IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 13, 2019, accepted June 26, 2019, date of publication July 2, 2019, date of current version July 16, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2926424*

# Research on 3D Point Cloud De-Distortion Algorithm and Its Application on Euclidean Clustering

**LONG WEN, LEI HE, AND ZHENHAI GAO**

State Key Laboratory of Automotive Simulation and Control, Jilin University, Changchun 130022, China

Corresponding author: Lei He (jlu_helei@126.com)

This work was supported in part by the complex road environment system perception and target tracking technology, under Grant 2017YFB0102601.

**ABSTRACT** The 3D point cloud data collected by 3D lidars have become a significant resource for autonomous vehicles to acquire road information. But these data tend to be inaccurate due to the turning or moving of autonomous vehicles while the lidar is working. Moreover, the traditional Euclidean clustering algorithm often causes false detection in the vicinity or missed detection in the distance if the Euclidean distance threshold is not selected properly. This paper proposes a method which contains three main steps to solve these problems: First, a de-distortion algorithm is applied to diminish the influence of distortion caused by the moving and turning of lidar; second, optimize the structure of the Euclidean clustering algorithm to make it run faster; and third, we applied an adaptive threshold of the Euclidean distance in the improved algorithm, so the improved algorithm is able to detect the relatively small objects in the distance while it can also detect the objects nearby without misjudgment. These features are all confirmed by the results of vehicle experiments which shows that with the help of proposed method, the distortion of point cloud reduced, the detection distance has increased by nearly 8 meters, and the clustering speed has also increased by 15%.

**INDEX TERMS** Autonomous vehicle, 3D-lidar, point cloud, Euclidean clustering, object detection.

## I. INTRODUCTION

The The field of autonomous driving is of great interest to researchers [1] and environment perception is one of the most important part of it [2]. 3D-lidars are widely equipped on autonomous vehicles to acquire road information [3] because of its wide detecting range, high degree of accuracy and Strong anti-interference ability [4]. In both [5] and [6], the vehicles that attend DARPA Challenge are all equipped with 3D-lidar which indicates that 3D-lidar is especially suitable for obtaining road information under complex conditions and has broad research prospects.

But, when the 3D lidar rotates to collect the point cloud data of surrounding environment, there is an inter-frame motion between the first point and the last point in each frame of point cloud due to the movement of autonomous vehicles. That is to say the points in the same frame of point cloud do not exist in the same Cartesian coordinate system, which inevitably leads to the distortion of point cloud. So, a de-distortion algorithm is needed to diminish the influence caused by this problem and make the point cloud data collected by lidars be able to reflect real environmental information.

object detection is an essential part of environment perception [7]. Methods for detecting objects based on 3D lidar mainly include: DBSCAN clustering algorithm, K-means clustering algorithm, Euclidean clustering algorithm and cluster analysis using depth image. The DBSCAN algorithm is a density-based clustering algorithm that can quickly cluster any shapes of point cloud data [8]. However, due to the large computation consumption of this method, when the amount of point cloud data is too large, the requirement for processor is very demanding. The K-means clustering algorithm can still ensure its accuracy and efficiency when the amount of data is large [9]. However, this method requires manual input of the number of clusters $K$. Since the

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

driving environment of the vehicle is extremely complicated, the calculating speed of $K$ is difficult to achieve the real-time requirement of autonomous vehicles. Cluster analysis using depth images requires a large amount of data [10]. Even multiple 32-line 3D lidars are needed during the experiment, the cost is too high. The Euclidean clustering algorithm is a simple and efficient algorithm, which has strong point cloud segmentation capability and does not require data volume [11]. Although a fixed distance threshold can cause problems when dealing with objects at different distances, but this defect can be improved. Therefore, the Euclidean clustering algorithm should be more suitable for object detection of 3D lidar.

The traditional Euclidean clustering algorithm can search the neighboring points for each laser point through the established topological relationship [12], and calculate the Euclidean distance between each point and their neighboring points. The points with the smallest distance are classified into one cluster, and then the Euclidean distance between the new clusters is calculated. Repeat the process above until the Euclidean distance between any two clusters is bigger than the threshold set in advance or the number of clusters is less than the number set in advance. So far, the Euclidean clustering is completed [13]. The traditional Euclidean clustering algorithm puts high requirements on the selection of the distance threshold [14]. If the distance threshold is not properly selected, it often causes false detection in the vicinity or missed detection in the distance when processing the point cloud with uneven density. This is because the point cloud is dense in the vicinity and sparse in the distance while the distance threshold is constant all the time. So traditional Euclidean clustering algorithm is always lack of detection capability for distant objects.

For the problem of point cloud distortion this paper uses two steps to reduce the distortion, including the extraction of inter-frame motion and the reduction of point cloud distortion. As for the problem of object detection, this paper introduces three processes to improve the ability of processing uneven point cloud, including candidate points extraction and filtering, optimization of the neighborhood searching method and application of adaptive Euclidean distance threshold.

The main contribution of this paper is to propose a point cloud de-distortion algorithm and improve the traditional Euclidean clustering algorithm. The point cloud de-distortion algorithm can effectively eliminate the point cloud distortion caused by the movement of the lidar during the acquisition process. And the algorithm has little demand for computer performance, which provides a possible solution for the point cloud de-distortion problem of autonomous vehicles. The improved Euclidean clustering algorithm effectively alleviates the shortcomings of traditional Euclidean clustering algorithms when dealing with multiple targets at different distances by introducing a adaptive distance threshold. It also provides a viable solution for handling uneven point clouds.

## II. RELATED WORKS

### A. RELATED WORKS ON THE EXTRACTION OF INTER-FRAME MOTION

For the extraction of inter-frame motion, we first need to extract the feature points, then the inter-frame motion can be easily calculated according to the coordinate value of feature points in two adjacent frame of point cloud. Lowe D G [15] proposed the SIFT algorithm which can be used to find feature descriptors that are less likely to be affected by the movement of lidar by using Gaussian differential function and Taylor series. Then feature points can be found through the feature descriptors. But this kind of method is only suitable for images or point clouds that are uniform. When the point cloud data become sparse and uneven, it is quite hard to find these feature descriptors which means it may take far more time to get a wrong result. This obviously does not meet the requirements on the autonomous vehicles. Tong LJ. [16], Remeli NH. [17] and Redondo-Cabrera C. [18] uses SURF operator to extract the matched point pairs of the projection image by using the SURF algorithm to detect the feature points of an image and its direction. In [19], Lin Hui extracted feature lines and feature planes instead of feature descriptor by representing the point cloud abstractly with simple lines and planes. But the accuracy of the extraction cannot be guaranteed which means the result is not reliable.

And all these algorithm mentioned above shared a common drawback that the extraction of feature points is too complex. So the processing cycle will be quiet long which is unable to meet the real-time requirements.

### B. RELATED WORKS ON THE PROCESSION OF UNEVEN POINT CLOUD

In order to process the uneven point cloud, Francisco de A.T. and de Carvalho [20] proposed an adaptive dynamic clustering algorithm which can change the threshold according to different shapes and sizes of clusters. But it does not work well in the processing of point cloud which is collected by lidars on autonomous vehicles because the amount of data is too large and it is hard to recognize all of them. In [21]–[23], a data-partitioning-based DBSCAN algorithm is proposed which divide the entire data space into a plurality of smaller parts according to the spatial distribution characteristics of the data. Then these parts are respectively clustered, and finally the local clusters are merged. These two algorithms provide a great inspiration of processing point clouds with uneven density. But the number of parts which is divided according to the spatial distribution is limited which means these algorithms cannot deal with the enormous data collected by 3D lidar.

### C. MAIN CONTENT OF THIS PAPER

This paper introduces a point cloud de-distortion algorithm and an improved Euclidean clustering algorithm for autonomous vehicles. The point cloud de-distortion algorithm get the inter-frame motion by processing the parameters

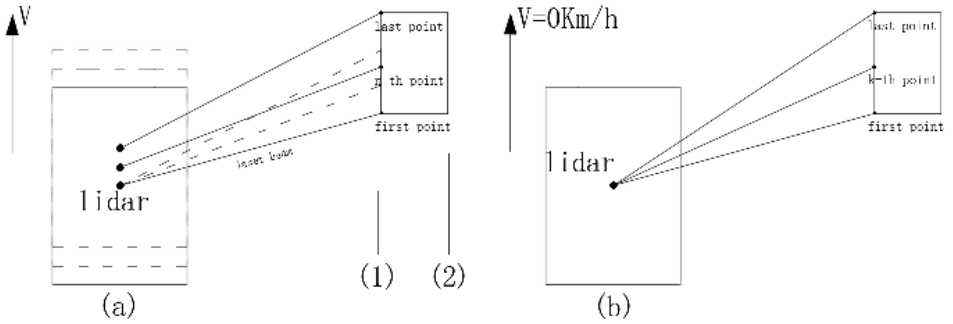



**FIGURE 1.** Comparison of object detected while driving straightly and still.

which are gotten from the sensors on the autonomous vehicle and use it to fix the distorted point cloud by the coordinate transformation. The improved Euclidean clustering algorithm has an adaptive Euclidean distance threshold which can change its value according to the distance between the objects and vehicle. These two methods can effectively solve the problem that the Euclidean clustering algorithm has low success rate of object detection due to uneven point cloud density and point cloud data distortion.

This paper is organized as follow: Section 2 presents the process of reducing point cloud distortion. Section 3 presents the decision of Neighborhood search algorithm and how the Euclidean clustering algorithm is improved. In section 4, it is presented the process of experiment and the conclusion after analyzing the experimental data.

## III. POINT CLOUD DE-DISTORTION ALGORITHM

### A. REASONS FOR POINT CLOUD DISTORTION

The figures below illustrates how the distortion occurs during the acquisition period. As can be seen from the figure 1 the autonomous vehicle is moving in a straight line and the object is on the right of the vehicle. The vehicle moved ahead a little when the lidar is collecting the $n$-th ($k > n$) point of the object and the vehicle moved further when it comes to the last point of the object. It is obvious that the length of the object detected while moving (line 1 in figure 1) become smaller compared to the length detected when the vehicle is still (line 2 in figure 1).

In figure 2, the vehicle is moving in a straight line and the object is in front of the vehicle. The distance between the object and lidar become smaller while lidar is collecting the data points on the object. That is to say, for every point collected by lidar, the distance between the obstacle and the vehicle is different. This makes the shape of the object distorted (line 1 and 2 in figure 2).

From figure 1 and figure 2 we can know that for a straight forward autonomous vehicle, the motion of lidar can lead to the distortion of width and length of the object.

In figure 3, the vehicle is turning and the object is in front of the vehicle. The figure shows that the interval between points detected on the object become larger ($k > n$) due to the

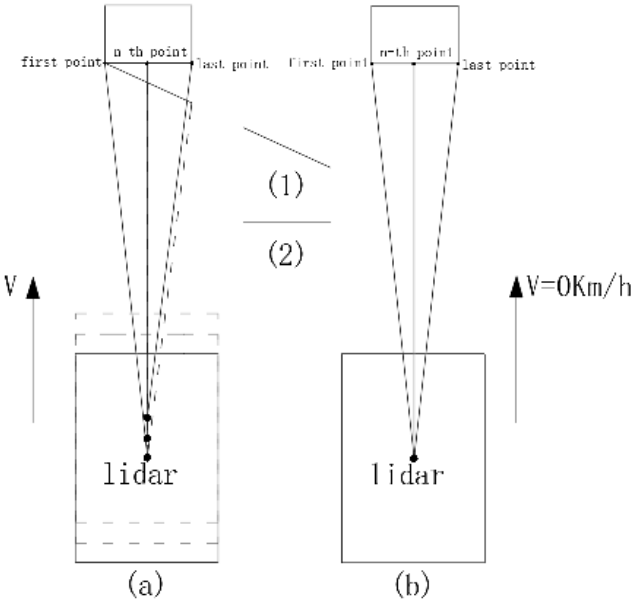



**FIGURE 2.** Comparison of object detected while driving straightly and still.

turning of vehicle which makes the shape of object changed. It will become smaller in size while the vehicle is turning right and will become larger while turning left. This is because the lidar collects point cloud data clockwise.

From figure 3 we can know that when the vehicle is turning both the shape and size of the detected object will be distorted.

It is worth noting that the distortions in pictures have been artificially magnified in order to illustrate easily, the real distortion is not so exaggerated. It is relatively small on the contrary.

Another thing need to be explained is that the vertical speed of the vehicle is 0 or very small for most of the time, so the distortion in Z-axis direction is quite small which does not need to be processed. So the de-distortion algorithm is mainly about the X and Y coordinates.

### B. PROCESSES OF DE-DISTORTION ALGORITHM

In order to solve the problem of point cloud distortion, the coordinates of all points in a frame of point cloud data are all transformed into the coordinate system of the first point in

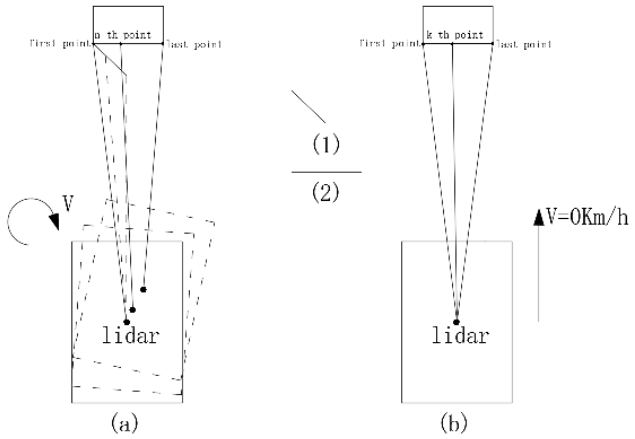



**FIGURE 3.** Comparison of object detected while turning and still.

the frame. Before the transformation, the scanning frequency of lidar need to be set.

The scanning frequency of the 16-line lidar can vary from 5 to 20 Hz. Although the low frequency reduces the computational pressure of the hardware device, but the system cannot meet the real-time requirements of autonomous vehicle if the frequency is too low. High frequency means higher real-time performance, but due to frequency limitations of other devices on the vehicle, such as inertial navigation system, the algorithm cannot deal with every frame of data collected by lidar because the inertial navigation system cannot provide vehicle speed, heading angle and other data in time. This make the overall effect of the algorithm worse. In the meantime, high scanning frequency means low angular accuracy of lidar, and the number data points in one frame of data will decrease. This will also affect the effect of the clustering algorithm. So the scanning frequency should not be too small or too large. Considering the fact that the frequency of inertial navigation system is 12 Hz, the scanning frequency of lidar is set to 10 Hz.

After the decision of scanning frequency, there are some parameters need to be collected:

The heading angle of the first point in a frame $\theta_F$, the heading angle of the last point in a frame $\theta_L$, the heading angle $\theta$ which is positive when the vehicle turns right and is negative when the vehicle turns left, the velocity of autonomous vehicle $V$. The $\theta_F$ and $\theta_L$ is the heading angle of autonomous vehicle when the lidar is collecting the first and the last point in a frame of point cloud. The heading angles and velocity can be collected from the inertial navigation system on autonomous vehicle. Since the acquisition period of the lidar is 0.1 second, the motion of autonomous vehicle can be approximated as a uniform motion in one acquisition cycle. Because the 3D lidar uses multiple beams of laser to collect environmental information simultaneously, so this paper presents the processing steps of point cloud data collected by one beam of laser for example. The processing steps of point cloud data collected by other beams of laser are totally the same.

The transformation steps are listed below:

First, the ground coordinate system is set as H(XOY), the vehicle coordinate system is set as U(X'O'Y'), the coordinates of the $I$-th point in the ground coordinate system is $(X_i^H, Y_i^H)$, the coordinates of the $I$-th point in the vehicle coordinate system is $(X_i^U, Y_i^U)$.

Second, transform the coordinates of all the points in a frame point cloud from vehicle coordinate system into ground coordinate system by (1).

$$\begin{cases} X_i^H = X_i^U \cdot \cos\theta_I + Y_i^U \cdot \sin\theta_I \\ Y_i^H = Y_i^U \cdot \cos\theta_I - X_i^U \cdot \sin\theta_I \end{cases} \tag{1}$$

$\theta_I$ is the heading angle of autonomous vehicle when the lidar is collecting the $I$-th point. Because the interval for lidar to collect each point is the same, so $\theta_I$ can be calculated by (2).

$$\begin{aligned} \theta_I &= \theta_F + \Delta\theta_I \\ \Delta\theta_I &= \frac{iR(\theta_L - \theta_F)}{360} \end{aligned} \tag{2}$$

Parameter $R$ is the angular resolution of lidar. Parameter $I$ is the ordinal position of points.For example, if the point is the 10th point in the frame, the value of 'I' is 10. $\Delta\theta_I$ is the change of heading angle during one acquisition period.

Third, after the transformation, the translation of all the points can be achieved by (3).

$$\begin{cases} X_i^{HF} = X_i^H + M \cdot \sin\beta \\ Y_i^{HF} = Y_i^H + M \cdot \cos\beta \end{cases} \tag{3}$$

$X_i^{HF}$ and $Y_i^{HF}$ are the coordinates of the $I$-th point that have finished the translation. $\beta$ is the Approximate heading angle and $M$ is the inter-frame motion, they can be calculated by (4) and (5).

$$\beta = \theta_F + \frac{\theta_L - \theta_F}{2} \tag{4}$$

$$M = \frac{0.1iRV}{360} \tag{5}$$

fourth, after reduce the distortion caused by the motion of lidar, the distortion caused by the rotation of lidar also need to be reduced by (6).

$$\begin{cases} X_i^{HR} = X_i^{HF} \cdot \cos(\theta_i - \theta_F) - Y_i^{HF} \cdot \sin(\theta_i - \theta_F) \\ Y_i^{HR} = X_i^{HF} \cdot \sin(\theta_i - \theta_F) + Y_i^{HF} \cdot \cos(\theta_i - \theta_F) \end{cases} \tag{6}$$

$X_i^{HR}$ and $Y_i^{HR}$ are the coordinates of the $I$-th point that have finished the rotation.

Finally, the coordinates of all the points need to be transformed back into vehicle coordinate system to finish the algorithm by (7).

$$\begin{cases} X_i^{UF} = X_i^{HR} \cdot \cos\theta_I - Y_i^U \cdot \sin\theta_I \\ Y_i^{UF} = X_i^{HR} \cdot \sin\theta_I + Y_i^U \cdot \cos\theta_I \end{cases} \tag{7}$$

$X_i^{UF}$ and $Y_i^{UF}$ are the coordinates of the I-th point that have finished the whole de-distortion algorithm.

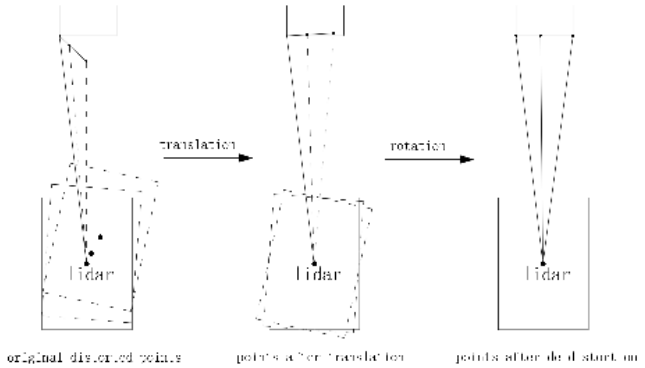



**FIGURE 4.** Processes of de-distortion algorithm.

The figure 4 below illustrates the whole process of point cloud de-distortion algorithm.

It can be seen clearly from the figure that the distortions of point cloud are consisted of two different parts. The translation operation can effectively reduce the distortion caused by the horizontal movement of lidar. And the distortion caused by the rotation of lidar can be reduced by rotation operation. Only after completing these two operations, the de-distortion algorithm is finished.

### C. ANALYSIS OF DE-DISTORTION ALGORITHM RESULT

The result of the de-distortion algorithm needs to be tested to determine that the algorithm is correctly completed, and its correctness can be determined by the following method.

First, we need to know the driving state of the vehicle. The driving state of the vehicle can be judged by the heading angle and the speed of the vehicle provided by the inertial navigation system. The vehicle is turning right if $\theta > 0$ and it is turning left if $\theta < 0$. The vehicle is moving ahead if $V > 0$ and it is moving backwards if $V < 0$.

Second, the specific distortion of the point cloud can be known from the driving state of the vehicle. If the length of plane which is vertical to the direction of vehicle is called length, and the length of the parallel plane is called width. Then the width of the object point cloud becomes smaller when the vehicle is moving forward and becomes larger when the vehicle is moving backwards. The length will become larger when the vehicle is turning left and become smaller when the vehicle is turning right. As for the length and width of the object, it can be judged by the coordinates of the point cloud.

Third, check the width and length of the point cloud after de-distortion to see whether the algorithm has reduced the distortion. If the distortion is not reduced but increases on the contrary. We can ensure that the result of this de-distortion is wrong. In this case the de-distortion operation of this frame of point cloud will be abandoned, the algorithm will only work when it can be valid.

But when the information provided by the inertial navigation system goes wrong, such as a wrong heading angle or speed, it will also affect the de-distortion effect, and the algorithm cannot judge the correctness of the result by itself. This problem can only be avoided by using a more reliable inertial navigation system.

After the test of algorithm results, the processed point cloud can be used for clustering algorithm.

## IV. POINT CLOUD SEGMENTATION AND EUCLIDEAN CLUSTERING

### A. POINT CLOUD SEGMENTATION

The point cloud data needs to be pre-processed before clustering [24]. In addition to the above-mentioned point cloud de-distortion algorithm, the point cloud should be segmented according to the measurement range of lidar. Since the lidar is mounted on the roof of vehicle, there is a certain range of detecting area blocked by the vehicle, and the data in this range does not need to be processed. So they should be removed to improve calculating efficiency.

In order to further reduce the amount of calculation, the ground should also be removed from the point cloud data [25]. Urban roads are relatively flat, so this paper segment the ground from point cloud data by the methods listed below:

The points that height coordinate Z is within the interval $[-h-\sigma, -h+\sigma]$ should be removed as ground data. Parameter $h$ is the installation height of lidar, the value of $\sigma$ is related to road roughness and detection distance, the initial value usually choose 0.05m and decreases as the detection distance increases.

After filtering out the ground points, the remain point cloud can be used for Euclidean clustering.

### B. DISTANCE INDICATOR

In order to finish the Euclidean clustering, a distance threshold $S_d$ needs to be defined. In this paper the Euclidean distance [26] is used to describe the distance between two different clusters. The Euclidean distance between the $I$-th and $j$-th cluster which contains $n$ points can be calculated [27] by (8).

$$d_{ij} = \sqrt{\sum_{k=1}^{n}(x_{ik} - y_{jk})^2} \tag{8}$$

### C. METHODS OF NEIGHBORHOOD SEARCH

As mentioned before, the point cloud collected by 3D lidar contains large amount of data which has a uneven distribution. So the efficiency of searching the target point is quite low. In order to meet the real-time requirements of autonomous vehicles, a topological relationship between discrete points should be established for the fast lookup of points or clusters within neighborhood [28]. Currently, there are two main methods of establishing topological relationship between discrete points: Octree and KD-tree.

The Octree method is to perform voxel splitting on spatial entities so that each individual elements have the same space-time complexity. By performing voxel splitting on geometric

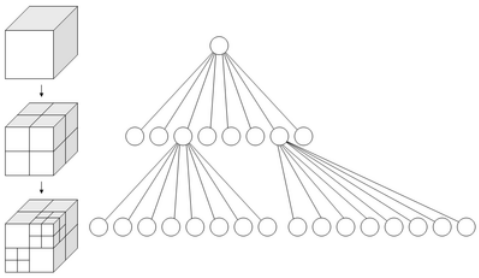


**FIGURE 5.** The structure of octree.

objects with a size of $2n \times 2n \times 2n$ in a cyclic recursive method, a direction map with root node can be established [29]. In Octree, a single leaf node consists of voxels of the same attribute, otherwise further split is needed. figure 5 illustrate the principle of octree method.

KD-tree is a multi-dimensional structure of binary tree, which can organize and store the multi-dimensional data. The KD-tree divides a data space into multiple disjoint subspaces, each node divides its data space into two subspaces. If the data contained in the node is less than the lower limit set in advance, this node will no longer divide its data space [30]. For point cloud data, it is usually processed in 3 dimensions, so we need establish a 3D KD-tree.

We did a comparative experiment to compare the search efficiency of these two point cloud search methods. Both of them have the equivalent search performance when the sample data is small. When the number of point that need searching in the neighborhood of each point become larger, it takes less time for Octree to establish topological relationships between scattered points. Considering that the amount of data collected by lidar is often quite large, so Octree method is more suitable for the neighborhood search of lidar point cloud.

### D. IMPROVED EUCLIDEAN CLUSTERING ALGORITHM

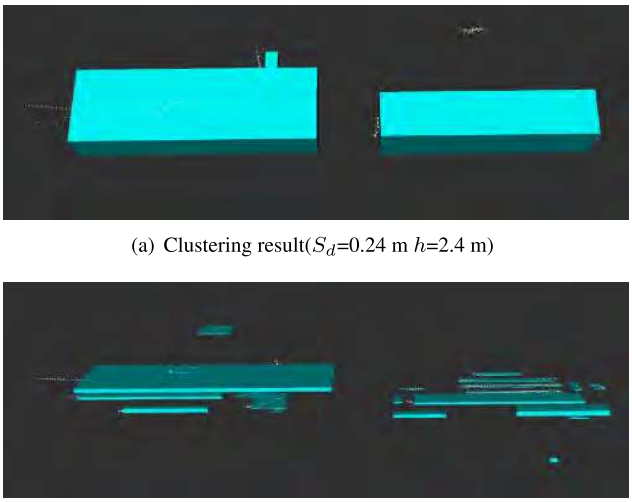
This paper presents a improved Euclidean clustering algorithm which is based on an adaptive threshold of Euclidean distance. The threshold can change its value according to the distance between the laser point and lidar. The specific processes are listed below:

First, input the point cloud data $C$ and establish the topological relationship between discrete points through Octree.

Second, create an empty cluster list $P$, a pending list $L$ and empty list $C_i^k$ (i,k $= 1, 2, 3, \cdots$), add each point $C_i$ in $C$ into the list $L$.

Third, as for the $I$-th point $C_i \in L$, search its neighborhood with radius $r < S_d$ and save the searched points in $C_i^k$. As for every points in $C_i^k$, calculate the Euclidean distance between them and point $C_i$. The points with the smallest Euclidean distance are classified into the same cluster and stored in list $P$. Repeat the steps above until all points in $L$ are processed.

Fourth, calculate the Euclidean distance between all the clusters in list $P$ and the clusters with the smallest Euclidean distance are classified into the same cluster. Repeat the process until the Euclidean distance between all the clusters in list $P$ is bigger than the distance threshold $S_d$ which can be calculated by (9).



(a) Clustering result($S_d$=0.24 m $h$=2.4 m)

(b) Clustering result($S_d$=0.15 m $h$=2.4 m)

**FIGURE 6.** Experiment to obtain the best distance threshold.

$$S_d = a(X_i^2 + Y_i^2) + b \cdot \sqrt{X_i^2 + Y_i^2} + c \tag{9}$$

$X_i$ and $Y_i$ are the coordinates of the point or the center point of cluster to be processed, $a$, $b$, and $c$ are adjustment parameters, their value normally change according to the angular accuracy of lidar. The higher the angular accuracy is the smaller the value of them should be, and vice versa. They should also be smaller if the lidar is installed longitudinally.

Eq. 9 is the most important part for $S_d$ to achieve adaptive changes. From the Equation, we can see that $S_d$ will change with the coordinates of the data points. In other words, whenever we calculate the Euclidean distance between a data point or cluster and its adjacent point or cluster, $S_d$ will adjust its value according to the coordinates of point or the center point of cluster. Thus $S_d$ will be a suitable value for each detection distance.

As for the most appropriate value of $a$, $b$, and $c$, it can be obtained by doing experiments to measure the best distance threshold $S_d$ at each detection distance and then curve fitting these points. Take one of these experiments for example. In this experiment, the distance between the lidar and the two vehicles is 6.5 m and 7.5 m. The aim of this experiment is to find the best $S_d$ that can perfectly complete the clustering of these two vehicles. The criteria for selecting the most appropriate value is shown in the figure 6, table 1, and table2.

The table 1 and table 2 are the output data of the algorithm from which we can obtain all the information about the clusters. Such as the length and width of the object, the distance between the object and the lidar, the number of data points contained in the object point cloud. It is worth noting that in order to make the data output clear and concise, we filter out all non-target clusters when performing data output.

**TABLE 1.** Output data when $S_d = 0.24$ m $h = 2.4$ m.

| Cluster information | Cluster 1 | Cluster 2 |
|---|---|---|
| Number of points | 2363 | 1088 |
| Cloud width (m) | 1.288 | 0.744 |
| Cloud length (m) | 3.964 | 3.517 |
| Cloud height (m) | 0.692 | 0.758 |
| Volume ($m^3$) | 3.025 | 1.985 |
| Detection distance (m) | 7.855 | 6.029 |

**TABLE 2.** Output data when $S_d = 0.15$ m $h = 2.4$ m.

| Cluster information | Cluster 1 | Cluster 2 | ··· | Cluster32 |
|---|---|---|---|---|
| Number of points | 488 | 451 | ··· | 26 |
| Cloud width (m) | 1.188 | 0.808 | ··· | 0.063 |
| Cloud length (m) | 3.466 | 1.207 | ··· | 0.270 |
| Cloud height (m) | 0.017 | 0.008 | ··· | 0.014 |
| Volume ($m^3$) | 0.0699 | 0.0078 | ··· | 0.0003 |
| Detection distance (m) | 7.637 | 7.742 | ··· | 5.994 |

**TABLE 3.** Best value of $S_d$ at each distance when the installation height $h$ is 2 m and 2.4 m.

| Distance(m) | 2 | 6 | 10 | 14 | 18 | 20 | 25 |
|---|---|---|---|---|---|---|---|
| $S_d$($h$=2 m) | 0.1 | 0.22 | 0.3 | 0.36 | 0.41 | 0.42 | 0.45 |
| $S_d$($h$=2.4 m) | 0.1 | 0.24 | 0.32 | 0.39 | 0.43 | 0.44 | 0.48 |

**TABLE 4.** Value of $a$, $b$, and $c$.

| Installation height $h$(m) | 2 | 2.4 |
|---|---|---|
| $a$ | $6\times10^{-4}$ | $7\times10^{-4}$ |
| $b$ | 0.031 | 0.034 |
| $c$ | 0.046 | 0.046 |

It can be known from picture (b) in figure 6 and table 2 that since the distance threshold is too small ($S_d = 0.15$ m), the two vehicles are divided into 32 different small clusters, such clustering is obviously unsuccessful. So, the distance threshold needs to be adjusted until the two vehicles are divided into two complete and independent Large clusters, just as shown in picture (a) in figure 6. At this time $S_d = 0.24$ m, so we can know that 0.24 m is the most appropriate distance threshold when detection distance is around 6 m.

Repeat the above experiment at different detection distances, once a result similar to that in picture (a) is obtained, the current distance threshold $S_d$ is the most appropriate value at the current detection distance. In this way we can obtain the most appropriate distance threshold under different detection distance.

The table 3 shows the best value of $S_d$ at each detection distance of the lidar used in this paper.

The result of the curve fitting is listed in table 4:

Finally, the volume of these clusters can be calculated by (10).

$$\begin{cases} V_n = l_n \cdot h_n \cdot d_n \\ l_n = Y^n_{max} - Y^n_{min} \\ h_n = Z^n_{max} - Z^n_{min} \\ d_n = X^n_{max} - X^n_{min} \end{cases} \tag{10}$$

**FIGURE 7.** The autonomous vehicle.

$X^n_{max}, X^n_{min}, Y^n_{max}, Y^n_{min}, Z^n_{max}, Z^n_{min}$ are the maximum and minimum value of the X Y Z coordinates in the $n$-th cluster. $V_n, l_n, h_n, d_n$ are the volume, length, height and width of the $n$-th cluster.

A noteworthy point after completing the above operation is both the weather conditions and the clutter generated by the lidar itself cause a small number of meaningless interference clusters in the clustering results. These clusters are usually small and often appear in the air. So, in order to get rid of the influences caused by these clusters, if the Z coordinates of all the data points in the cluster are larger than 0.5m and the volume of the cluster is smaller than 5 cubic centimeter then the cluster should be filter out. Clusters which contains less than 5 data points and clusters with extremely small length, width and height should also be removed. So far, the improved Euclidean clustering algorithm is completed.

## V. EXPERIMENTAL VERIFICATION IN REAL VEHICLE

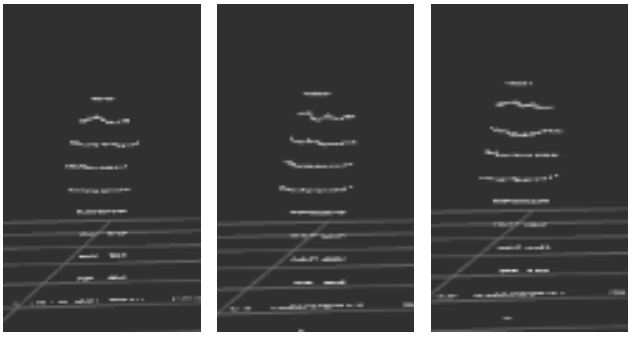
### A. EXPERIMENT PLATFORM

This paper designs several real vehicle experiments to verify the proposed algorithm. The experimental platform is an autonomous vehicle modified from a certain brand of SUV which is showed in figure 6 and the experimental scene is the campus of Jilin University. figure 7 shows the 16-line 3D lidar produced by Raytheon Intelligent Company which is installed horizontally on the autonomous vehicle. The software environment used in this experiment is Linux Ubuntu 16.04. The programming and subsequent data processing are all done on the ROS (Robot Operating System) [31] platform. And the visualization tool rviz is also used to evaluate the experimental result. Before the experiment, the lidar should be calibrated by a certain calibration method [32] so that the lidar coordinate system is consistent with the experimental vehicle coordinate system.

### B. DE-DISTORTION ALGORITHM VERIFICATION

Since the point cloud data distortion is generated during the motion of the lidar, it is necessary to let the experimental vehicle pass the target to be detected in different motion modes and compare the changes of the point cloud data before

**FIGURE 8.** The 16-line lidar.



(a) Normal point cloud (b) Distorted point cloud (c) Calibrated point cloud

**FIGURE 9.** De-distortion while driving straightly.

**TABLE 5.** Output data of experiment (1).

| Cluster information | Picture (a) | Picture (b) | Picture (c) |
|---|---|---|---|
| Number of points | 367 | 359 | 359 |
| Cloud width (m) | 0.241 | 0.293 | 0.258 |
| Cloud length (m) | 0.584 | 0.583 | 0.583 |
| Cloud height (m) | 1.692 | 1.684 | 1.684 |
| Volume ($m^3$) | 0.238 | 0.285 | 0.252 |
| Distance (m) | 6.196 | 6.127 | 6.147 |

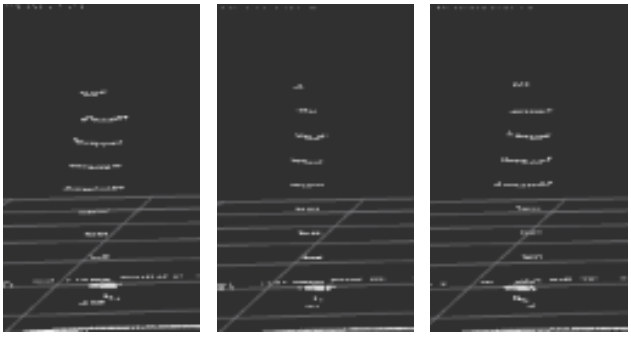
and after the de-distortion. A collecting line is set to begin the collection of point cloud when the vehicle passes it. That is to say, the distance between pedestrian and vehicle is the same in each frame of point cloud collected during the experiment. The vehicle speed is 30 Km/h, and the object to be detected is a pedestrian with a height of 1.75 m. The vehicle passes the target in the following three ways:

1) Vehicle is driving in a straight line while the pedestrian is in front of the vehicle: The comparison between the original point cloud and the data after de-distortion processing is shown in the figure 9 and table 5. In order to reflect the de-distortion effect better, the pedestrian is extracted separately.

In figure 9, picture (a) is the point cloud collected while vehicle is still and picture (b) is collected while driving, we can see that the point cloud of pedestrian in picture (b) distorted. Combined with the object volume estimation method mentioned in chapter 3.4 which considers the object as a regular cube, the change of width of the pedestrian can be

**TABLE 6.** Change of size after de-distortion.

| Size of pedestrian | Length(cm) | Width(cm) |
|---|---|---|
| Original | 58.4 | 24.1 |
| Distorted | 58.3 | 29.3 |
| De-distorted | 58.3 | 25.8 |



(a) Normal point cloud (b) Distorted point cloud (c) Calibrated point cloud

**FIGURE 10.** De-distortion while driving straightly.

**TABLE 7.** Output data of experiment (2).

| Cluster information | Picture (a) | Picture (b) | Picture (c) |
|---|---|---|---|
| Number of points | 286 | 239 | 259 |
| Cloud width (m) | 0.246 | 0.199 | 0.242 |
| Cloud length (m) | 0.581 | 0.584 | 0.584 |
| Cloud height (m) | 1.692 | 1.694 | 1.694 |
| Volume ($m^3$) | 0.242 | 0.197 | 0.239 |
| Distance (m) | 4.302 | 4.343 | 4.343 |

easily calculated. The result in table 5 shows that the width of pedestrian get bigger which will result in the wrong judgment of the size of obstacles as well as the vehicle pass ability. Picture (c) is the point cloud after de-distortion procession, the change is not so clear because the translation occurs mainly in the Y-axis direction but we can know the effect of de-distortion algorithm form the fact that the width of the pedestrian become smaller compared with the distorted one. The change of length and width are listed in table 6.

2) Vehicle is driving in a straight line while the pedestrian is on the right of the vehicle: The comparison is shown in the figure 10 and table 7.

In figure 12, picture (a) is the point cloud collected while vehicle is still and picture (b) is collected while driving, the distortion is quite clear and the width of pedestrian gets smaller. The picture (c) shows the result of de-distortion, it is worth mentioning that the vacancies after translation are filled with points in order to facilitate the subsequent clustering when the length or width become larger. Data of picture (b) and (c) in table 7 show that the width of pedestrian increases after the de-distortion algorithm. The change of length and width are listed in table 8.

3) The vehicle passes the pedestrian while turning: The comparison is shown in the figure 11 and table 9.

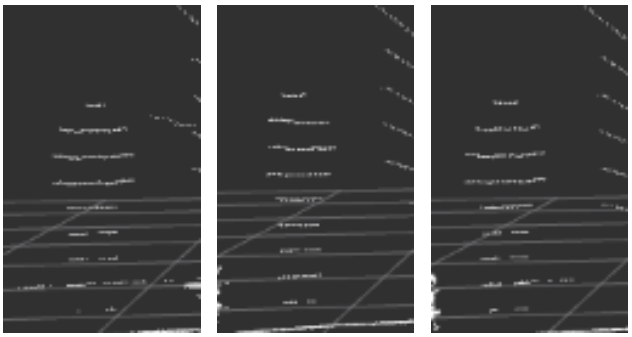
In figure 11, picture (a) is the point cloud collected while vehicle is still and picture (b) is collected while turning, the length and width of the pedestrian are simultaneously

**TABLE 8. Change of size after de-distortion.**

| Size of pedestrian | Length(cm) | Width(cm) |
|---|---|---|
| Original | 58.1 | 24.6 |
| Distorted | 58.4 | 19.9 |
| De-distorted | 58.4 | 24.3 |

**TABLE 9. Output data of experiment (3).**

| Cluster information | Picture (a) | Picture (b) | Picture (c) |
|---|---|---|---|
| Number of points | 378 | 328 | 358 |
| Cloud width (m) | 0.244 | 0.284 | 0.262 |
| Cloud length (m) | 0.586 | 0.537 | 0.570 |
| Cloud height (m) | 1.697 | 1.695 | 1.695 |
| Volume ($m^3$) | 0.243 | 0.258 | 0.253 |
| Distance (m) | 5.360 | 4.867 | 4.953 |



(a) Normal point cloud (b) Distorted point cloud (c) Calibrated point cloud

**FIGURE 11. De-distortion while turning.**

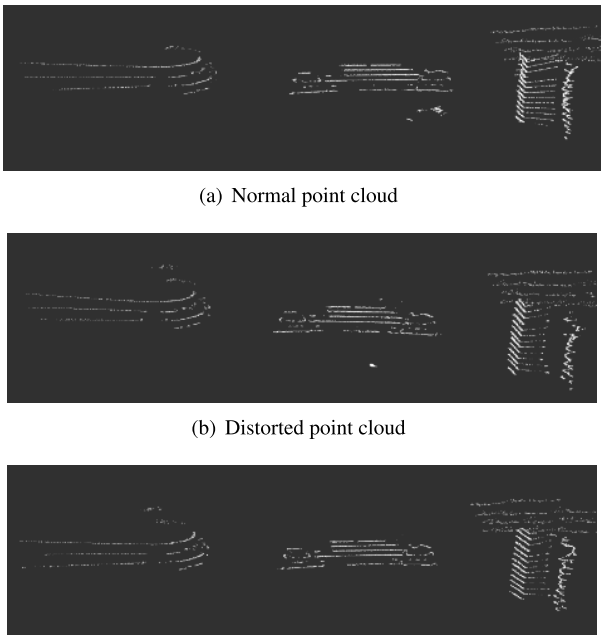
**TABLE 10. Change of size after de-distortion.**

| Size of pedestrian | Length(cm) | Width(cm) |
|---|---|---|
| Original | 58.6 | 24.4 |
| Distorted | 53.7 | 28.4 |
| De-distorted | 57 | 26.2 |

distorted, with the width increased and the length decreased. This is because the vehicle has moved in both X-axis and Y-axis direction while turning and the lidar also rotated. Data of picture (c) in table 9 shows the result of de-distortion: The width and length of pedestrian both become closer to the value while vehicle is still. The change of length and width are listed in table 10.

The situation of multiple targets and multiple kind of obstacles should also be verified. We chose two vehicles of certain brand, a pedestrian of 1.75m high and a rectangular carton with a height of 2 m and a length and width of 0.75 m to simulate various obstacles that may appear on the road. One vehicle is in front of the experimental vehicle and the other is in the left front. Pedestrian and carton are on the right side of the vehicle. The experimental vehicle travels straight through the collecting line at a speed of 30 km/h. The comparison is shown in the figure 12 and table 11-13.

In figure 12, picture (a) is the point cloud collected while vehicle is still and picture (b) is collected while moving. We can find that the point clouds of the four targets are more or less distorted, but since the change is not intuitive



(a) Normal point cloud

(b) Distorted point cloud

(c) Calibrated point cloud

**FIGURE 12. Multi-target verification experiment.**

**TABLE 11. Output data of normal point cloud.**

| Cluster information | Vehicle A | Vehicle B | Carton | Pedestrian |
|---|---|---|---|---|
| Number of points | 1999 | 932 | 488 | 291 |
| Cloud width (m) | 1.288 | 0.796 | 0.681 | 0.261 |
| Cloud length (m) | 3.992 | 3.635 | 0.637 | 0.542 |
| Cloud height (m) | 0.745 | 0.795 | 1.895 | 1.617 |
| Volume ($m^3$) | 3.825 | 2.299 | 0.820 | 0.228 |
| Distance (m) | 10.079 | 8.802 | 7.512 | 6.299 |

**TABLE 12. Output data of distorted point cloud.**

| Cluster information | Vehicle A | Vehicle B | Carton | Pedestrian |
|---|---|---|---|---|
| Number of points | 1907 | 872 | 415 | 249 |
| Cloud width (m) | 1.223 | 0.746 | 0.633 | 0.229 |
| Cloud length (m) | 3.985 | 3.649 | 0.639 | 0.553 |
| Cloud height (m) | 0.736 | 0.774 | 1.891 | 1.626 |
| Volume ($m^3$) | 3.583 | 2.107 | 0.764 | 0.206 |
| Distance (m) | 9.716 | 8.645 | 7.261 | 6.138 |

**TABLE 13. Output data of calibrated point cloud.**

| Cluster information | Vehicle A | Vehicle B | Carton | Pedestrian |
|---|---|---|---|---|
| Number of points | 1957 | 892 | 435 | 279 |
| Cloud width (m) | 1.271 | 0.783 | 0.668 | 0.256 |
| Cloud length (m) | 3.985 | 3.649 | 0.639 | 0.553 |
| Cloud height (m) | 0.736 | 0.774 | 1.891 | 1.626 |
| Volume ($m^3$) | 3.730 | 2.206 | 0.824 | 0.229 |
| Distance (m) | 9.716 | 8.645 | 7.261 | 6.138 |

enough from the point cloud, we use the data in table 11-13 to establish the table 14 shown below.

We can know from the table that the distortion of four targets conform to the laws mentioned previously and the de-distortion algorithm can effectively reduce the distortion of all these four targets. The distance between these four targets and lidar is 10 m, 8.8 m, 7.5 m, and 6.3 m which can prove that

**TABLE 14. Change of size after de-distortion.**

| Kind of object | Vehicle A | Vehicle B | Pedestrian | Carton |
|---|---|---|---|---|
| Original width(cm) | 128.8 | 79.6 | 68.1 | 26.1 |
| Distorted width(cm) | 122.3 | 74.6 | 63.3 | 22.9 |
| Calibrated width(cm) | 127.2 | 78.3 | 66.8 | 24.9 |
| Original length(cm) | 399.2 | 363.5 | 63.7 | 54.2 |
| Distorted length(cm) | 398.5 | 364.9 | 63.9 | 55.3 |
| Calibrated length(cm) | 398.5 | 364.9 | 63.9 | 55.3 |

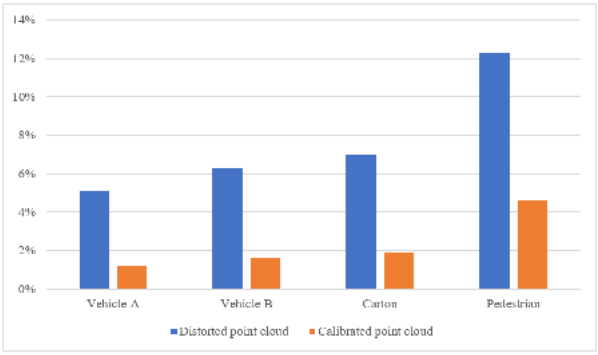



**FIGURE 13. The distortion of volume compared with normal point cloud.**

the de-distortion algorithm is effective for multiple targets at different distances.

We made a column chart for the distortion ratio of the point cloud volume before and after the de-distortion. In this way we can clearly illustrate the efficacy of the de-distortion algorithm. The column chart is shown in figure 13. From the chart we can see that the distortion of all targets is greatly reduced after processed by the algorithm, both above 60%.

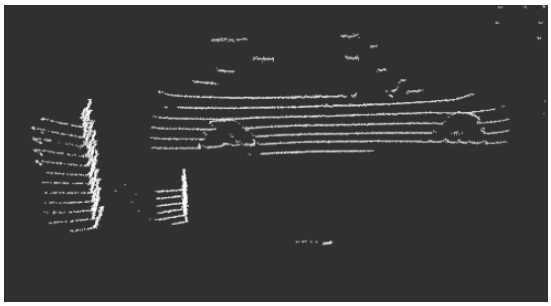
The experiment above shows that the de-distortion algorithm can efficiently reduce the influence of object size caused by point cloud distortion in various situation. The host computer receives 8 to 9 frames of processed point cloud per second which meets the real-time requirements of the autonomous vehicles.

### C. EXPERIMENT TO PROVE THE ADVANTAGE OF ADAPTIVE DISTANCE THRESHOLD

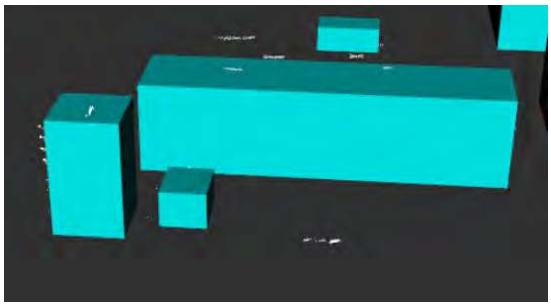
In this experiment, the point cloud will be processed by the de-distortion algorithm before it is used for clustering. First we need to verify the advantage of the improved Euclidean clustering algorithm compared with the traditional one. That is to say the advantage of adaptive distance threshold should be reflected. In the first experiment two cartons of different size are placed 6m in front of the experimental vehicle and there is a vehicle at 8 m away. The installation height of lidar is 2.4 m, so, a $= 7 \times 10^{-4}$, b $= 0.034$, c $= 0.046$.

The comparison of the traditional and improved Euclidean clustering algorithm is shown in figure 14.

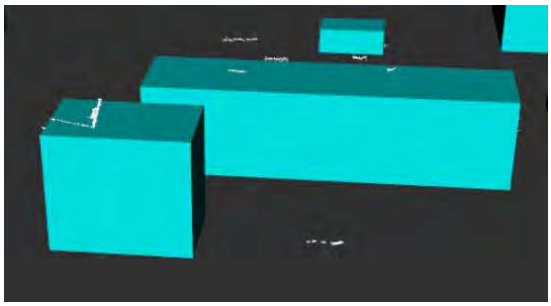
Picture (a) is the original point cloud, we can know from the picture that the distance between two cartons are quite small which can easily lead to the misjudgment for traditional algorithm. Picture (b) shows the result of improved clustering algorithm, these three targets are replaced by regular cubes to stand for their size. It is clear that the clustering is very successful. The size and location of the vehicle and cartons are accurately presented. The picture (c) shows the result of traditional Euclidean clustering algorithm. In order to successfully complete the clustering of the vehicle, we choose a larger distance threshold $S_d = 0.32$ m. Although this method successfully completed the clustering of the vehicle, it do divide two cartons into a same cluster because $S_d$ is too big for the nearby point cloud. In picture (d) we try to



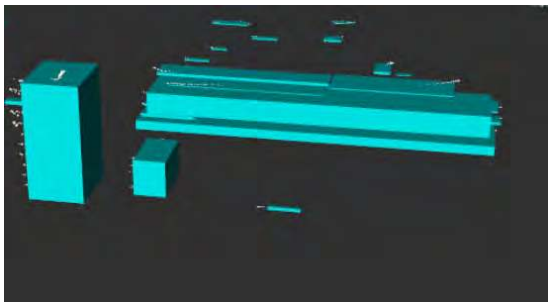
(a) Oringinal point cloud



(b) Resullt of improved Euclidean clustering algorithm



(c) Result of traditional Euclidean clustering algorithm when $S_d$=0.32 m



(d) Result of traditional Euclidean clustering algorithm when $S_d$=0.2 m

**FIGURE 14. Experiment to prove the advantage of adaptive distance threshold.**

**FIGURE 15.** The placement of roadblocks.

**TABLE 15.** Output data of improved algorithm ($d$ = 7.5 m).

| Cluster information | A | B | C | D | E |
|---|---|---|---|---|---|
| Number of points | 268 | 264 | 203 | 161 | 151 |
| Cloud width (m) | 0.129 | 0.312 | 0.173 | 0.303 | 0.203 |
| Cloud length (m) | 0.246 | 0.273 | 0.0.263 | 0.220 | 0.201 |
| Cloud height (m) | 0.497 | 0.501 | 0.491 | 0.542 | 0.457 |
| Volume ($m^3$) | 0.015 | 0.043 | 0.024 | 0.036 | 0.019 |
| Distance (m) | 7.317 | 7.317 | 7.320 | 7.894 | 7.884 |

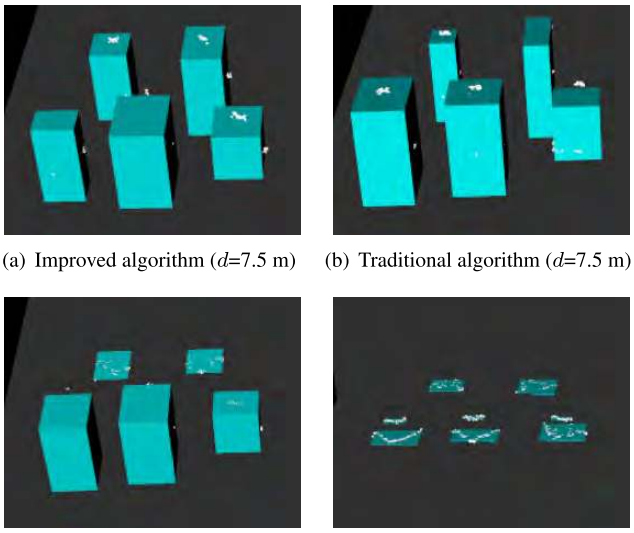
complete the clustering of two cartons by choosing a smaller distance threshold $S_d = 0.2$ m. But this time the vehicle is divided into multiple small clusters because $S_d$ is too small for the distant point cloud. Picture (c) and picture (d) illustrate the shortcoming of fixed distance threshold: It cannot ensure the accuracy of clustering for both nearby and distant object when the detection distance is too far.

The experiment above obviously confirm the advantages of adaptive distance threshold. It can accurately complete the clustering of distant objects while ensuring that no near objects are misjudged. Thus extending the detection distance by make up the shortcomings of the traditional Euclidean clustering algorithm.

### D. ALGORITHM PERFORMANCE PARAMETER COMPARISON EXPERIMENT

After verifying the advantage of adaptive distance threshold, we need to compare the detection distance, detection accuracy, and the running speed of these two algorithms. In this experiment we select 5 roadblocks which are shown in figure 15 as detection targets, they are 0.3m long and wide and 0.55m high. Comparing the performance parameters of the algorithm under the detection distances $d$ of 7.5 m and 15 m. The result of experiment are shown in figure 16 and table 15-18.

In figure 15, picture (a) and (b) are the result of improved and traditional Euclidean clustering algorithm when the detection distance $d = 7.5$ m. We can know from the data in table 15 that the improved Euclidean clustering algorithm is very successful in clustering 5 roadblocks, the size and location information are relatively accurate. And the result



(a) Improved algorithm ($d$=7.5 m) (b) Traditional algorithm ($d$=7.5 m)

(c) Improved algorithm ($d$=15 m) (d) Traditional algorithm ($d$=15 m)

**FIGURE 16.** Algorithm performance parameter comparison experiment.

**TABLE 16.** Output data of traditional algorithm ($d$ = 7.5 m).

| Cluster information | A | B | C | D | E |
|---|---|---|---|---|---|
| Number of points | 259 | 260 | 195 | 101 | 105 |
| Cloud width (m) | 0.309 | 0.322 | 0.164 | 0.285 | 0.151 |
| Cloud length (m) | 0.249 | 0.236 | 0.247 | 0.123 | 0.104 |
| Cloud height (m) | 0.497 | 0.495 | 0.398 | 0.476 | 0.504 |
| Volume ($m^3$) | 0.038 | 0.037 | 0.017 | 0.017 | 0.008 |
| Distance (m) | 7.320 | 7.327 | 7.287 | 7.745 | 7.793 |

**TABLE 17.** Output data of improved algorithm ($d$ = 15 m).

| Cluster information | A | B | C | D | E |
|---|---|---|---|---|---|
| Number of points | 61 | 58 | 49 | 26 | 24 |
| Cloud width (m) | 0.229 | 0.316 | 0.175 | 0.293 | 0.203 |
| Cloud length (m) | 0.248 | 0.224 | 0.262 | 0.200 | 0.193 |
| Cloud height (m) | 0.492 | 0.503 | 0.499 | 0.049 | 0.057 |
| Volume ($m^3$) | 0.028 | 0.035 | 0.023 | 0.003 | 0.002 |
| Distance (m) | 15.122 | 15.127 | 15.138 | 15.626 | 15.624 |

**TABLE 18.** Output data of traditional algorithm ($d$ = 15 m).

| Cluster information | A | B | C | ··· | H |
|---|---|---|---|---|---|
| Number of points | 30 | 28 | 31 | ··· | 14 |
| Cloud width (m) | 0.155 | 0.167 | 0.179 | ··· | 0.031 |
| Cloud length (m) | 0.249 | 0.232 | 0.247 | ··· | 0.103 |
| Cloud height (m) | 0.048 | 0.048 | 0.047 | ··· | 0.021 |
| Volume ($m^3$) | 0.0018 | 0.0019 | 0.0021 | ··· | 0.0001 |
| Distance (m) | 15.067 | 15.066 | 15.073 | ··· | 15.058 |

of traditional algorithm shows that the algorithm successfully clusters the three roadblocks in the front row, but the clustering of the two rear roadblocks is significantly smaller in length. This is because the algorithm fails to gather the point clouds of the rear roadblocks into a same cluster. And those clusters which are too small are removed as clutter, resulting in smaller cluster lengths. At this time, the data output of the two algorithms is 74 frames and 63 frames every 10 seconds, the detection accuracy is 100% and 60%.

**TABLE 19. Comparison between two algorithms.**

| Performance | Traditional | Improved |
|---|---|---|
| Detection distance(m) | 7.3 | 15.1 |
| Frames per 10 second | 64 | 75 |
| Detection accuracy | 30% | 80% |

picture (c) and (d) are the result of improved and traditional Euclidean clustering algorithm when the detection distance $d = 15$ m. Again we can know from the data in table 17 that the improved algorithm fails in the clustering of rear roadblocks because there are too little point cloud data which leads to complete loss of height information. And traditional algorithm fails in the clustering of all roadblocks because 15 m significantly exceeds the detection distance of traditional algorithms. Of all the information collected by the traditional algorithm, only the location information is correct. At this time, the data output of the two algorithms is 77 frames and 65 frames every 10 seconds, the detection accuracy is 60% and 0%.

We synthesize the performance parameters of the two experiments and list them in table 19.

It can be seen from the table that the improved algorithm greatly improves the accuracy of detecting distant targets and increases the detection distance. The calculating speed of improved algorithm is 15% faster than the traditional one because inappropriate distance thresholds can lead to the increase of useless clusters, which slows clustering.

## VI. CONCLUSION

In order to obtain more realistic point cloud data for clustering, this paper proposes a new de-distortion algorithm. Through the coordinate transformation of the distorted point, the data distortion caused by the motion of the lidar is reduced. So, the real environment information is restored which provides guarantee for the accuracy of the subsequent clustering process. The advantage of the point cloud de-distortion algorithm in this paper is that the calculation amount is smaller than other algorithms, and the fast computing speed means better real-time performance. It is suitable for autonomous vehicles, which have high requirements for real-time performance. And the de-distortion effect is also obvious. However, the de-distortion algorithm does not perform so well when the vehicle is turning. At the same time, this paper also proposes an improved algorithm for the disadvantages of traditional Euclidean clustering algorithm in dealing with 3D lidar point cloud data. By correlating the distance threshold and the detection distance, the distance threshold can be flexibly changed, which means the threshold is small when detecting the objects in nearby area and it will become larger when the detection distance increases. In this way the problem caused by uniform point density of point clouds can be dealt with, so the obstacles in nearby and distance area can be detected quickly and accurately. So, the advantage of the improved Euclidean clustering algorithm is that it can handle uneven point clouds very well and can detect objects in a wide range with high accuracy, which is more flexible than the traditional Euclidean clustering algorithm.

The next research goal of this paper is to further optimize the clustering algorithm structure and speed up the clustering calculation, so as to meet the real-time requirements of vehicles at high speed. And further improve the performance of the de-distortion algorithm when the vehicle is turning.

## REFERENCES

[1] K. Bimbraw, "Autonomous cars: Past, present and future a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology," in *Proc. 12th Int. Conf. Inform. Control, Automat. Robot. (ICINCO)*, vol. 1, Jul. 2015, pp. 191–198.
[2] H. Gao, B. Cheng, J. Wang, K. Li, J. Zhao, and D. Li, "Object classification using CNN-based fusion of vision and LIDAR in autonomous vehicle environment," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4224–4231, Sep. 2018.
[3] M. Himmelsbach, A. Müller, T. Lüttel, and H.-J. Wünsche, "LIDAR-based 3D object perception," presented at the Proc. 1st Int. Workshop Cognition Tech. Syst., Jan. 2008.
[4] Y. Zhang, J. Wang, X. Wang, and J. M. Dolan, "Road-segmentation-based curb detection method for self-driving via a 3D-LiDAR sensor," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 12, pp. 3981–3991, Dec. 2018.
[5] G. Seetharaman, A. Lakhotia, and E. P. Blasch, "Unmanned vehicles come of age: The DARPA grand challenge," *Computer*, vol. 39, no. 12, pp. 26–29, Dec. 2006.
[6] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, and K. Lau, "Stanley: The robot that won the DARPA grand challenge," *J. Field Robot.*, vol. 23, no. 9, pp. 661–692, 2006.
[7] G. Wang, J. Wu, R. He, and S. Yang, "A point cloud-based robust road curb detection and tracking method," *IEEE Access*, vol. 7, pp. 24611–24625, 2019.
[8] S.-R. Feng and W.-J. Xiao, "An improved DBSCAN clustering algorithm," *J.-China Univ. Mining Technol.*, vol. 37, no. 1, pp. 105–111, Jan. 2008.
[9] D. Kong and W. Xy, "Vehicle target identification algorithm based on poin cloud of vehicle 32-line laser lidar," *Sci. Technol. Eng.*, vol. 18, no. 5, pp. 81–85, May 2018.
[10] X. Z. Wang, J. Li, H. J. Li, and B. X. Shang, "Obstacle detection based on 3D laser scanner and range image for intelligent vehicle," *J. Jilin Univ. (Eng. Technol. Ed.)*, vol. 46, no. 2, pp. 360–365, Apr. 2016.
[11] S. I. R. Rodríguez and F. de Assis Tenorio de Carvalho, "Fuzzy clustering algorithm based on adaptive Euclidean distance and entropy regularization for interval-valued data," presented at the Artif. Neural Netw. Mach. Learn., Oct. 2018.
[12] L. Bai, L. Yan, and Z. M. Ma, "Determining topological relationship of fuzzy spatiotemporal data integrated with XML twig pattern," *Appl. Intell.*, vol. 39, no. 1, pp. 75–100, 2013.
[13] D. H. Zhu, "Euclidean cluster extraction in PCL," in *Point Cloud Library PCL Study Tutorial*, 2th ed. Beijing, China: Univ. Aeronautics and Astronautics Press, 2012, pp. 338–350.
[14] H. Rashmanlou, S. Sahoo, R. A. Borzooei, M. Pal, and A. Lakdashti, "Computation of shortest path in a vague network by Euclidean distance," *J. Multiple-Valued Logic Soft Comput.*, vol. 30, no. 1, pp. 115–123, 2018.
[15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[16] L. Tong and X. Ying, "3D point cloud initial registration using surface curvature and SURF matching," *3D Res.*, vol. 9, no. 3, p. 41, Sep. 2018.
[17] N. H. Remeli, R. L. A. Shauri, F. H. Yahaya, N. M. Salleh, K. Nasir, and A. I. M. Yassin, "SURF based 3D object recognition for robot hand grasping," *Pertanika J. Sci. Technol.*, vol. 25, pp. 287–296, Mar. 2017.
[18] C. Redondo-Cabrera, R. J. López-Sastre, J. Acevedo-Rodriguez, and S. Maldonado-Bascón, "SURFing the point clouds: Selective 3D spatial pyramids for category-level object recognition," presented at the IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2012.
[19] H. Lin, "The lidar based obstacle detecion and map building," M.S. thesis, Dept. Electron. Eng., Zhe Jiang Univ., Zhejiang, China, 2017.
[20] F. de Assis Tenorio de Carvalho, "A fuzzy clustering algorithm for symbolic interval data based on a single adaptive Euclidean distance," presented at the Neural Inf. Process., Mar. 2006.

[21] S. G. Zhou, A. Y. Zhou, and J. Cao, "A data-partitioning-based DBSCAN algorithm," *J. Comput. Res. Develop.*, vol. 37, no. 10, pp. 1153–1159, May 2000.
[22] C.-H. Guan, Y.-D. Chen, H.-Y. Chen, and J.-W. Gong, "Improved DBSCAN clustering algorithm based vehicle detection using a vehicle-mounted laser scanner," *Trans. Beijing Inst. Technol.*, vol. 30, no. 6, pp. 732–736, Jun. 2010.
[23] Z. Song, R. Fu, M.-F. Zhang, and X.-Y. Liu, "The application of partitioning-density-based spatial clustering of applications with noise algorithm in LIDAR based pedestrian detection system," *Sci. Technol. Eng.*, vol. 17, no. 18, pp. 282–287, 2017.
[24] Y. Zhou, D. Wang, X. Xie, Y. Ren, G. Li, Y. Deng, and Z. Wang, "A fast and accurate segmentation method for ordered LiDAR point cloud of large-scale scenes," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 11, pp. 1981–1985, Nov. 2014.
[25] K. Liu, W. Wang, R. Tharmarasa, J. Wang, and Y. Zuo, "Ground surface filtering of 3D point clouds based on hybrid regression technique," *IEEE Access*, vol. 7, pp. 23270–23284, 2019.
[26] J. Pach, R. Pollack, and J. Spencer, "Graph distance and Euclidean distance on the grid," *Topics Combinatorics Graph Theory*, vol. 6, pp. 555–559, Nov. 1990.
[27] D. Vasiliu, T. Dey, and I. L. Dryden, "Penalized Euclidean distance regression," *Stat*, vol. 7, no. 1, p. e175, Feb. 2018.
[28] S. Angelo, S. Said, and B. Jack, "Variable neighborhood search," presented at the 6th Int. Conf., ICVNS, Sithonia, Greece, Oct. 2018.
[29] A.-V. Vo, L. Truong-Hong, D. F. Laefer, and M. Bertolotto, "Octree-based region growing for point cloud segmentation," *ISPRS J. Photogramm. Remote Sens.*, vol. 104, pp. 88–100, 2015.
[30] Z. Xiao and W. Huang, "Kd-tree based nonuniform simplification of 3D point cloud," presented at the 3rd Int. Conf. Genetic Evol. Comput., Aug. 2009.
[31] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, 2009, vol. 3, no. 2, pp. 5–10.
[32] C. Guibin, G. Zhenhai, and H. Lei, "Step-by-step automatic calibration algorithm for exterior parameters of 3D lidar mounted on vehicle," *Chin. J. Lasers*, vol. 44, no. 10, 2017, Art. no. 1010004.

**LONG WEN** received the B.S. degree in vehicle engineering from the Wuhan University of Technology, Wuhan, China, in 2017. He is currently pursuing the M.S. degree in vehicle engineering with Jilin University, Changchun, China. His current research interests include environment perception and intelligent vehicle technology.

**LEI HE** received the M.S. and Ph.D. degrees in vehicle engineering from Jilin University, Changchun, China, where he is currently an Associate Professor with the State Key Lab of Automotive Simulation and Control. His research interests include dynamics simulation and control, automotive intelligent driving, and assisted driving.

**ZHENHAI GAO** received the M.S. and Ph.D. degrees in vehicle engineering from Jilin University, Changchun, China. He is currently the Associate Dean of the Faculty of Vehicle Engineering, Jilin University. He has authored over 60 peer-reviewed international journals and conferences. His current research interests include driver maneuvering behavior and humanized design, methods and key technologies for safety, comfort, and amenity vehicles.