ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20230507000612
Original scientific paper

A Novel Point Cloud Compression Algorithm for Vehicle Recognition Using Boundary Extraction

Yanjun ZHANG, Pengcheng SONG, Qi JING, Qingyun LI*

Abstract: Recently, research on the hardware system for generating point cloud data through 3D LiDAR scanning has improved, which has important applications in autonomous driving and 3D reconstruction. However, point cloud data may contain defects such as duplicate points, redundant points, and an unordered mass of points, which put higher demands on the performance of hardware systems for processing data. Simplifying and compressing point cloud data can improve recognition speed in subsequent processes. This paper studies a novel algorithm for identifying vehicles in the environment using 3D LiDAR to obtain point cloud data. The point cloud compression method based on the nearest neighbor point and boundary extraction from octree voxels center points is applied to the point cloud data, followed by the vehicle point cloud identification algorithm based on image mapping for vehicle recognition. The proposed algorithm is tested using the KITTI dataset, and the results show improved accuracy compared to other methods.

Keywords: autonomous driving; compression; point cloud; vehicle recognition; 3D LiDAR; 3D reconstruction

1 INTRODUCTION

With the rapid economic development in China, people's living standards have improved, leading to significant changes in transportation modes and more choices for individuals. According to the 2020 statistical report by the National Bureau of Statistics, the number of civilian cars increased from 232 million in 2018 to 253 million in 2019, representing an 8% growth. However, the increasing number of cars has also led to a rise in traffic accidents in China, with 247,646 accidents reported in 2019, resulting in significant loss of life and property [1].

In the real-world traffic system, driving vehicles are often surrounded by pedestrians and other moving vehicles, posing potential collision hazards. Therefore, the ability of autonomous vehicles to quickly identify and avoid targets, such as other vehicles during travel, is critical for passenger safety. Advanced Driver Assistance Systems (ADAS) have become a research focus in the vehicle industry. ADAS uses 3D laser ranging to scan the surrounding environment and obtain point cloud data, which is then quickly identified and processed by the decision and control system to assist the driver or take over the vehicle when necessary. The "Made in China 2025" document released by the State Council in April 2015 emphasizes that intelligent vehicles equipped with advanced driving assistance systems can improve traffic efficiency and reduce the accident rate by over 30%.

To recognize targets such as vehicles, it is necessary to conduct 3D scanning and obtain the point cloud data, which is then processed for accurate recognition. With the continuous advancement of technology, the accuracy of laser-ranging sampling has reached the sub-millimeter level [2]. During 3D laser scanning, a lot of model information, such as the object's coordinates, surface reflection intensity, length, width, and height, can be obtained after a scanning cycle. However, the point cloud data obtained may have various problems, such as outliers, missing data, duplication, redundancy, and excessive density due to precision errors of the scanning equipment, different measurement methods, operator proficiency levels, environmental factors, equipment shaking during scanning, dust, smoke, and others [3]. Using these massive points for model reconstruction and subsequent processing and recognition can pose significant hazards to the safety of high-speed vehicles and passengers and create tremendous pressure on the hardware system while consuming a lot of time. These problems also directly affect the accuracy of target recognition in the 3D point cloud. Therefore, simplifying and compressing point cloud data while minimizing the impact on data quality is important for subsequent target recognition. Improving the recognition speed of surrounding vehicles can lead to more timely measures to avoid obstacles, which is of great significance for the safety of high-speed vehicles and passengers.

Many experts and scholars both domestically and internationally have proposed methods for simplifying and compressing point cloud data while minimizing the impact on data quality and ensuring necessary information for model reconstruction. Currently, there are two main types of point cloud compression solutions. The first is video-based compression methods, which use traditional image or video coding methods to map the geometric and attribute information of 3D point cloud data onto a 2D plane before compressing it. This method is suitable for dense point cloud data. For relatively sparse point cloud data, a geometry-based compression method is often used. This method involves processing the geometric information of the point cloud data by dividing it into spatial structures, reconstructing geometric information, and finally processing attribute information.

2 LITERATURE REVIEW

2.1 Point Cloud Compression

Numerous techniques have been proposed by experts and scholars worldwide to simplify and compress point cloud data while maintaining data quality and ensuring necessary information for model reconstruction. Currently, two primary point cloud compression methods have emerged. The first method is based on video point cloud compression, using traditional image or video encoding techniques to map the geometric and attribute information of 3D point cloud data onto a 2D plane before compressing it. This method is suitable for dense point cloud data. For

relatively sparse point cloud data, a geometry-based point cloud compression method is often used. This method involves processing the geometric information of the point cloud data by dividing its spatial structure, reconstructing the geometric information, and finally processing the attribute information.

2.1.1 Video-Based Point Cloud Compression

Video-based point cloud compression, also known as mapping-based methods, involves mapping 3D dynamic point cloud data to 2D planes in the form of patches or regions, generating continuous 2D video frame sequences, which are then processed using traditional video or image compression technologies such as H.264/AVC [4] or H.265/HEVC [5]. The challenge of this method lies in efficiently utilizing intra-frame and inter-frame prediction techniques to ensure minimal loss of point cloud data while maintaining compression encoding efficiency. Budagavi et al. proposed a tree-based method [6], dividing point cloud data into tree structures and scanning each tree node from top to bottom, mapping all points in the point cloud onto the same 2D plane. Although this method is simple and easy to implement, the resulting 2D video frame sequences have weak spatiotemporal correlations, resulting in low compression efficiency. Another mapping method based on bounding frameworks was adopted by He et al. [7]. This method involves first calculating the range of the point cloud to determine its overall bounding framework and then projecting the point cloud onto one face of the bounding framework. The spatiotemporal correlation of the video frame sequences obtained by this method is better than that of the tree-based method. However, point cloud data often contains many duplicate points, resulting in significant overlap in the projection process that causes data loss and distortion in the subsequent reconstruction process.

2.1.2 Geometry-Based Point Cloud Compression

Geometry-based point cloud compression methods are effective for both high-density and low-density point clouds. For instance, the geometry-based Point Cloud Compression (G-PCC) standard [8] was released by MPEG. This method processes the geometric information of the point cloud by dividing its spatial structure, followed by the attribute information based on the reconstructed geometric information. Reasonable and effective spatial structure division methods can significantly reduce the spatial redundancy of 3D point clouds, thus enhancing the compression efficiency of point cloud geometry and attribute information. Currently, the primary method is based on tree-structured spatial divisions. An octree-based point cloud compression method [9] was proposed, which is simple and effective and can handle unorganized point clouds of any size and density. This method iteratively divides each level from the parent node into 8 child nodes down to the lowest level of the octree, with each child node using a flag bit to record whether point information is present. As seen from the division process, the occupancy of child nodes is determined by the parent node, and the occupancy of adjacent nodes also has a certain correlation. Therefore, context-based methods are employed for encoding occupancy information. Garcia et al. proposed a node approximation method to obtain geometric context [10], which gradually increases the resolution of the tree structure, generating an approximate child node on each tree branch, and using it as a context to drive the arithmetic encoder. However, as the resolution increases without limit, the extra information to be encoded cannot balance the gains achieved by the context coding method.

2.2 Point Cloud Recognition

At present, 3D target recognition technology is mainly divided into two categories: 3D target recognition algorithm based on 2D image [10-12, 26] and 3D target recognition algorithm based on 3D point cloud and grid [13-16, 27, 29].

2.2.1 Three-Dimensional Target Recognition Method

The appearance of 3D objects can be reflected by the brightness, posture, shape, etc. in 2D images. According to different recognition methods, image-based recognition is divided into: recognition based on geometry or model, and recognition based on local feature description.

1. Geometry Model Based Recognition

This method primarily focuses on the geometric descriptions of the target's appearance, which can be projected into 2D images through cameras. Roberts proposed an edge detection and line fitting algorithm [18] for 3D object recognition based on edge features extracted from images taken at different angles. However, the drawback of this method is that it often requires feature extraction from multiple images taken at different angles to accurately determine the target's structure. Yang, on the other hand, extracts geometric primitives with view-invariant properties, such as lines and loops [19]. This recognition method relies on users to provide prior knowledge about the target's appearance. The computer then obtains the target's geometric feature descriptions from the input image based on this appearance information and matches them with feature descriptions in the template library to recognize 3D objects. The recognition method based on geometry or models requires prior knowledge about the target's appearance, and these geometric descriptions often only include the shape information of 3D objects, neglecting the color and other texture information of the target's surface. Therefore, this method is highly sensitive to target occlusion or complex scene problems.

2. Local Feature Description Based Recognition

Merely relying on geometric features such as overall shape, edges, and surface information is inadequate for accurately and efficiently recognizing targets, especially in complex scenes where the target is occluded or surrounded by other objects. To address this limitation, Lowe proposed a method of detecting representative locations in targets, known as scale-invariant feature points [20]. By detecting SIFT feature points based on unexcluded information and using the gradient information of the surrounding pixels of these feature points for feature description, recognition methods based on local feature description can achieve 3D object recognition in complex scenes and under occlusion conditions, as long as the target is not completely occluded.

This approach is less sensitive to complex scenes and can effectively handle occlusion.

2.2.2 3D Point Cloud Grid Based Recognition Method

Guo et al. [21] specifically studied object recognition in complex scenes and occlusion problems in 3D point clouds and 3D meshes. They divided the 3D object recognition methods in point clouds and meshes into two categories: global feature-based 3D object recognition and local feature-based 3D object recognition. The global feature-based recognition techniques treat the object as a whole and use a feature vector to describe the object's geometric properties. Common global features include Viewpoint Feature Histogram (VFH) [22] and Clustered Viewpoint Feature Histogram (CVFH) [23]. The VFH feature calculates the pose using the entire 3D point cloud information while adding histogram information of the angle between the viewpoint direction and other point normal. The CVFH feature, based on the original VFH, takes into account the missing point cloud and calculates global features for stable regions in the cluster by segmenting the point cloud model using a region-growing segmentation algorithm.

Global features are a type of feature description that represents the target by calculating the features of the entire target (or the segmented target). However, when the target is in a complex scene or occluded, the model must be complete or an appropriate segmentation algorithm is needed to separate the target model from the scene. While global features ignore the details of the target shape, they are not suitable for 3D object recognition in complex scenes and partially occluded environments. Instead, it is more effective to extract salient and stable feature points on the target surface and then extract local features from these points when dealing with complex scenes and local occlusion problems. This approach has been validated in 2D image object recognition, where local features involve extracting geometric information around feature points and encoding it into a high-dimensional vector [24, 28, 30]. For 3D local feature object recognition, Guo [25] studied ten commonly used local feature descriptors, including rotational descriptors, 3D shape context descriptors, LSP, Fast Point Feature Histogram, among others. They tested the descriptiveness and robustness of these descriptors against disturbances such as Gaussian noise, sharp noise, support domain radius, occlusion, and others in eight model libraries. While local features can successfully solve occlusion and complex scene problems, they may not be effective in recognizing objects under other disturbances, such as noise and changes in mesh resolution or point cloud density.

3 COMPRESSION METHOD BASED ON OCTREE VOXEL AND BOUNDARY EXTRACTION

3.1 Octree Voxel Based Point Cloud Compression Method

This method mainly uses the nearest neighbor point to the center point of octree voxel to replace the center point and simplifies the point cloud data to achieve the purpose of compression. The advantage of this method is that it can retain the original structure of point cloud data. For a complete point cloud data set, $P=\{P_1,P_2,\ldots,P_n\}$, n is the total number of points of the point cloud data set. Calculate the maximum and minimum values of the point cloud data set on the 3D coordinate axis, and create a minimum bounding rectangle according to this, as shown in Eq. (1):

$$(M)s=m_{i-1}m_{i-2},\ldots,m_1m_0 \tag{1}$$

The minimum bounding rectangle corresponds to the root node of the octree, and then the minimum bounding rectangle is divided. After division, each rectangle corresponds to a sub-node of the octree, and this is coded in octal mode in turn. The coding of nodes is shown in Eq. (2):

$$(M)_{10}=m_{i-1}8^{i-1}+m_{i-2}8^{i-2}+\ldots+m_18^1+m_08^0 \tag{2}$$

The partition path of octree nodes is represented by m_0 to m_{i-1}. The conversion relationship between octet-encoded nodes and decimal encoding is shown in formula (3). The octree partition level is represented by i, and m_i is octal encoding, m_0, $m_{i-1}\in[0,7]$. Calculate the position of any point $P(x,y,z)$ in the octree in the point cloud data set, and mark the index values of its sub-nodes as e, f and g.

$$\begin{cases} e=\left[(x-x_{\min})/a\right] \\ f=\left[(y-y_{\min})/b\right] \\ g=\left[(z-z_{\min})/c\right] \end{cases} \tag{3}$$

The coordinate $(X_{\min},Y_{\min},Z_{\min})$ is the minimum vertex coordinate of the smallest outer rectangle on the three-dimensional coordinate axis. Eq. (3) is converted into binary representation as follows:

$$\begin{cases} e=a_{i-1}2^{i-1}+a_{i-2}2^{i-2}+\ldots+a_12^1+a_02^0 \\ f=b_{i-1}2^{i-1}+b_{i-2}2^{i-2}+\ldots+b_12^1+b_02^0 \\ g=c_{i-1}2^{i-1}+c_{i-2}2^{i-2}+\ldots+c_12^1+c_02^0 \end{cases} \tag{4}$$

In the above formula $(a_0,a_1,\ldots,a_{i-1})\in\{0,1\}$.

The cut-off condition of octree structure is judged by voxel size. If the threshold condition is not reached, the partition will continue, and if the threshold condition is reached, the partition will stop immediately. Then calculate and obtain the position index of octree voxels and the center point coordinates of octree voxels. Finally, Euclidean distance is used to calculate the point closest to the center point in the octree voxel, and this nearest neighbor point is used to replace all other points to complete the point cloud compression.

3.2 Point Cloud Boundary Information Extraction

Determine whether a point is a boundary point according to the vector angle of the projection of a point and other points in its neighborhood in the tangent plane. If the point is a boundary point, there must be an angle

between two projection vectors that is much greater than the angle between other vectors. The main process of the algorithm is as follows:

The plane fitting is performed for the sampling point and its neighborhood points. Assuming that the detected point is point P, its neighborhood point set is $P_k = \{P_0, P_1, \ldots, P_{k-1}\}$, and the least squares fitting algorithm is used. The general expression of plane equation is as follows:

$$A*x + B*y + C*z + D = 0 (C \neq 0) \tag{5}$$

$$Z = -\frac{A}{C}*x - \frac{B}{C}*y - \frac{D}{C} = 0 \tag{6}$$

$$a_0 = -\frac{A}{C}, a_1 = -\frac{B}{C}, a_2 = -\frac{D}{C} \tag{7}$$

Substitute Eq. (7) into Eq. (6) to obtain the following formula:

$$a_0 * x + a_1 * y + a_2 \tag{8}$$

For a series of points $n(n \geq 3)$, (x_i, y_i, z_i). $i = 0,1,\ldots,n-1$. To fit the plane equation with the point n, that is, making Eq. (9) the smallest, it should meet $a\frac{\partial S}{\partial a_k} = 0, k = 0,1,2$.

$$S = \sum_{i=0}^{n-1} \left(a_0 * x + a_1 * y + a_2 - z\right)^2 \tag{9}$$

$$\begin{cases} \sum 2\left(a_0 * x_i + a_1 * y_i + a_2 - z_i\right) * x_i = 0 \\ \sum 2\left(a_0 * x_i + a_1 * y_i + a_2 - z_i\right) * y_i = 0 \\ \sum 2\left(a_0 * x_i + a_1 * y_i + a_2 - z_i\right) = 0 \end{cases} \tag{10}$$

If the Eq. (10) are changed to matrix form, it is:

$$\begin{vmatrix} \sum x_i^2 & \sum x_i y_i & \sum x_i \\ \sum x_i y_i & \sum y_i^2 & \sum y_i \\ \sum x_i & \sum y_i & n \end{vmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum x_i z_i \\ \sum y_i z_i \\ \sum z_i \end{bmatrix} \tag{11}$$

Solve the equation set of the above formula, obtain the parameter a_0, a_1, a_2, and substitute them to obtain the plane equation. After solving the plane equation, calculate the projection of the point and its neighborhood points in the plane. Assuming that the sampling point is a point p, the vertical line of the plane can be fitted through the point p. Assuming the vertical foot is $p'(x', y', z')$, then the line pp' is parallel to the normal vector $\vec{n}$ of the plane, and the plane normal vector $\vec{n} = (A, B, C)$ can be obtained from the plane equation. The parameter equation of p is as follows:

$$\begin{aligned} x &= x_i - At \\ y &= y_i - Bt \\ z &= z_i - Ct \end{aligned} \tag{12}$$

Bring the point (x', y', z') into the plane equation to obtain t:

$$t = \frac{Ax_i + By_i + Cz_i + D}{A_2 + B_2 + C_2} \tag{13}$$

Then substitute t into Eq. (12) to obtain the coordinates of the projection point p'.

The next step is to filter the edge points from the projection point set. After calculating the coordinates of the projection point, assume that the set of projection points of k neighborhood points of the point is $p_k' = \{p_0', p_1', \ldots, p_{k-1}'\}$, construct $p'p_0', p'p_1', \ldots, p'p_{k-1}'$ a total of k vectors as this section to extract the boundary idea, and determine whether point p is a boundary point by comparing the included angle between k vectors.

The process of solving the vector angle is as follows:

1) Take point p' as the starting point, take all points in the set of points p_k' as the ending point, create k vectors, and select any one of them as the starting vector. For the convenience of calculation, this paper takes $p'p_0'$ as the starting vector, and first calculates the vector product of vector $p'p_0'$ and plane normal vector as vector 1.
2) Obtain the angles α_i and β_i between vector $p'p_i'$ and vector 1 respectively with $p'p_i' (i = 1,2,\ldots,k)$ in turn. If $\beta_i > 90°$, then $\alpha_i = 360 - \alpha_i$.
3) Arrange the α_i obtained in (2) in ascending order, and calculate the angle γ_i between the two vectors $p'p_0'$ and $p'p_i' (i = 1,2,\ldots,k)$, according to α_i, as shown in Eq. (3-14):

$$\gamma_i = \begin{cases} \alpha_i \ (i = 1) \\ \alpha_i - \alpha_{i-1} \ (i = 2,3,\ldots,k-2) \\ 360 - \alpha_i \ (i = k-1) \end{cases} \tag{14}$$

4) Compare the value γ_i calculated in (3), take the maximum value and compare it with the set angle threshold. If it is greater than the set threshold, then the corresponding point p_i' is the boundary point and added to the boundary point set.

3.3 Key Feature Point Extraction

Observe the normal vectors in different areas of the point cloud, and the place where the normal vector changes slowly is relatively flat. The places where the normal vector changes rapidly are steep and undulating. Define the characteristic degree at a point p_i, that is, the change trend

of the normal vector, as the arithmetic mean value of the angle between the normal vector at the point p_i and the normal vector at the k nearest neighbor point is Eq. (16), where p_j is the nearest neighbor point of point p_i and the angle between the normal vector of point p_i and p_j is θ_{ij} .

$$f_i = \frac{1}{k}\sum_{j=1}^{k}\theta ij \tag{15}$$

To sum up, the greater the change trend is, the greater the degree of local characteristics is, so we can use this feature to extract the key feature points of point cloud. Select a more appropriate threshold ε_1 , filter out the relatively flat area in the point cloud, and take $f_1 > \varepsilon_1$ as the condition of the reserved point. If it is satisfied, it is reserved. For any point p_m that meets the reserved condition, if there is:

$$f(p_m) = \max\left[f(p_{m1}), f(p_{m2}), \ldots, f(p_{mk})\right] \tag{16}$$

Then p_m will be the characteristic point, and the characteristic degree of the k nearest neighbor of point p_m is $f(p_{m1}), f(p_{m2}), \ldots, f(p_{mk})$, p is point cloud data, and obtain the data set $p_i = \{p_{i1}, p_{i2}, \ldots, p_{im}\}$ of p by feature extraction.

The point cloud compression is realized by combining the nearest neighbor point of the center point of the point cloud obtained above, the point cloud boundary and the extracted feature points.

4 POINT CLOUD VEHICLE RECOGNITION ALGORITHM BASED ON IMAGE MAPPING

4.1 Image Mapping Point Cloud Vehicle Recognition

This paper proposes an image-mapping-based point cloud vehicle recognition algorithm. First, the YOLO algorithm is used to preliminarily recognize vehicles in the image, and then the calibration matrix and projection relationship between different acquisition devices are used to establish the mapping between the image and the compressed point cloud data. The target points are filtered through these mappings, and then the K-means clustering algorithm is used for optimization, finally recognizing the vehicle targets in the point cloud scene. The flowchart of this method is shown in Fig. 4-1.

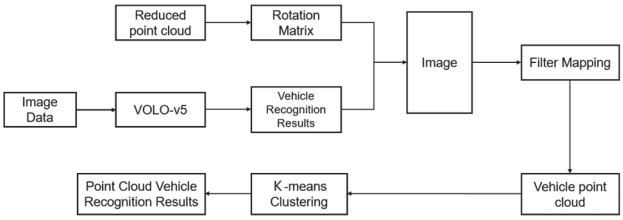


Figure 4-1 Block diagram of point cloud vehicle recognition based on image mapping

The detailed recognition methods in this paper are: (1) image vehicle recognition based on YOLO, (2) mapping between point cloud data and image, (3) point cloud target detection based on K-mean clustering. In this paper, YOLO-v5 is used as the basic network of image recognition, and the pixel points in the image target bounding box are corresponding to the point cloud data, so as to narrow the range of the point cloud target, and the vehicle point cloud is filtered through the vehicle target bounding box output by logistic regression [25]. After mapping from image to point cloud, there will inevitably be interference data in the point cloud data, so it is necessary to process the mapped point cloud data. K-means algorithm is used to cluster the mapped point cloud and identify the vehicles in the point cloud data.

This paper also uses the Darknet-53 network to extract the features of the input image, and uses the residual module to connect the features of different layers to prevent the loss of effective information and the vanishing gradient [26].

To better evaluate the proposed algorithm for mapping image-based convolutional vehicle recognition to point cloud object recognition, the KITTI dataset's image data is used as training data for the convolutional network. First, the image data in the KITTI dataset is converted from png format to VOC format, then input into the YOLO network model, and finally, the converted data is calibrated using the calibration files in the KITTI dataset.

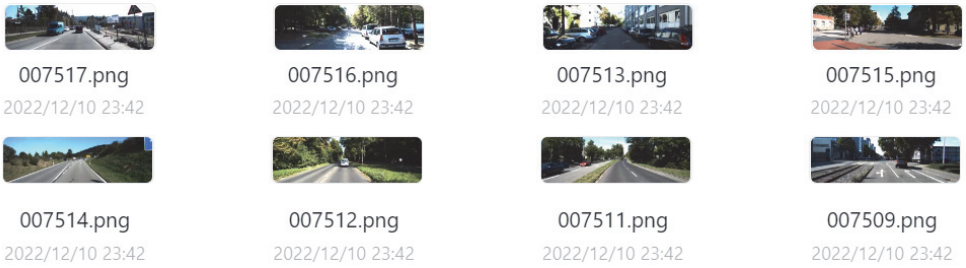


Figure 4-2 Pictures of KITTI data set in PNG format

Figure 4-3 Pictures of KITTI data set in JPG format

car 0.00 0 -1.56 564.62 174.59 616.43 224.74 1.61 1.66 3.20 -0.69 1.69 25.01 -1.59
car 0.00 0 1.71 481.59 180.09 512.55 202.42 1.40 1.51 3.70 -7.43 1.88 47.55 1.55
car 0.00 0 1.64 542.05 175.55 565.27 193.79 1.46 1.66 4.05 -4.71 1.71 60.52 1.56

Figure 4-4 Vehicle marking data

Only the vehicle targets in the scene need to be identified. Therefore, the labelled data of the vehicles in the scene are retained and other non-vehicle data are deleted when labelling the image, as shown in Fig. 4-4. In addition, it is also necessary to convert the label format to *xml* format of VOC. For the converted bounding box center coordinate (x, y), calculation is shown in Eqs. (17) and (18). The size (w, h) of the bounding box is calculated using Eqs. (19) and (20).

$$x = (x_{\min} + x_{\max}) / 2 \tag{17}$$

$$y = (y_{\min} + y_{\max}) / 2 \tag{18}$$

$$w = (x_{\max} - x_{\min}) \tag{19}$$

$$h = (x_{\max} - x_{\min}) \tag{20}$$

According to the conversion formula calibrated during the acquisition of the KITTI dataset, the projection of the target's coordinates $X = (x, y, z, 1)^T$ under the point cloud coordinate system to the image pixel coordinate $Y = (u, v, 1)^T$ follows the Eq. (21).

$$Y = P_{\text{rect}}^{(i)} R_{\text{rect}}^{(0)} T_{\text{velo}}^{\text{cam}} X \tag{21}$$

Use this formula to establish the one-to-one correspondence between points and image pixels of point cloud data, where, $P_{\text{rect}}^{(i)}$ is the internal parameter matrix representing the ith camera, $R_{\text{rect}}^{(0)}$ is the correction matrix from each camera to camera 0, $T_{\text{velo}}^{\text{cam}}$ is the rotation and translation matrix from camera to laser radar. Mapping image pixels to point cloud data requires row and column expansion of each matrix. X represents the homogeneous coordinate form of point cloud data, and $T_{\text{velo}}^{\text{cam}}$ is the external parameter matrix of laser radar and camera obtained through calibration, including rotation matrix and translation matrix.

$$\alpha \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} p_{00} & p_{01} & p_{02} & p_{03} \\ p_{10} & p_{11} & p_{12} & p_{13} \\ p_{20} & p_{21} & p_{22} & p_{23} \end{pmatrix} * \begin{pmatrix} r_{00} & r_{01} & r_{02} & r_{03} \\ r_{10} & r_{11} & r_{12} & r_{13} \\ r_{20} & r_{21} & r_{22} & r_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} t_{00} & t_{01} & t_{02} & t_{03} \\ t_{10} & t_{11} & t_{12} & t_{13} \\ t_{20} & t_{21} & t_{23} & t_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{22}$$

Using the transformation matrix between the LIDAR and the camera, points in the point cloud data can be mapped to image coordinates as shown in Eq. (22). By inputting the image with object recognition from the YOLO network, the point cloud can be mapped to an image with target bounding boxes. After coordinate transformation, pixels within the bounding boxes can be determined, and the corresponding points in the point cloud data can be obtained through mapping. The coordinates of the point cloud mapped to the image within the bounding box can be used to calculate the 3D bounding box of the vehicle in the point cloud data. This method enables high recognition rates for detecting the position of the point cloud vehicle target. However, due to the presence of redundant pixels in the image bounding box, there may be errors when mapping to point cloud data, necessitating point cloud data optimization.

The detection boundary frame of YOLOv5 can be decoded from the output feature map and a priori frame. The boundary frame decoding diagram is shown in Fig. 4-6. (b_x, b_y) can be obtained by boundary box decoding. To obtain the true center coordinates under the image coordinate system, it needs to multiply by the sampling rate of the feature map. Then, traverse the pixel points in the boundary box to determine whether they are pixel points mapped by the point cloud, and count the point clouds mapped in the image range by the mapping matrix Eq. (4-6).

4.2 Point Cloud Vehicle Clustering Based on K-means

K-means algorithm is a relatively simple unsupervised clustering algorithm. The input of the algorithm is a sample

set. By randomly selecting k initial cluster center, calculate the distance between each sample and the cluster center, divide the sample set into k subsets, and output the cluster results when the division of the subsets meets certain requirements. This chapter takes point cloud set as input, takes Euclidean distance as clustering condition, and uses the results of two-dimensional image vehicle recognition as a priori knowledge to cluster vehicles that meet the clustering conditions in the point cloud scene.

K-means algorithm is to divide the sample set into k clusters according to the size of the distance between sample points, and then use iterative method to solve the minimum square error. Details are shown in Eq. (23).

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} \left\| X - U_i \right\|_2^2 \tag{23}$$

$$ui = \frac{1}{\left| C_i \right|} \sum_{x \in C_i} x \tag{24}$$

where, u_i is the mean vector of C_i, and the expression is shown in Eq. (24). Calculate the distance from each point to all center points, and classify them into the cluster of the nearest center point through judgment. Then iterate several times, calculate the center point of the new cluster several times, and repeat the previous steps until the same iteration results appear continuously. In the K-means algorithm, the determination of the initial mean vector k value and the selection of the location will affect the result of vehicle clustering and the running time. The commonly used methods to determine k value include elbow method and contour coefficient method. The elbow method uses the sum of squares of errors to determine the most appropriate value of k, such as Eq. (25). With the increase of k value, the sample division will become more and more refined, and the aggregation degree of each cluster will become higher and higher. The core index of the contour coefficient method is the contour coefficient, as shown in Eq. (26). The nearest cluster is determined by the contour coefficient.

$$SSE = \sum_{i=1}^{k} \sum_{p \in Ci} \left| p - m_i \right|^2 \tag{25}$$

$$S = \frac{b - a}{\max\left(a, b\right)} \tag{26}$$

The k value and k mean vectors are determined. The process of point cloud vehicle clustering algorithm based on K-means is as follows:

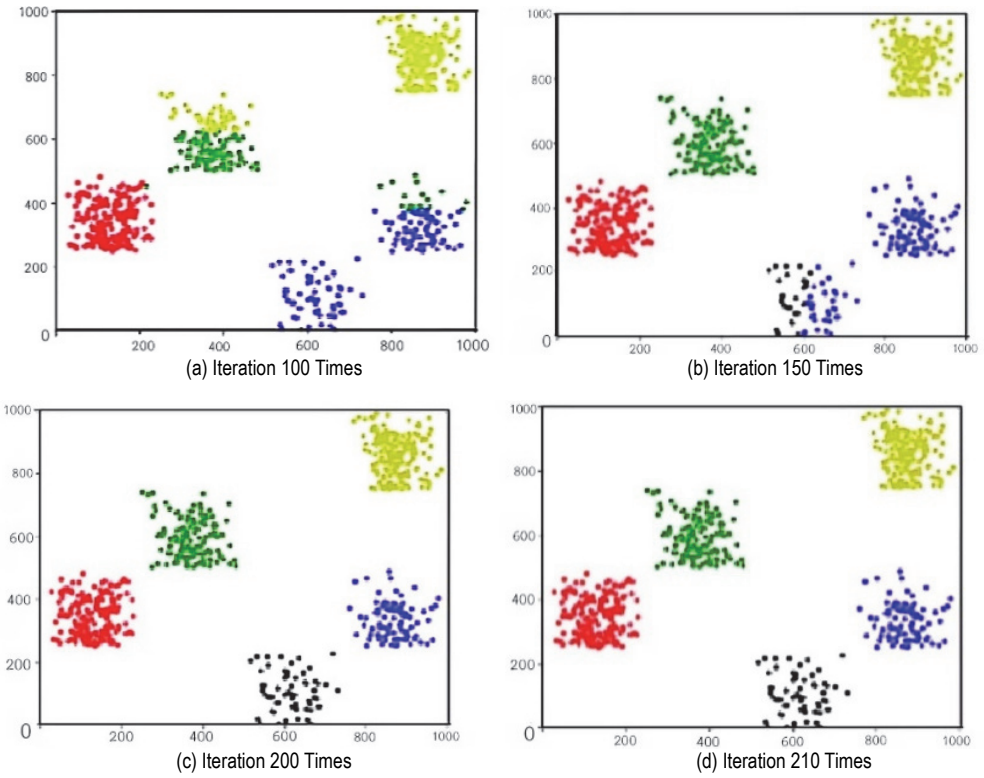


Figure 4-5 K-means clustering process

(1) After mapping the image to the point cloud, determine the input point cloud set $D = \{x_1, x_2, \ldots x_m\}$, and use K-means algorithm to cluster vehicles on the point cloud.

(2) Then select the appropriate k value. This chapter uses the prior knowledge of image recognition to select the number of final bounding boxes in image vehicle recognition as the k value.

(3) Select k mean vectors $\{u_1, u_2, \ldots u_k\}$ from the point cloud set as the center of clustering. This paper randomly selects the points in the image bounding box mapped as point clouds as the mean vector of clustering, and selects one mapping pixel point in each bounding box.

(4) The point cloud set is iteratively divided and initialized to $C=\{C_1,C_2,\ldots C_k\}$, and the distance between the sample point $x_i(i=1,2,3,\ldots m\}$ and the initial mean vector $=u_j\{j=1,2,3,\ldots,k\}$ of the input point cloud is calculated, as shown in Eq. (27).

$$D_{ij}=\left\|x_i-u_j\right\|_2^2 \tag{27}$$

Put x_i into the category with the smallest distance mean vector, and then update the cluster.

$$C_{\lambda_i=}C_{\lambda_i}\cup\{x_i\} \tag{28}$$

(5) Recalculate the mean vector for all points in C_j, as shown in Eq. (29).

$$u_j=\frac{1}{\left|C_j\right|}\sum_{x\in C_j}x \tag{29}$$

(6) Judge whether all mean vectors have changed. If there is no change, output the clustered point cloud $C=\{C_1,C_2,\ldots C_k\}$.

5 EXPERIMENTAL RESULTS AND ANALYSIS

5.1 Block Diagram of the Whole Process of the Experiment in This Paper

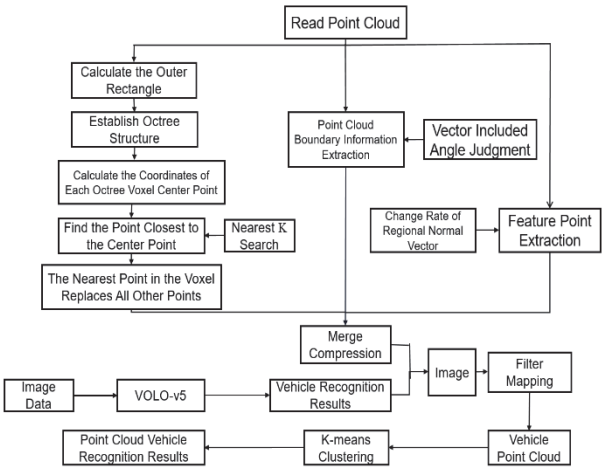


Figure 5-1 Block Diagram of the Whole Process of the Experiment

5.2 Compression Test and Result Analysis

The experiment was completed using C++ language on the platform of visual studio 2019. Part of the content was combined with the PCL library. A set of road vehicle point cloud data (in the format of pcd file, file name: 07510.pcd, 07513.pcd, total points of 456803 and 53101 respectively) in the KITTI dataset was used as the experimental subject. This group of data is named Scene 1 and Scene 2. The platform tmc13 is used, and the traditional uniform grid method, curvature compression method and the method in this paper are used for compression comparison.

The experimental environment configuration is shown in Tab. 5-1, and the compression results of scene 1 and scene 2 using different method at different compression rates are shown in Tabs. 5-2, 5-3, 5-4, 5-5:

Table 5-1 Compression experiment environment configuration

Name	Type
Processor	Inter(R) Core(TM) i5-9300HF CPU @2.40GHz 2.40GHz
Graphics Card	NVIDIA GeForce GTX1650
Reference Software	TMC13-v3.0
Test Sequence	Scene 1, Scene 2
Configuration	positionQuantizationScale:1, bitdepth:8
Software	Visual Studio 2019

Table 5-2 Compression results of scene 1 using uniform grid method at different compression rates

Uniform Grid Method				
Points	Surface	MSE	PSNR	Compression
45669	27.4421	$3.02555e^{-06}$	20.7325	10%
136721	29.2012	$1.05521e^{-06}$	25.7119	30%
228398	29.3212	$5.86667e^{-07}$	27.8659	50%
318887	29.7295	$3.05516e^{-07}$	31.5438	70%
410329	29.8316	$4.37981e^{-08}$	39.6262	90%

Table 5-3 Compression results of scene 1 using the method in this paper at different

Algorithm in This Paper				
Points	Surface Area	MSE	PSNR	Compression Rates
45588	27.1443	$2.9211e^{-06}$	20.8856	10%
136698	27.8698	$9.62849e^{-07}$	25.8079	30%
227860	29.3883	$5.40401e^{-07}$	28.2135	50%
319201	29.6846	$2.50969e^{-07}$	31.9908	70%
410101	29.6386	$3.286670e^{-08}$	40.3731	90%

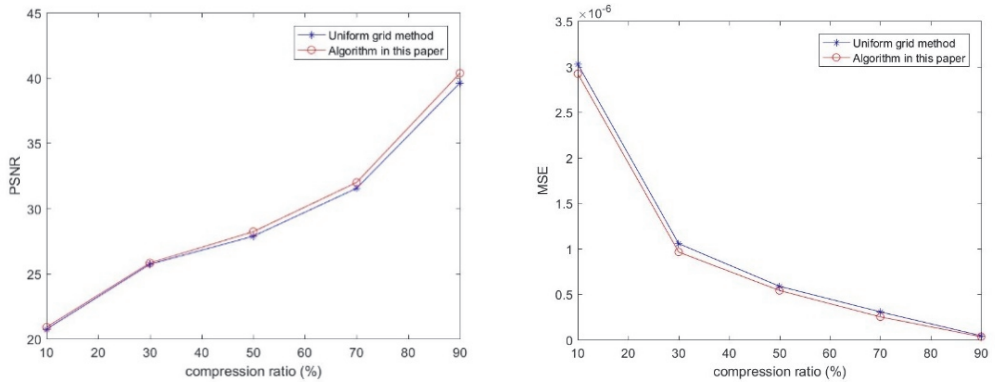
Table 5-4 Compression results of scene 2 using random sampling method at different compression rates

Random Sampling Method				
Points	Surface Area	MSE	PSNR	Compression Rates
5320	85.4081	$2.44531e^{-04}$	29.5181	10%
15999	85.2071	$7.81383e^{-05}$	34.4739	30%
26559	84.0951	$4.1271e^{-05}$	37.2471	50%
37201	83.2437	$2.00861e^{-05}$	40.3742	70%
47701	79.0281	$4.26081e^{-06}$	47.1071	90%

Table 5-5 Compression results of scene 2 using the algorithm in this paper at different compression rates

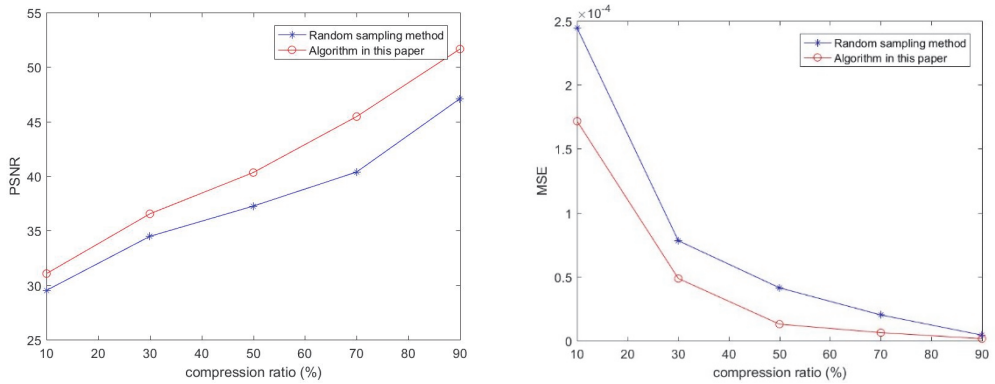
Algorithm in This Paper				
Points	Surface Area	MSE	PSNR	Compression
5289	88.2821	$1.71652e^{-04}$	31.0558	10%
15951	85.3402	$4.85769e^{-05}$	36.5398	30%
26898	84.3109	$1.28160e^{-05}$	40.3261	50%
36991	80.2485	$6.22619e^{-06}$	45.4611	70%
47589	79.3971	$1.49447e^{-06}$	51.6579	90%

A higher PSNR value indicates a smaller loss of the point cloud model during the compression process, resulting in less distortion. The degree of matching with the original point cloud model determines the compression quality, with a higher degree of matching indicating higher compression quality. A smaller MSE value indicates better point cloud compression performance. Based on the results, the proposed algorithm in this paper outperforms the uniform grid method and random sampling method, regardless of the compression rate. The advantage becomes more prominent as the compression rate increases, with the largest advantage observed at a compression rate of 90%. Compared to the uniform grid algorithm, the proposed algorithm improves the PSNR by about 1.88%, and by about 9.66% when compared to the random sampling method.



(a) The contrast of PSNR (b) The contrast of MSE

Figure 5-2 Comparison of PSNR and MSE of uniform grid method and the method in this paper under the same compression rate



(a) PSNR Comparison (b) MSE Comparison

Figure 5-3 Comparison of PSNR and MSE under the same compression rate between the random sampling method and the method in this paper

5.3 Vehicle Recognition Experiment and Result Analysis

The experiment is conducted using the YOLO-v5 algorithm. The results show that the YOLO-v5 algorithm has a relatively high recognition rate for vehicle targets in images, with significant advantages compared to most recognition networks that use point clouds as input. As shown in Fig. 5-4, the YOLO-v5 algorithm can achieve high vehicle target recognition accuracy when recognizing vehicle targets in image data from different scenarios.

In the experiment, a point cloud corresponding only to the image and excluding other parts was used. The mapping from point cloud to image was achieved through the given transformation relationship between the LiDAR and camera in the KITTI dataset. The result of the point cloud mapped to the image coordinates after coordinate transformation is shown in Fig. 5-5.

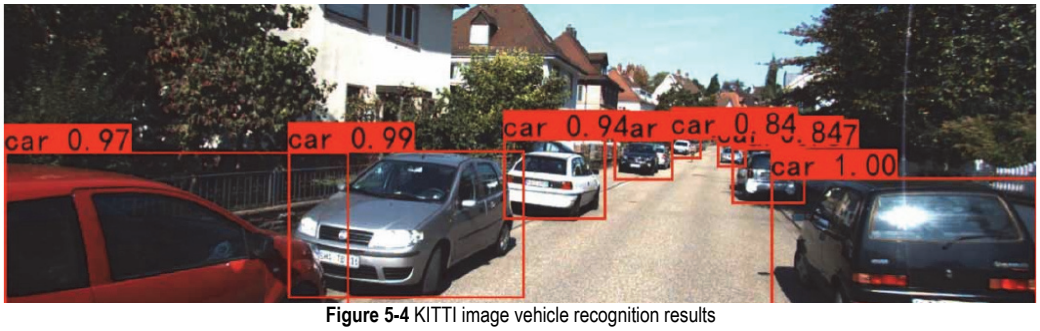


Figure 5-4 KITTI image vehicle recognition results

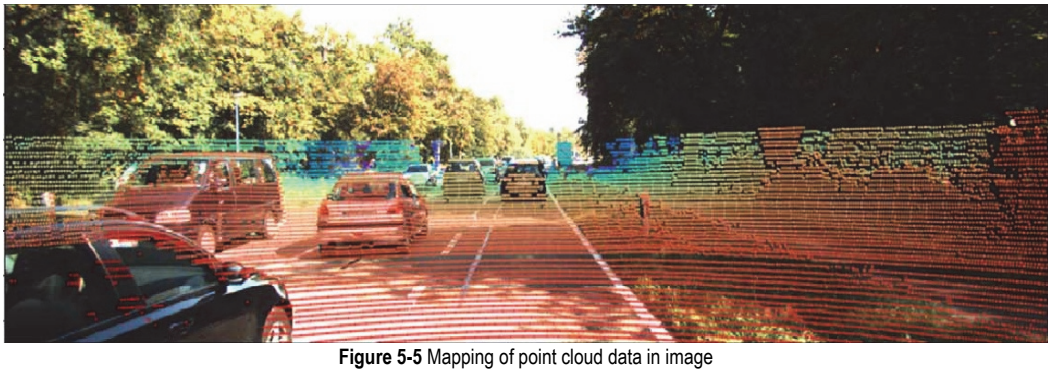


Figure 5-5 Mapping of point cloud data in image

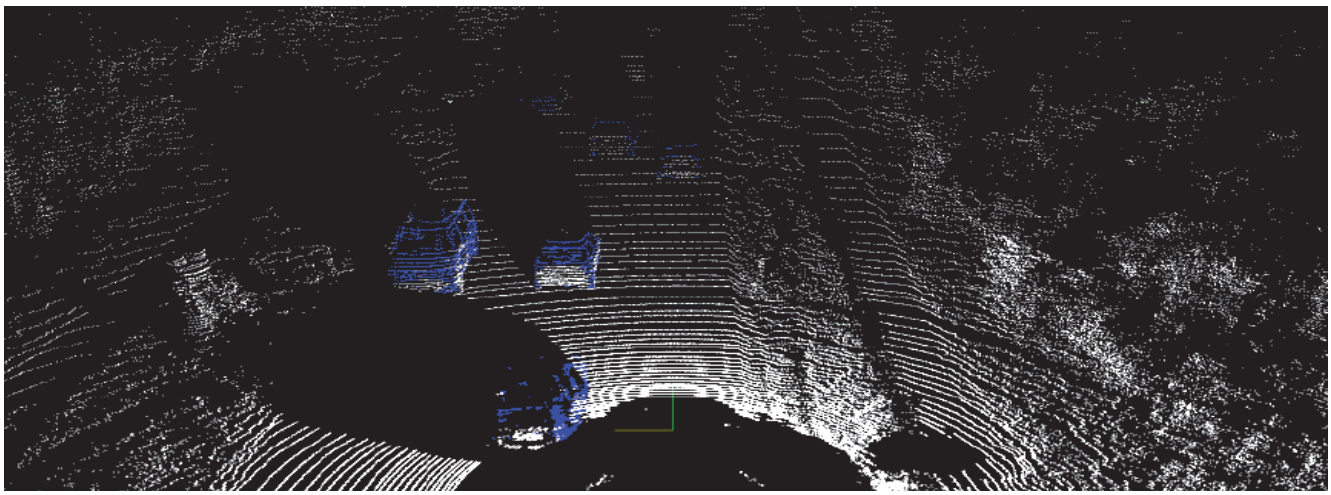
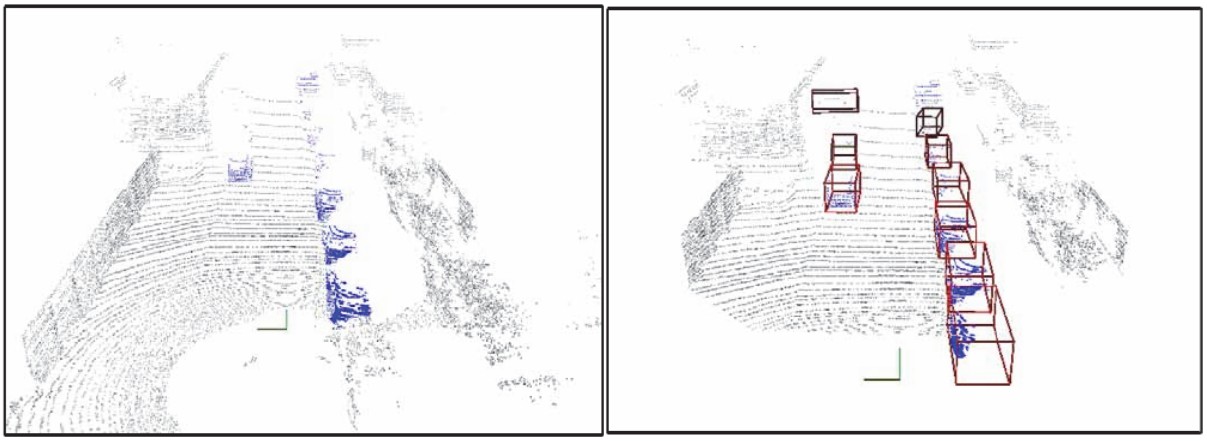


Figure 5-6 Image pixel mapping to point cloud data

Using the rotation matrix between point cloud data and images, the point cloud data can be filtered through the prior bounding boxes of the YOLO algorithm, where the blue points are the point clouds mapped from the vehicle pixel points in the image, as shown in Fig. 5-6.

Sometimes targets overlap, causing bounding boxes to be duplicated during recognition, resulting in the loss of point cloud target positions. In addition, the recognition process will also include other interferences, such as noise from vegetation, steps, and the ground. Therefore, the K-means clustering algorithm is used to optimize the targets after point cloud filtering, preserving vehicle information, reducing interference from other noise in the environment, and improving the position recognition accuracy of point cloud vehicles.



a) Vehicle point cloud clustering b) Calculate bounding box

Figure 5-7 Vehicle point cloud clustering and bounding box calculation

As shown in Fig. 5-7 above, after iterative calculation using K-means clustering algorithm, the vehicle clustering results mapped from image pixels to point cloud data are obtained, and the mean vector of K vehicle clusters is also obtained.

5.3.1 Comparison of Experimental Results of Different Algorithms

To verify the effectiveness of the algorithm for point cloud vehicle recognition in this chapter, we compared it with point cloud vehicle recognition algorithms using voxel division, such as Point-pillars, vehicle recognition algorithms fusing point cloud data and images, such as MV3D, and image-based vehicle recognition algorithms like YOLO. The vehicle recognition accuracy of different algorithms is shown in Tab. 5-1.

Table 5-1 Comparison of recognition accuracy of different algorithms for KITTI data set

Algorithm	Data	Simple	Medium	Difficult
Point-pillars	Point Cloud	83.10%	73.55%	69.10%
YOLO-v5	image	97.08%	89.07%	88.48%
MV3D	Point Cloud + Image	75.04%	64.05%	54.33%
This Paper	Point Cloud + Image	93.01%	88.01%	81.66%

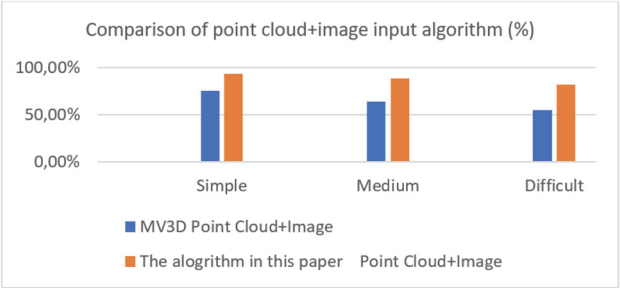


Figure 5-8 Comparison of point cloud + image algorithm recognition accuracy

In comparison to the MV3D algorithm that employs point cloud data and images as input, the algorithm presented in this chapter has improved recognition accuracy by 17.97%, 23.96%, and 27.33% in simple, medium, and difficult scenarios, respectively. However, when compared to the image-based YOLO-v5 algorithm, the method outlined in this paper exhibits slightly lower recognition accuracy in different complexity scenes. This is because the method compresses point cloud data in point cloud scenes and then maps it to the image for recognition, resulting in some data loss that affects subsequent recognition accuracy. In contrast, YOLO-v5 recognizes images directly without the compression step, leading to higher recognition accuracy than the method presented in this paper.

6 CONCLUSIONS

Recognizing and perceiving the environment and targets on the road is a crucial module in autonomous driving technology, and its speed is directly related to the safety of autonomous vehicles and passengers. This paper proposes a vehicle recognition study based on boundary extraction of point cloud compression algorithms. In this compression method, the nearest neighbour of the octree voxel centre point is used to replace other points, extracting point cloud boundaries and feature points, and the three are combined to achieve point cloud compression. Then, a point cloud vehicle recognition algorithm based on image mapping is used for recognition. Experimental results indicate that the algorithm in this paper improves the compression accuracy by approximately 3.19%, and the recognition accuracy in simple, medium, and difficult scenarios is increased by 17.97%, 23.96%, and 27.33%, respectively. However, this research has some shortcomings. The training samples collected in this paper are insufficient to cover various types of vehicles that may appear in real road scenarios. Therefore, it is necessary to expand the training samples, increase more target types, and include samples of different targets in more complex scenarios to reduce missed detections and misidentifications. Additionally, the algorithm needs optimization to improve the detection and recognition of distant targets.

Acknowledgement

This research was funded by Guangdong Provincial Department of Education New Generation Information Technology Key Special Field Fund project: Research on Key Technologies of Encoding and Decoding of Real-time Vehicular Lidar Point Cloud Sequences, grant number 2021ZDZX1123; and the innovation team of Big Data Analysis and Decision of Discrete Manufacturing project, grant number 2022KCXTD064.

7 REFERENCES

[1] Yiming, Z., Yanhua, L., Yanan, S. et al. (2014). Application and Development Trend of Laser Radar. *Telemetry and Telecontrol*, *35*(05), 4-22.

[2] Hu, F. J. & Zhao, Y. W. (2013). An Improved Method of Discrete Point Cloud Filtering based on Complex Environment. *International Journal of Applied Mathematics and Statistics*, *48*(18), 514-522.

[3] Wiegand, T., Sullivan, G. J., Bjøntegaard, G. et al. (2003). Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *13*(7), 560-576. https://doi.org/10.1109/TCSVT.2003.815165

[4] Sullivan, G. J., Ohm, J., Han, W. et al. (2012). Overview of the High Efficiency Video Coding (HEVC)standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *22*(12), 1649-1668. https://doi.org/10.1109/TCSVT.2012.2221191

[5] Budagavi, M., Faramarzi, E., Ho, T. et al. (2017). *Samsungs Response to CFP for Point cloud Compression (Category 2), Document m41808*. Macau, China, ISO/IEC JTC1/SC29/WG11 MEPG

[6] He, L. Y., Zhu, W. J., & Xu, Y. L. (2017). Best-effort Projection Based Attribute Compression for 3D Point Cloud. *2017 23rd Asia-Pacific Conference on Communications (APCC)*, 1-6. https://doi.org/10.23919/APCC.2017.8304078

[7] MPEG-3DG. Text of ISO/IEC CD 23090-9: Geometry-based Point Cloud Compression, Document N18478. Geneva: ISO/IEC JTC1/SC29/WG11 MEPG, 2019.

[8] Ruwen, S. & Klein, R. (2006). Octree-based Point-Cloud Compression. *PBG@SIGGRAPH, Euro Graphics*, 111-120.

[9] Garcia, D. C., Fonseca, T. A., Ferreira, R. U. et al. (2020). Geometry Coding for Dynamic Voxelized Point Clouds Using Octrees and Multiple Contexts. *IEEE Transactions on Image Processing*, *29*, 313-322. https://doi.org/10.1109/TIP.2019.2931466

[10] Shifei, L. (2010). *Research on 3D Target Recognition Technology Based on Depth Image*. National University of Defense Technology.

[11] Sheng, X. (2010). *Research on 3D Object Recognition*. University of Electronic Science and Technology of China.

[12] Alhamzi, K., Elmogy, M., & Barakat, S. (2014). 3D Object Recognition Based on Image Features: A Survey. *Jilin University*, *2014*(3), 651-660.

[13] Alhamzi, K., Elmogy, M., & Barakat, S. (2015). 3D Object Recognition Based on Local and Global Features Using Point Cloud Library. *Jilin University*, *7*, 43-54.

[14] Alrawabdeh, A., He, F., Mousaa, A. et al. (2016). Using an Unmanned Aerial Vehicle-Based Digital Imaging System to Derive a 3D Point Cloud for Landslide Scarp Recognition. *Remote Sensing*, *8*(2), 95. https://doi.org/10.3390/rs8020095

[15] Czerniawski, T., Nahangi, M., Haas, C. et al. (2016). Pipe spool Recognition in Cluttered Point Clouds Using a Curvature-based Shape Descriptor. *Automation in Construction*, *71*, 346-358. https://doi.org/10.1016/j.autcon.2016.08.011

[16] Kasaei, S. H., Tom, É. A. M., Lopes, L. S. et al. (2016). GOOD: A Global Orthographic Object Descriptor for 3D Object Recognition and Manipulation. *Pattern Recognition Letters*, *83*, 312-320. https://doi.org/10.1016/j.patrec.2016.07.006

[17] Roberts, L. G. (1963). Machine perception of three-dimensional solids. *Optical and Electro-Optical Information Processing Cambridge, MA, USA, MIT Press*, *20*, 31-39.

[18] Yang, M. H. (2009). Object Recognition. *Encyclopedia of Database Systems*, 1936-1939. https://doi.org/10.1007/978-0-387-39940-9_1042

[19] Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Viewpoints. *International Journal of Computer Vision*, *60*, 91-111. https://doi.org/10.1023/B:VISI.0000029664.99615.94

[20] Guo, Y., Bennamoun, M., Sohel, F. et al. (2014). 3D Object Recognition in Cluttered Scenes with Local Surface Features: A Survey. *Pattern Analysis & Machine Intelligence IEEE Transactions*, *36*(11), 2270-2287. https://doi.org/10.1109/TPAMI.2014.2316828

[21] Rusu, R. B., Bradski, G., Thibaux, R. et al. (2010). Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram. *International Conference on Intelligent Robots and Systems, IEEE*, 2155-2162. https://doi.org/10.1109/IROS.2010.5651280
[22] Aldoma, A., Vincze, M., Blodow, N. et al. (2011). CAD-model Recognition and 6DOF Pose Estimation Using 3D Cues. *IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops*, 585-592. https://doi.org/10.1109/ICCVW.2011.6130296
[23] Guo, Y., Bennamoun, M., Sohel, F. A. et al. (2013). 3D Free Form Object Recognition Using Rotational Projection Statistics. *IEEE Workshop on Applications of Computer Vision. IEEE Computer Society*, 1-8. https://doi.org/10.1109/WACV.2013.6474992
[24] Guo, Y., Bennamoun, M., Sohel, F. et al. (2016). A Comprehensive Performance Evaluation of 3D Local Feature Descriptors. *International Journal of Computer Vision*, *116*(1), 66-89. https://doi.org/10.1007/s11263-015-0824-y
[25] Fukai, Z., Feng, Y., & Ce, L. (2019). Fast Vehicle Detection Method Based on Improved YOLOv3. *Computer Engineering and Application*, *55*(02), 12-20.
[26] Shuai, F., Long, Z., & Xiaohui, H. (2017). Pedestrian detection based on Jetson TK1 and deep convolution neural network. *Information Technology*, *2017*(10), 62-64+68.
[27] Woźniak, M. (2021). *Advanced Computational Intelligence for Object Detection, Feature Extraction and Recognition in Smart Sensor Environments*. MDPI - Multidisciplinary Digital Publishing Institute. https://doi.org/10.3390/s21010045
[28] Anwaar, U. et al. (2022). *Advances in Object and Activity Detection in Remote Sensing Imagery*. MDPI - Multidisciplinary Digital Publishing Institute. https://doi.org/10.3390/books978-3-0365-4230-0
[29] Esraa, K. et al. (2022). Evaluation of 3D Vulnerable Objects' Detection Using a Multi-Sensors System for Autonomous Vehicles. *Sensors*, *22*(4), 2022, 1663. https://doi.org/10.3390/s22041663
[30] Kashevnik, A. & Ammar, A. (2022). 3D Vehicle Detection and Segmentation Based on EfficientNetB3 and CenterNet Residual Blocks. *Sensors*, *22*(20), 7990. https://doi.org/10.3390/s22207990

Contact information:

Yanjun ZHANG
Zhong Shan Polytechnic,
China
E-mail: 843418789@qq.com

Pengcheng SONG
Heilongjiang University,
China

Qi JING
Heilongjiang University,
China

Qingyun LI
(Corresponding author)
Zhong Shan Polytechnic,
China
E-mail: Qingyun.li@gmail.com