb}$) and the actual coordinate system of lidar (O_{lr}-$X_{lr}YlrZ_{lr}$), are designed for lidar. As shown in figure 1, the difference between the lidar reference coordinate system and the vehicle centroid coordinate system is on the Z-axis. Furthermore, the actual lidar coordinate system differs from the reference coordinate system by one side inclination on the X-axis and one pitching angle on the Y-axis.

According to the lidar geometric model, each laser point (D_i, I) of lidar can be converted into (P_x,P_y,P_z) in the radar coordinate system. By establishing the relationship between the actual coordinate system of the laser radar and the world coordinate system (body coordinate system), the (P_x,P_y,P_z) in the actual coordinate system of the radar can be converted to (P_x',P_y',P_z' in the world coordinate system. The actual coordinate system of the lidar is (O_{lr}-$X_{lr}Y_{lr}Z_{lr}$), the origin is the center point of the lidar transmitter, the longitudinal direction of the vehicle body is the X-axis, the transverse direction is the Y-axis, and the vertical direction is the z-axis. Given the above analysis, the relationship between the actual coordinate system of lidar and the body coordinate system of intelligent vehicle is expressed by rotation matrix and translation vector.

$$\begin{bmatrix} P_x' \\ P_y' \\ P_z^1 \end{bmatrix} = R\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} + T + Z \tag{2}$$

where R is the rotation matrix. Z is the translation vector from the lidar reference coordinate system to the actual coordinate system. T is a translation vector of the lidar coordinate system origin in the X-axis, Y-axis and Z-axis of the vehicle body coordinate system, which can be confirmed by installation position and vehicle body coordinate system origin. Since both the lidar origin and the world coordinate system origin are in Z axis, T is defined as $[t_x,t_y,t_z]^T$.

During installation process, the roll angle around the X-axis, the pitch angle around the Y-axis, and the deflection angle around the Z-axis show deviations in different degrees. The error of R directly determines the error of the final return value of lidar origin, so accurate calibration of the lidar is necessary. It is assumed that the pitch angle is α, the roll

angle is β', the deflection angle is γ'. Since the lidar is installed in the center of the longitudinal vertical plane of the vehicle body, only α and β' need to be calibrated. As shown in Fig. 2, by using a rectangular calibration plate, the roll angle of the lidar is calibrated.

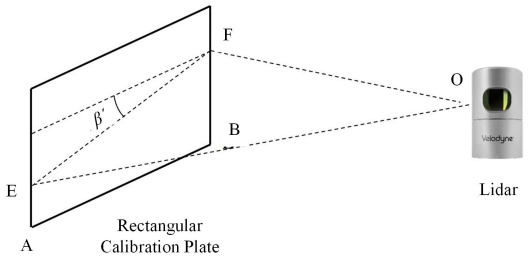


Fig. 1. The calibration of lidar roll angle

According to the rectangular calibration plate, the angle FOE is the azimuth difference between the edge points E and F. l_{OE} and l_{OF} are the distance between the two points (E and F) and the lidar. Based on the cosine theorem, l_{EF} can be calculated. The roll angle β' is as follow:

$$\beta' = \arccos\left(\frac{l_{AB}}{l_{EF}}\right) \tag{3}$$

Where l_{AB} is the width of rectangular calibration board.

The lateral inclination transformation matrix R_y of the radar is:

$$R_y = \begin{bmatrix} \cos\beta' & 0 & \sin\beta' \\ 0 & 1 & 0 \\ \sin\beta' & 0 & \cos\beta' \end{bmatrix} \tag{4}$$

Furthermore, the pitch angle is calibrated by using an isosceles triangle calibration plate. As shown in Fig. 2, the triangular calibration plate is placed at point A_1. The angle F_1OE_1, l_{OE1}, and l_{OF1} can be obtained by radar data. Similarly, l_{EF1} can be obtained by the cosine theorem.

$$l_{E_1Z_1} = l_{E_1F_1}\cos\gamma' \tag{5}$$

$$l_{BZ_1} = l_{BC} - l_{E_1Z_1} \tag{6}$$

Then, the triangle calibration plate is moved to point A_2. Similarly, l_{BZ2} is obtained. The radar elevation angle is calculated as:

$$\alpha = \arctan\left(\frac{l_{BZ_1} - l_{BZ_2}}{l_{A_1A_2}}\right) \tag{7}$$

Radar inclination transformation matrix R_x is shown as follow:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \tag{8}$$

Finally, the point (P_x', P_y', P_z') in the world coordinate system can be converted into (P_x, P_y, Pz) in the lidar coordinate system:

$$\begin{bmatrix} P_x' \\ P_y' \\ P_z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta' & 0 & \sin\beta' \\ 0 & 1 & 0 \\ \sin\beta' & 0 & \cos\beta' \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} + T + Z \tag{9}$$

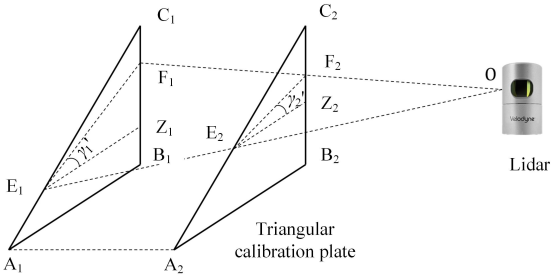


Fig. 2. The calibration of lidar pitch angle

III. Road Boundary Identification and Extraction of Lidar Point Cloud Data

In order to segment ground and non-ground point clouds and deal with fewer point clouds in the non-ground point cloud, obstacles outside the road boundary line need to be removed. Consequently, in this section, the road boundary line is identified and extracted according to the lidar point cloud data.

A. Road Width Limit

During the operation of the vehicle, the distance between the left and the right boundary points of the road is defined as the road width. The distance between the center line of the vehicle body and the boundary points of the road is less than the road width. Here, (x_l,y_l,z_l) and (x_r,y_r,z_r) are defined as candidate left and right boundary points of road, respectively. The two points should meet the following conditions:

$$\begin{cases} 0 < |x_l| < W \\ 0 < |x_r| < W \\ W - \Delta W \le |x_l| + |x_r| \le W + \Delta W \end{cases} \tag{10}$$

where W is the transverse distance between the left and right boundary points of the road, i.e., the road width. ΔW is the error of road width.

B. Extraction of Height Features and Smooth Features

(1) Extraction of height feature. 3D points of the lidar are segmented and projected onto the X-O-Y plane by using a grid. Then in each grid, the height difference between the highest point Z_{max} and the lowest point Z_{min} of the laser spot is calculated. For a given grid, if $D_1 \le (Z_{max} - Z_{min}) \le D_2$, the points in the grid are added to the set of height feature points.

(2) Extraction of smooth feature. The smoothing feature of the point cloud means the smoothing characteristics of the lidar in a certain scanning area. In the same plane, the smoothness of the lidar scanning line shows perfect smoothing feature, while at the junction of different planes, the smoothness feature is poor. The Velodyne HDL-32E lidar point cloud data used in this paper is stored separately according to the 32 laser beam bundles. In a scanning period T, the point cloud data set of the 32 laser beams is defined as ST={P_0, P_1, ..., P_{31}}. The local smoothing feature is obtained by the point set P_k of each line. In this paper, $P_{k,i}$ is a random point on the point set P_k. L is a set of point clouds on the scan line between P_k and $P_{k,i}$. The smoothing feature is:

$$S = \frac{1}{|L| \cdot \|P_{k,i}\|} \left\| \sum P_{k,i \in L, j \ne i} (P_{k,j} - P_{k,i}) \right\| \tag{11}$$

According to the structural pavement smoothness characteristics, a smoothness threshold S_0 is selected for the experimental road boundary. If $S>S_0$, the smoothing characteristics are poor, and edge folding occurs in the laser beam P_k. If $S<S_0$, the smoothing characteristics are good, and there is no edge folding in the laser beam P_k. Therefore, the point $P_{k,i}$ is deleted. Finally, points in the intersection of height feature point set and smoothness feature point set are selected as candidate road boundary points.

C. Road Boundary Line Fitting

Since the common least square fitting method is highly sensitive to noise the road boundary fitting effect will be deviated when the lidar scanning point is slightly bumped by the vehicle body. Therefore, in this paper, random sampling consistency (RANSAC) algorithm is used to fit the boundary line [17,18]. The algorithm is described as follows:

(1) Suppose there are n significant feature point sets $X=\{X_1, X_2... X_n\}$, the corresponding feature point set in the other view is $Y=\{Y_1, Y_2... Y_n\}$. The number of point sets is greater than or equal to eight.

(2) Initialize the iteration count k=1.

(3) Randomly select eight significant feature points from X to fit the polar geometry model and obtain the basic matrix F.

(4) Based on the basic matrix F, project all the feature points in X to another view. The feature point set $X'=\{X_1', X_2'... X_n'\}$ is confirmed.

(5) Given a deviation ε, calculate the residuals between the corresponding feature points point in set X and Y. Furthermore, calculate the number of the residual errors within the scope of deviation. If the number of feature points is greater than a given threshold t, the model is fitted depending on the same feature point set, and the feature point set is extracted. Then, the algorithm is terminated.

(6) Set $k=k+1$. If k is less than a given value, skips to the step 3. Otherwise, the model with the largest set of consistent points is adopted, or the algorithm fails.

IV. Ground and Non-ground Segmentation Based on Plane Fitting

In view of the huge data load during the transmission and processing of 3D lidar data, it is necessary to identify and segment the ground and non-ground in the navigable area so as to reduce the amount of data processing and improve the real-time performance of the algorithm. In this paper, the ground plane is fitted based on the plane fitting method. The distance between other points and the fitting plane is compared to the size of the set threshold, in order to distinguish whether the points are non-ground points or not. Generally speaking, since the ground points are unable to form a perfect plane, lidar is easy to generate noise when measuring distant targets, the single-plane model cannot represent the real ground. The point cloud is evenly divided into several segments along the driving direction of the vehicle, and the ground plane fitting algorithm is applied to each segment.

As shown in Tab. I, for each point cloud segment, the ground plane fitting algorithm first extracts a group of seed points with low height value. Then the algorithm uses these seed points to estimate the initial plane model of the ground. Each point in cloud segment P estimates the candidate plane according to the plane estimation model, and generates the orthogonal projection distance from the point to the candidate plane. By comparing such distance with a pre-defined threshold Th_{dist} value, we can determine determines whether the point belongs to the ground or not. If the point belongs to the ground, it is regarded as seed point and used to re-estimate the new plane by the plane estimation model. The process discussed above is repeated n times. Finally, the ground points of each point cloud segment generated by the algorithm can be connected and integrated into the whole ground plane.

TABLE I
GPF PSEUDOCODE

GPF pseudocode : Ground plane fitting algorithm

Symbol: P_g : Point on the ground plane
P_{ng} : points that are not ground plane
Initialization:
P:Enter point cloud
N_{iter}:number of iterations
N_{LPR}:points used to estimate the lowest representative point
Th_{seeds}: Initialize seed point threshold
Th_{dist}: point to plane distance threshold
Main loop:
P_g=*ExtractInitialSeeds* (P, N_{LPR}, Th_{seeds}) ;
for i =1:Niter do
model=EstimatePlane (P_g) ;
clear (P_g, P_{ng}) ;
for k =1: *P do*
if model (P_k) <*Thdist then;*
$P_g \leftarrow P_k$;
else
$P_{ng} \leftarrow P_k$;
end
end
end
Extract Initial Seeds :
P_{sorted} = *SortOnHeight* (P) ;
LPR = *Average* (P_{sorted}(*1*:N_{LPR})) ;
for k = 1:P do
if P_k.*height* < *LPR.height* + Th_{seeds} *then*
seeds $\leftarrow P_k$;
end
end
return(seeds);

The lowest representative point *LPR* is introduced to select the initial seed point, which is the average value of the lowest height value of N_{LRP} laser points. Once *LPR* is calculated, it is considered as the lowest height value point of point cloud P, and the points within the height threshold Th_{seeds} are used as the initial seed points for the plane estimation model. For plane estimation, a simple linear model is used as follows:

$$ax + by + cz + d = 0 \tag{12}$$

$$n^T x = -d \tag{13}$$

where n=$[a, b, c]^T$, x=$[x, y, z]^T$. The normal vector is obtained from the covariance matrix c and S of seed point set.

$$C = \sum_{i=1:|S|} (s_i - \hat{s})(s_i - \hat{s})^T \tag{14}$$

where $\hat{s} \in R^3$ is the mean of all S_i, $S_i \in$ S. Variance matrix C reflects the dispersion degree of seed points. After singular value decomposition, three singular values can be obtained, which represent the three directions of dispersion, respectively. In addition, as the model is basically a plane, the normal vector n represents the direction with the smallest

variance, which can be obtained by the singular vector. The singular value corresponding to the singular vector is the minimum.

V. Non-ground Laser Point Cloud Clustering Based on Improved DBSCAN Algorithm

A. The Basic Principle of DBSCAN Algorithm

To improve the DBSCAN algorithm, this paper proposes a new algorithm, of which the basic principles are as follows:

(1) Define the adaptive domain radius ε. Since the data density of the lidar point cloud and the distance of the laser beam scanning are linear, a fixed distance threshold is adequate to adapt to the point cloud clustering of the close distance point and the long distance point. Therefore, in this paper, the x value in the point cloud data is ten meters, the point cloud selects a fixed threshold ε_1 in the region of ten meter. The region within twenty meter and beyond ten meter selects another fixed threshold ε_2, and so on.

(2) Data set cube partitioning. Due to the traversal of the entire data set D and the search set S in the DBSCAN clustering process, the DBSCAN algorithm has the problems of long time and intensive memory. To solve these problems, the non-ground point cloud data in the new algorithm is divided into small cubes with a diagonal length of ε, which is conducive to finding each small cube and determining the core point. If the number of points in the cube is greater than M_{inPts}, the maximum distance between any two points in the cube is less than or equal to ε. Then all the points in the cube are directly marked as core points. Any point in the cube is greater than or equal to M_{inPts}.

B. The Improved DBSCAN Algorithm

(1) Determine the parameters (ε, M_{inPts}). As shown in Fig. 3, according to the maximum and minimum values in the x, y, and z directions on the point cloud map and the threshold ε_i, the length, width and height of the cubes are set. Each laser point corresponds to a small cube. For any point falling on the border of cube, an extra row or column is added to include it. For a laser spot with a longitudinal distance of ten meter, ε_1 is set to 0.8. For each subsequent ten meter, ε_i = 0.8+0.2(i-1) is set. M_{inPts} is 20.

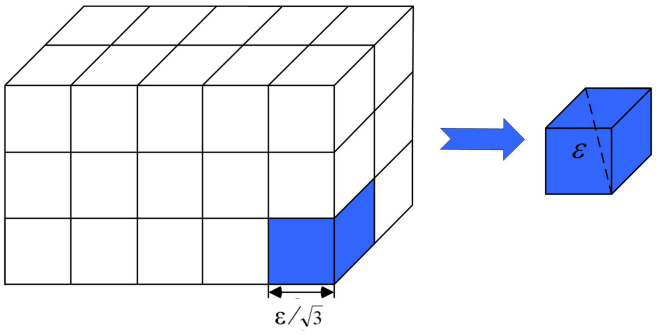


Fig. 3. Cube partitioning of data sets

(2) Traverse non-empty cubes. The number of points in the small cube is determined in turn. If the number of points in the small cube is greater than M_{inPts}, the algorithm marks the point in the cube as the core point. Otherwise, the algorithm calculates the distance between the point in the cube and the point in the adjacent cube. If the distance is less than or equal to ε, the point is set as a core point. The algorithm stops when all the small cubes are traversed.

(3) Combine the same points between adjacent cubes. The number of points in the small cube is determined in turn. If the distance between the two points from the two adjacent cubes is less than ε, the two points are grouped together. As shown in Fig. 4, assume that there are five points in each adjacent cube, M_{inPts}=4, and ten points of the two cubes are core points. Although point a and point b belong to two cubes, as the distance between them is less than ε, they still belong to the same cluster.

For the point cloud cluster after clustering, the algorithm calculates the centroid to determine the center of the obstacle, calculates the length and width of the point cloud cluster to determine the three-dimensional bounding box of the cloud cluster.

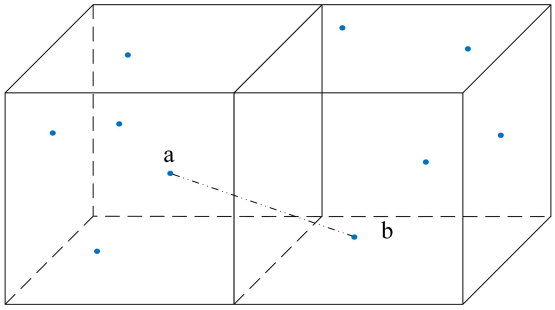


Fig. 4. Shift and Switch Schedules for the MEV

VI. Experimental Verification

The lidar-based obstacle detection experiment includes the following steps: radar calibration, removal of invalid point cloud data, road boundary extraction, ground and non-ground point cloud segmentation, non-ground point cloud according to the improved DBSCAN algorithm.

A. Calibration of the Radar

The test vehicle is shown in Fig.5. The vehicle is equipped with a 32-line lidar, two millimeter wave radars and a monocular camera. To verify the effectiveness of proposed method with 32-line lidar, the millimeter wave radars and a monocular camera are powered off during the test. The specific parameters of the lidar are shown in Tab. II.

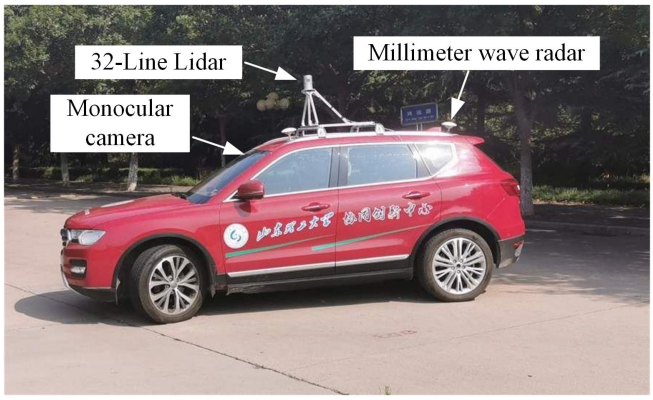


Fig. 5. The test vehicle

The length and width of the selected rectangle are one thousand millimeter and three hundred millimeters, respectively. The side length of the right triangle is one thousand millimeters. Since the pitch angle of lidar is calibrated in a right triangle, the triangle needs to be set in two positions. The longitudinal distance between the two positions is set to five meters.

TABLE II
VELODYNE HDL-32E PARAMETERS

INDEX	PARAMETE	UNIT
Precision	±2	cm
Harness	32	Line
Vertical distance	80-100	m
Vertical range	-30~10	
Update frequency	10	Hz
weight	1	Kg
Number of point clouds	700000	point /s
Horizontal range	360	degree
Power	12	w

Fig. 6 and Fig. 7 show the visualization of the lidar on the Rviz interface of the ROS robot operating system with two calibration plates. After calibration, the lidar roll angle β' is 1.09°, the pitch angle α is 3.14°. Consequently, translation vector T is $[0, 0, 1588.9]^T$, and Z is $[0, 0, 6]^T$. Finally, the conversion relationship is:

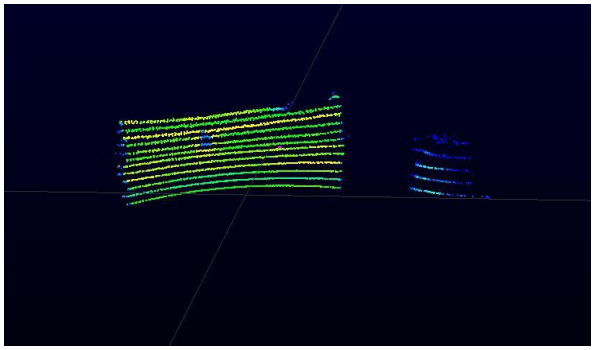


Fig. 6. Rectangular calibration plate

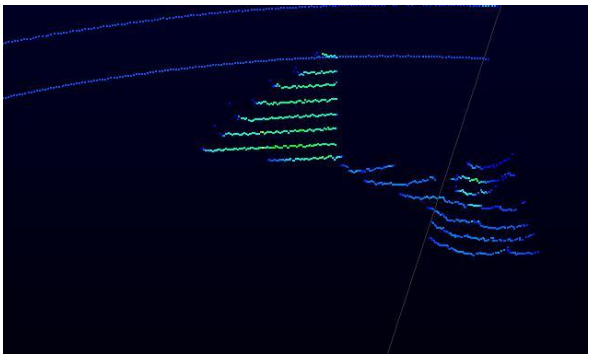


Fig. 7. Isosceles right triangle calibration plate

$$\begin{bmatrix} P_x' \\ P_y' \\ P_z' \end{bmatrix} = \begin{bmatrix} 0.9404 & 0 & 0.3400 \\ 0.2860 & 0.5407 & 0.7911 \\ 0.1839 & -0.8412 & 0.5085 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1594.9 \end{bmatrix} \quad (15)$$

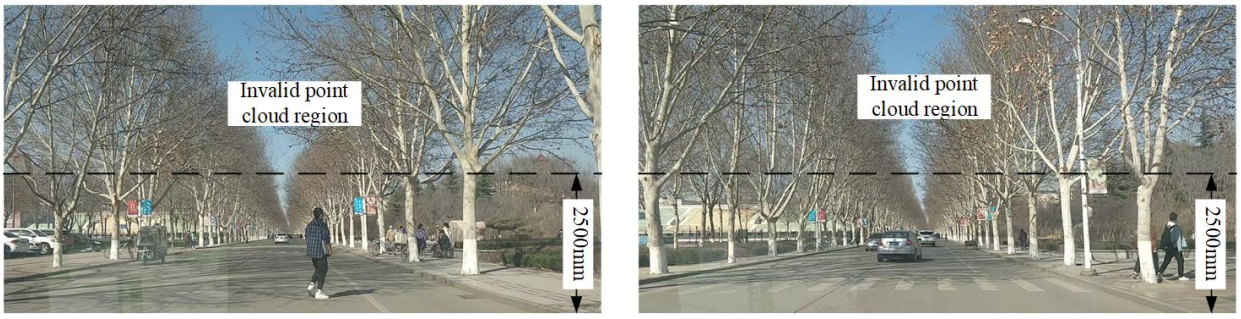
B. Removal of Invalid Point Cloud Data

The lidar used in this paper generates seven hundred thousand points per second. During the process of data acquisition, there is bound to be a point cloud that is ineffective for obstacle detection. As shown in Fig. 8, as the installation height of the lidar is two thousand and two hundred millimeters, which is lower than the point cloud whose height exceeds two thousand and five hundred millimeters.

C. Road Boundary Extraction

The redundant point cloud data on both sides of the road is removed by using a constraint. Specifically, the distance between the center line of the vehicle body and the boundary points on both sides of the road is smaller than the road width. The driving scene of the test vehicle is two ways and two lanes. The two-way and two-lane width is nine meters. The road width error is set to 0.2 meter. The 3D point cloud is projected onto the X-O-Y plane. Furthermore, grid with twenty centimeters in width and twenty centimeters in length is used to generate the grid map. The height feature of point cloud is extracted on the grid map. Test result shows that the z_{max} of the road edge is the height of the roadside stone. The threshold D_2 is tested to be ten centimeters. Since the lowest point needs to be greater than the highest slope of the ground, D_1 is set to five centimeters.

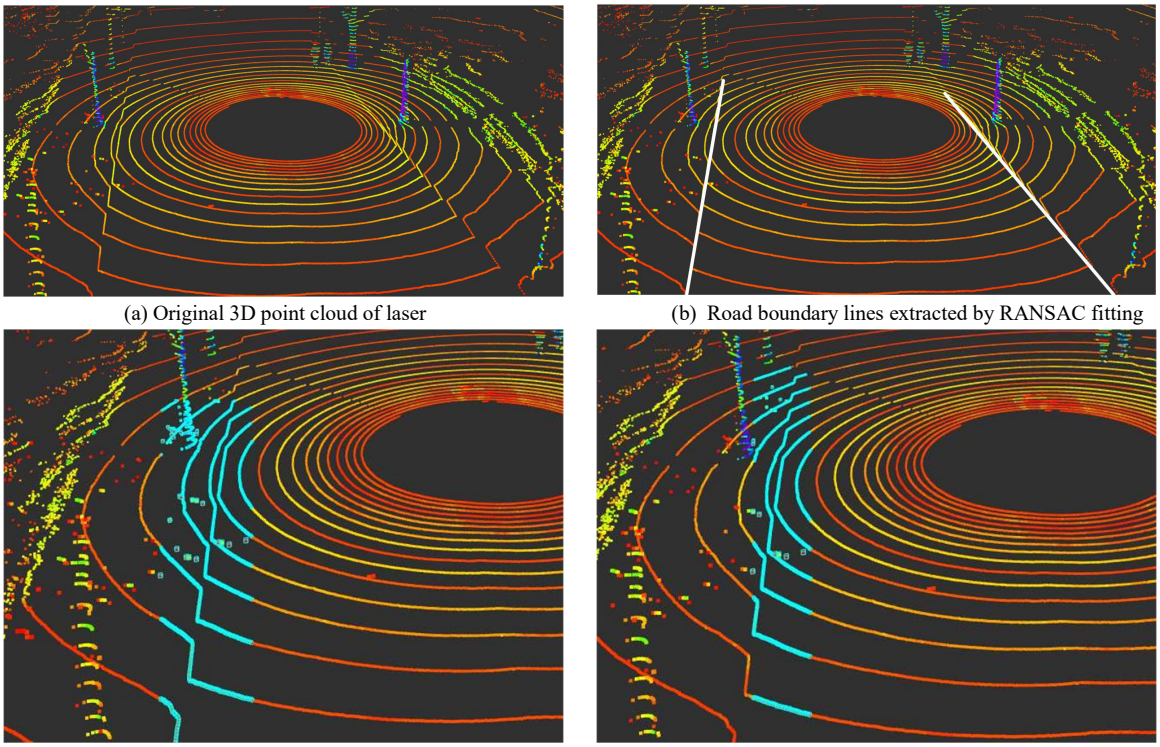
Figs. 9(a) shows the top view of the original 3D point cloud data collected by the lidar. The 3D point cloud data is visualized under the ROS system. The white point in figure 9(c) diagram is the point that satisfies the height feature. The white point in figure 9(d) is the point that satisfies the smoothness feature. In the figure 9(b), the white line segment is the road boundary line after the height feature and the smoothness feature are simultaneously extracted and fitted by the RANSAC algorithm.



(a) Original area of laser point cloud

(b)Original area of laser point cloud

Fig. 8. Reflection of the point cloud area that the lidar needs to filter



(a) Original 3D point cloud of laser (b) Road boundary lines extracted by RANSAC fitting

(c) Height characteristic point (d) Smooth feature point

Fig. 9. Lidar point cloud data road boundary extraction

D. Segmentation of Ground and Non-ground Point Clouds

During test, the number of segment divided by the point cloud is 3(N_{segs}=3), the number of iteration is 3(N_{iter}=3), the number of candidate lowest representative point is 20(N_{LPR}=20), the threshold of seed point is 0.4 meter (Th_{reseds}=0.4m), the distance threshold from the fitting plane is 0.2 meter (Th_{dist}=0.2. m). As shown in figure 10, compared with the ray ground filter (RGF) method, the ground plane filter (GPF) method shows better real-time performance.

Fig. 11 shows the comparison of the non-ground effect of GPF and RGF. As shown in figure 11(a), the original point cloud data collected by lidar has ground and non-ground point clouds. Figure 11(b) shows the non-ground effect of RGF method segmentation, where ground and non-ground point clouds still exist. In figure 11(c), owing to GPF method, the ground point cloud separates from the non-ground point cloud.

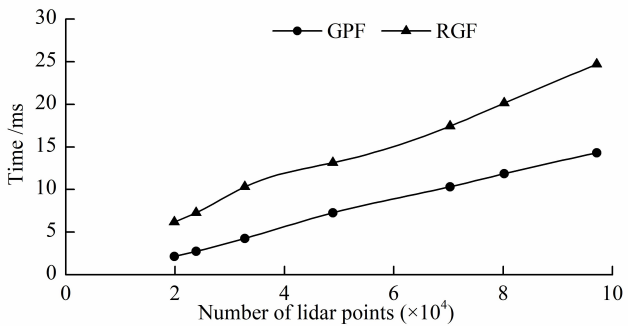


Fig. 10. Time performance of GPF and RGF method

E. Non-ground point cloud based on improved DBSCAN algorithm clustering

Fig. 12 shows the real-time performance of the improved DBSCAN and the traditional DBSCAN clustering algorithms. With the increase of the number of three-dimensional points, the algorithm based on DBSCAN consumes exponentially. The improved DBSCAN algorithm shows perfect real-time performance compared with DBSCAN method.

Figs. 13 (a) and (b) show the bounding box renderings generated by two clustering algorithms. As the improved DBSCAN algorithm takes the change of point cloud density into account, its clustering effect is much better. However, as the original DBSCAN algorithm clusters the whole clustering process with a given radius without taking the change in the density of the obstacle into consideration, its segmentation effect is worse than that of the improved DBSCAN.

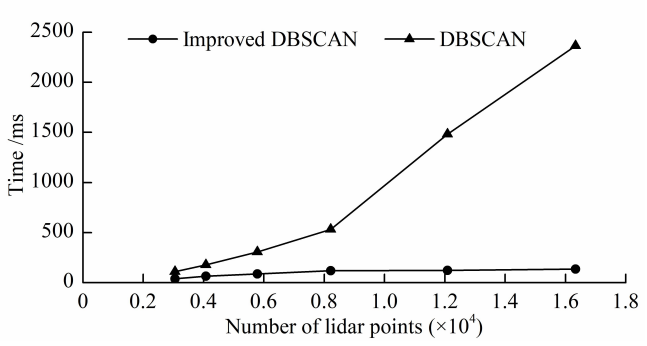
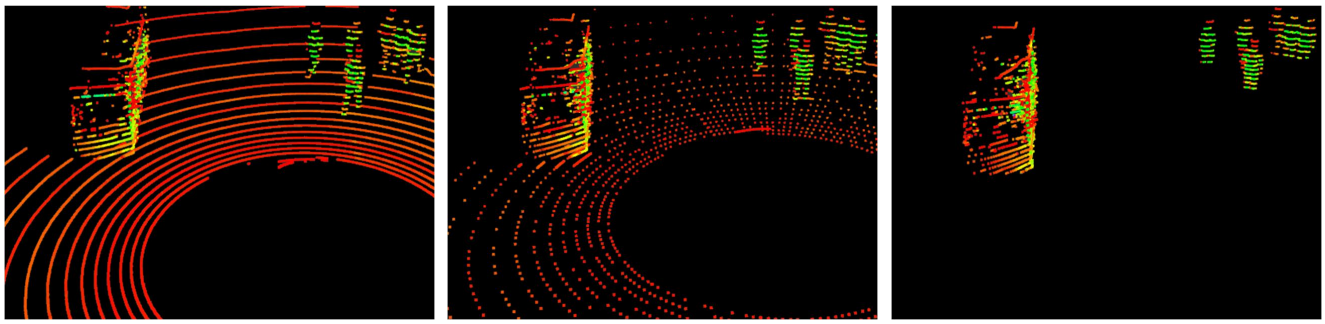


Fig. 12. Real-time performance of DBSCAN and RANSAC algorithms

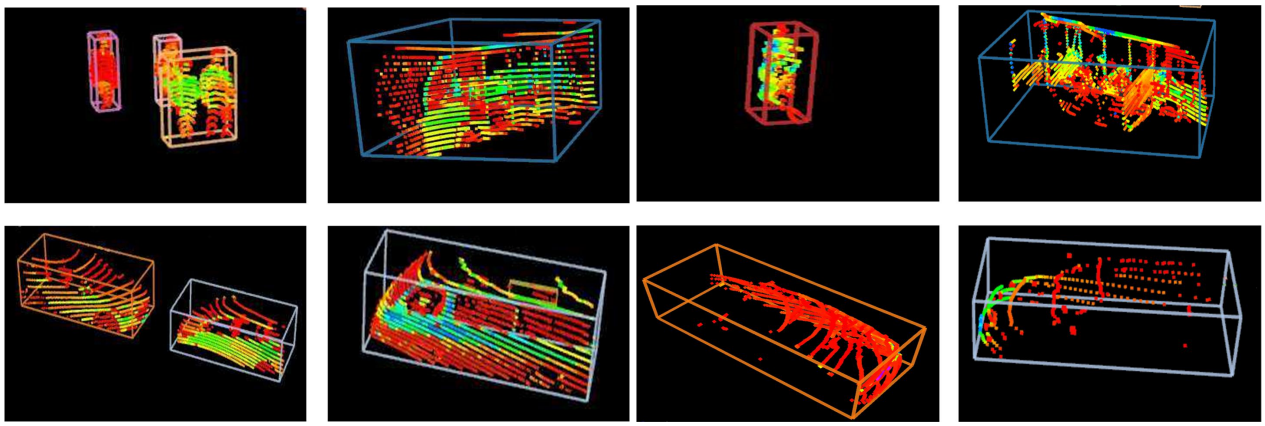
F. Lidar-based classification detection of obstacles

The data values returned by the lidar are obtained by clustering the width and height of the projected rectangle of the bounding box formed on the Y-O-Z plane. The aspect ratio of the car profile in the Y-O-Z plane is smaller than that of the pedestrian profile. In order to distinguish pedestrians from vehicles quickly and effectively, the aspect ratio threshold (λ) is set to 2.2. If λ is smaller than the threshold, the bounding box is classified as vehicle. Otherwise, it is classified as a pedestrian.



(a) Original point cloud (b) Ray ground filtering method (c) Ground plane fitting method

Fig. 11. Non-ground segmentation effect between RGF and GPF



(a) Improved DBSCAN clustering and generation of Bounding Box (b) DBSCAN clustering and generating Bounding Box

Fig. 13. Segmentation effect of the improved DBSCAN and the original RANSAC algorithm

Table 3 shows the classification results of obstacles by using lidar alone. As the number of obstacles in the test scene is less than the number of pedestrian targets, and the point cloud density is higher than that of the pedestrian. The detection accuracy of the obstacle vehicle is 5.02% higher than that of pedestrian. However, single type obstacle detection still has high false detection rate. Consequently, multi-type sensors are required to improve the accuracy of obstacle detection and classification.

TABLE III
LIDAR OBSTACLE DETECTION RESULTS

Obstacle type	Total number of targets	Checkout number	Number of false detection
Vehicle	893	741	152
Pedestrian	1257	980	277
Obstacle type	Detection rate	False detection rate	
Vehicle	82.98%	17.02%	
Pedestrian	77.96%	22.04%	

VII. CONCLUSION

Based on a 3D 32-line lidar, this paper studies radar data calibration, invalid point cloud data removal, road boundary extraction, ground and non-ground point cloud segmentation, and non-ground point cloud clustering. Following conclusions are obtained:

(1) Compared with the RGF method, the GPF method shows better real-time performance and higher accuracy in segmentation of the ground and non-ground point cloud.

(2) Compared with the traditional DBSCAN clustering algorithm, the improved DBSCAN clustering algorithm proposed in this paper considers the change of point cloud density caused by the distance between the obstacle and the lidar. By defining the adaptive domain radius and data set cube segmentation, the new method has significantly improved real-time performance and accuracy of clustering.

(3) High false detection rate is still a problem in lidar-based obstacle detection and classification. It is necessary to cooperate with machine vision and millimeter wave radar to improve the accuracy in obstacle detection and classification. On the basis of this study, the future work will focus on multi-sensor obstacle detection and data fusion methods based on lidar, machine vision and millimeter wave radar.

REFERENCES

[1] D. Elliott, W. Keen, L. Miao, "Recent advances in connected and automated vehicles," *J. Traffic Transp. Eng. (Engl. Ed.)*, vol.6, no.2, pp. 109-131, 2019.

[2] A. Sumalee, H.W. Ho, "Smarter and more connected: Future intelligent transportation system," *IATSS Res.*, vol.42, no.2, pp. 67-71, 2018.

[3] P. Penmetsa, E. K. Adanu, D. Wood, et al., "Perceptions and expectations of autonomous vehicles-A snapshot of vulnerable road user opinion," *Technol. Forecast. Soc. Change*, vol.143, pp. 9-13, 2019.

[4] L. Chen, D. Qin, X. Xu, et al., "A path and velocity planning method for lane changing collision avoidance of intelligent vehicle based on cubic 3-D Bezier curve," *Adv. Eng. Software*, vol.132, pp. 65-73, 2019.

[5] N. Mahajan, A. Hegyi, P. Serge, "Hoogendoorn, Bart van Arem. Design analysis of a decentralized equilibrium-routing strategy for intelligent vehicles," *Transp. Res. Part C Emerg. Technol.*, vol. 103, pp. 308-227, 2019.

[6] X. Sun, Y. Cai, X. Xu, et al., "Optimal control of intelligent vehicle longitudinal dynamics via hybrid model predictive control," *Rob. Autom. Syst.*, vol. 112, pp. 190-200, 2019.

[7] Y. Zhao, F. Ding, J. Li, et al., "The intelligent obstacle sensing and recognizing method based on D-S evidence theory for UGV," *Future Gener. Comput. Syst.*, vol. 97, pp. 21-29, 2019.
[8] J. V. Brummelen, M. O'Brien, D. Gruyer, et al., "Autonomous vehicle perception: The technology of today and tomorrow," *Transp. Res. Part C Emerg. Technol.*, vol. 48, pp. 384-406, 2018.
[9] P. V. Manivannan, P. Ramakanth, "Vision Based Intelligent Vehicle Steering Control Using Single Camera for Automated Highway System," *Procedia Comput. Sci.*, vol. 133, pp. 839-846, 2018.
[10] X. Pei, Zh. Liu, G. Ma, et al., "Safe distance model and obstacle detection algorithms for a collision warning and collision avoidance system,". *J. Automotive Safety and Energy*, vol.3, no.1, pp. 26-33, 2012.
[11] A. Asvadi, C. Premebida, P. Peixoto, et al., "3D Lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes," *Rob. Autom. Syst.*, vol.83, pp. 299-311, 2016.
[12] L. Wang, Y. Zhang, J. Wang, "Map-Based Localization Method for Autonomous Vehicles Using 3D-LIDAR," *IFAC-Papers Online*, vol.50, no.1, pp.276-281, 2017.
[13] J. Liu, "Research on Key Technologies in Unmanned Vehicle Driving Environment Modelling Based on 3D Lidar," Ph.D. dissertation, Veh. Eng., University of Science and Technology of China, Hefei, China, 2016.
[14] X. Wang, Y. Jin, Z. Ding, "A path planning algorithm of raster maps based on artificial potential field," *2015 Chinese Automation Congress (CAC)*, Wuhan, China, pp. 627-632. 2015.
[15] J.M. Armingol, J. Alfonso, N. Aliane, et al., "Intelligent Vehicles: Enabling Technologies and Future Developments," Oxford, United Kingdom: Butterworth-Heinemann, pp. 20-32. 2018.
[16] T. Apinya, M. Songrit, "U-DBSCAN: A density-based clustering algorithm for uncertain objects," *IEEE 26th Int. Conf. on Data Engi. Workshops (ICDEW 2010)*, Long Beach, CA, USA, pp.136-143. 2010.
[17] A. Al-Kaff, D. Martín, F. García, et al., "Survey of computer vision algorithms and applications for unmanned aerial vehicles," *Expert Sys. Appl.*, vol. 92, pp. 447-463, 2018.
[18] Y. Shang, and J. Huang, "Fixed-Time Stabilization of Spatial Constrained Wheeled Mobile Robot via Nonlinear Mapping," *IAENG Int. J. Appl. Math.*, vol. 50, no.4, pp. 791-796, 2020.