Real-time Drivable Region Planning
Based on 3D LiDAR

Zewei Wang, Chunnian Zeng, Xu Yang and Jie Luo, Jinmin Hu

ABSTRACT

Stable, real-time, drivable area planning in a dynamic environment is an essential feature of autonomous vehicles. This paper presents an efficient 3D laser radar-based drivable area planning algorithm. In order to extract the drivable area, the original point cloud data is first downsampled to obtain a relatively sparse point cloud to reduce the complexity. Then, based on the geometric features of the pavement point and dividing the road plane, the obstacle point after the road plane is divided. The cloud is transformed into a 2D aerial view, and a series of expansion, convolution and other regional operations are performed on the bird's-eye view, and the road edge points are extracted, and the curve fitting is performed based on the least squares method to plan the drivable region. The algorithm proposed in this paper was tested on the KITTI dataset and obtained robust, high-efficiency experimental results.

KEYWORDS

3D LiDAR, Downsample, Drivable region planning, Bird’s eye view.

INTRODUCTION

Environment perception is one of the indispensable technologies for mobile robots and self-driving cars. It stores the environmental information around the vehicle in a digital way and analyzes and calculates it through the onboard computer to provide information for decision making and path planning of smart cars. Traffic scene perception can be divided into object detection and road and lane detection. Object detection is mainly used to identify dynamic objects or traffic signs in traffic scenes, including pedestrian detection[1], vehicle detection[2] and traffic light/signal detection[3]. Road and lane detection is mainly used for drivable region planning which avoids the deviation of the vehicle from the road. It mainly includes road marking detection[4], lane detection[5] and driving region detection[6-9]. This paper mainly studies the drivable region planning based on 3D LiDAR, which is the basic component of automatic driving and can be used for motion planning of the vehicle.

Drivable region planning can be achieved with a variety of sensors. Among them, camera is widely used as a low-cost sensor. Meanwhile, image can provide rich texture information, which is very suitable for segmentation and recognition of objects in the image through machine learning. However, as a sensor that passively receives light, camera is easily affected by the environment and is prone to dynamic blur in a dynamic environment. LiDAR detects the target by
actively emitting laser light, and can detect the three-dimensional structure of the environment precisely. It is relatively more stable and is not easily interfered by the environment. Therefore, LiDAR is widely used in the development of smart cars. Researchers have accumulated a lot of experience in the application of LiDAR in the development of smart cars. Fernandes[11] realized the detection of road areas by combining 3D laser point clouds into 2D reference planes by combining upsampling and sliding windows. Zhang[10] first processed the original point cloud data to roughly distinguish between road and non-road areas, and then used the sliding window method to perform real-time edge detection and precise road segmentation. Yecheng Lyu[9] et al. constructed the road segmentation problem as a semantic segmentation task using deep neural networks, and designed the algorithm on the FPGA to implement the real-time processing of the point cloud data. Instead of using dedicated hardware, Kiin[8] chose to optimize the algorithm to improve the performance. They combined the model-based and region-based segmentation methods to implement the real-time drivable region detection under complex urban environments. Optimizing the algorithm avoids the use of additional hardware to guarantee real-time processing, thus reduces cost and makes the entire solution more light-weighted.

This paper focuses on the implementation of the driving region detection using 3D LiDAR. By exploring the structural feature, an efficient drivable region planning algorithm is proposed. The method of this paper firstly downsamples the original point cloud to make it relatively more sparse. The experimental results show that reasonable downsampling does not have much influence on the segmentation result of the travelable area, but can significantly increase the running speed of the algorithm. The road plane is then extracted from the downsampled point cloud, and the detailed steps of the road plane detection are given in the section 1.1. After segmenting the road, the remaining point cloud is transformed into a 2D aerial view image. Section 1 describes the details of the processing on the aerial view image. The algorithm proposed in this paper has been tested on the KITTI dataset and the expected experimental results have been obtained.

IMAGE PROCESSING

ROAD PLANE DETECTION

Road plane detection is one of the main contributions in this paper. The LiDAR returns point cloud with high density which contains large amount of data. However, there is a lot of redundancy for the road surface detection. Therefore, the point cloud needs to be downsampled first. In our experiment, we paper downsamples the point cloud with a sampling rate of 20% , and the points behind the back of the vehicle is removed because the drivable region detection needs to be performed directly in front of the vehicle. Through experiments, it can be found that downsampling does not affect the detection of the road surface, but it can greatly speed up the process.

In addition to downsampling, we also makes full use of the geometric information of the point cloud. The geometric characteristics of the lidar points on the road surface and the data points on the obstacles are significantly different. As shown in Figure 1, the coordinate system represents the lidar coordinate system, the thick solid line represents the road surface, the blue fold line is the obstacle, and the red lines are two adjacent laser scan. $p_1$ and $p_3$ are the
points on the road plane and \( p_2 \) is the point on the obstacle. We can calculate the angle \( \angle a \) to determine whether the points on the adjacent scan lines are on the same plane. If one of the points falls on the obstacle, such as \( p_2 \), it is obviously that \( \angle a \) will be greater than 0 degrees. If there is no obstacle, the LiDAR will scan along the red dotted line and \( \angle a \) is close to 0 degrees.

Figure 1. Schematic diagram of geometric characteristics of lidar data points.

Select the pairs of points that are evenly distributed in space which makes \( \angle a \approx 0 \). Then using the least squares method to fit the selected points into a plane (1). The road detection algorithm only needs to traverse the point cloud once, which greatly avoids unnecessary computation and can efficiently estimate the plane model.

\[
Z = aX + bY + c
\]  

Figure 2. Image of screening road plane points.

**EXTRACT RANGE FEATURE**

After getting the plane model, we need to separate the point cloud of the road, leaving only the obstacle part, and then transform the point cloud of the obstacle part into a bird's eye view.

The range feature map describes the closest distance \( d \) from each position in the bird's-eye view to the laser point cloud. The larger the \( d \), the larger the corresponding pixel value in the distance feature map, and vice versa, and \( d \) satisfies:

\[
0 \leq d \leq 255
\]  

Taking the scenario of Figure 3 as an example, the corresponding lidar point cloud image of the road surface is shown in Fig. 4, and the corresponding bird's eye view I is shown in Figure 5.
In order to compute the range feature, we need to invert the pixels of the original image $I$ to obtain $J$:

$$J = 255 - I$$  \hspace{1cm} (3)$$

Set the template parameters $m_1, m_2, m_3, m_4$ and then extract the range features by processing two iterations of traversals on the image $J$.

First iteration:
If the image has $M$ rows and $N$ columns, the traversal order is from the second row to the $M-1$ th row, the second column to the $N-1$ th column. For the pixel $p$, the four pixels in the lower right neighborhood are added and the corresponding template parameters are added for comparison. Denote the neighbor pixel as $p'$, and then update the pixel as follows.

$$p_i' = \begin{cases} p_c + m_i, & \text{if } p_c + m_i < p_c \\ p_i, & \text{else} \end{cases}$$  \hspace{1cm} (4)$$

Second iteration:
The direction of the second traversal is opposite to the first one. For the pixel $p$, the four pixels of the upper left neighborhood are updated. The range feature obtained after two traversals are shown in Figure 6.

Figure 3. Sample scenario.

Figure 4. LiDAR point cloud of the sample scenario.
SEPARATE UNOCCUPIED SPACE

Set the threshold value \( t \) and then perform threshold segmentation on the distance feature map obtained in the previous step, set the pixel on the range feature map to 0 if it is larger than \( t \), and set the rest of the pixels to 255 to obtain some white pixel blocks indicating the obstacle.

The white dense pixel blocks of Figure 7 is similar to the dilation operation of Figure 5. The point cloud generated by the LiDAR is too sparse to be directly used for planning. Through the above algorithm, we obtain a two-dimensional dense representation of the obstacle. Next we need to extract the drivable region, as shown in the green part of Figure 7.

The extraction of the drivable region is divided into two steps: (1) Extract the drivable region between the center of the LiDAR coordinate and the nearest obstacle. (2) Obtain the drivable region between the left and right obstacles.

Step 1:
As shown in Figure 8(a), the mapping relationship between a search point \( p_A \) and its mapping point \( p_A' \) is:

\[
\begin{align*}
    p_A &= (x_A, y_A)^T \\
    p_A' &= (x_A', y_A')^T \\
    x_A' &= x_A \\
    y_A' &= \begin{cases}
        M - \frac{(M - y_A)(N - x_A)}{N}, & \text{if } x_A > N \\
        M - \frac{(M - y_A)(x_A - N)}{W - N}, & \text{else}
    \end{cases}
\end{align*}
\]

Where \( W \) is the width of the image. For each of the mapped point \( p_A' \), if the pixel value of \( p_A' \) is 0, the point belongs to the unoccupied space and is marked, otherwise it is considered an obstacle.

Step 2:
As shown in Figure 8(b), the mapping relationship between the search point \( p_B \) and its mapping point \( p_B' \) is:
\[ x'_n = N - \frac{(M - y_n)(N - x_n)}{M} \]  

(9)

\[ y'_n = y_n \]  

(10)

The way in which it marks the unoccupied space is the same as Step 1, and the resulting unoccupied space is as shown in Figure 9.

Figure 6. Range feature.

Green--road region  
White--obstacle

Figure 7. Road region and obstacle region.

Figure 8. Coordinate mapping.
MODEL PARAMETER ESTIMATION

After the unoccupied space is obtained, the edge points on the left and right sides of the unoccupied space are extracted for model parameter estimation.

After obtaining the set of edge points on the left and right sides of the drivable region, the edges are fitted into two functions using the least squares method.

\[ f(x) = \sum_{i=0}^{n} a_i x^i \]  

(11)

For some edge point \( p_i = (x_i, y_i)^T \), the corresponding residual is:

\[ e(x_i, y_i) = x_i - f(y_i) \]  

(12)

Denote the model parameter to be estimated is \( v = (a_0, a_1, \ldots, a_n)^T \), then the total cost is:

\[ S(v) = \frac{1}{2} \sum_{i=0}^{n} w_i (e(x_i, y_i))^2 \]  

(13)

Where \( w_i \) is the weight parameter. By minimizing \( S(v) \), we can get the optimal model parameter \( v^* \):

\[ v^* = \arg\min_v S(v) \]  

(14)

In our experiment we find that \( n = 2 \) is enough for obtaining a good fitting result. So the above problem can be simplified into a linear least squares problem. The matrix form of the problem is as follows:

\[
\begin{pmatrix}
  x_1^2 & x_1 & 1 \\
  x_2^2 & x_2 & 1 \\
  \vdots & \vdots & \vdots \\
  x_n^2 & x_n & 1 \\
\end{pmatrix}
\begin{pmatrix}
  a_2 \\
  a_1 \\
  a_0 \\
\end{pmatrix}
= 
\begin{pmatrix}
  y_1 \\
  y_2 \\
  \vdots \\
  y_n \\
\end{pmatrix}
\]  

(15)

The above equation can be written in the simplified form:
\[ A v = b \]  \hspace{1cm} (16)

And the least-square solution of \( v \) is:

\[
\begin{bmatrix}
    a_2 \\
    a_1 \\
    a_0
\end{bmatrix} = (A^T A)^{-1} A b 
\]  \hspace{1cm} (17)

After obtaining the model parameters, two smooth curves can be planned to limit the drivable region. The result is shown in Figure 10. The two green smooth curves are the curves drawn with the estimated model parameters.

**EXPERIMENT**

**EXPERIMENT DESIGN**

The experiment in this paper uses the KITTI dataset, which collects data using a Velodyne HDL-64E LiDAR. The experiment tests the robustness and efficiency of the algorithm in different scenarios, and compares the efficiency with the methods proposed in [8],[9] and [12]. We then project the drivable region onto the image captured by camera for visual analysis.

**RESULTS AND ANALYSIS**

The results of the comparative experiment are shown in Table 1. As can be seen from Table 1, the method of the proposed method can significantly speed up the detection of the road surface. This is mainly because the road plane detection algorithm in this paper makes full use of the geometric information of the lidar point cloud. It only needs to traverse the point cloud once to mark the data points on the plane, thus avoids a lot of redundant calculation. The paper [8],[9],[12] does not use the geometric information of the lidar point cloud in the preprocessing and so there is still some potential for improvement in speed. Paper [12] takes advantage of the powerful fitting ability of convolutional neural networks. But it requires the use of GPUs which increase the cost of the entire solution.

Figure 11 and Figure 12 shows some result of the proposed method. The green part of Figure 12 is the projection of the drivable region on the image taken by camera. As can be seen from Figure 11 and Figure 12, the algorithm proposed in this paper can effectively avoid obstacles and plan the driving region reasonably.
TABLE I. ALGORITHM EFFICIENCY COMPARISON.

| method     | time required/ms |
|------------|------------------|
| this paper | 5.56             |
| paper[8]   | 16.9             |
| paper[9]   | 30               |
| paper[12]  | 18               |

Figure 10. Model parameter estimation.

Figure 11. Results of the drivable region planning.

Figure 12. Projection of the drivable region on the image.
CONCLUSION

Firstly, the proposed method in this paper makes full use of the geometric characteristics of the LiDAR point cloud in the step of detecting the road plane, avoiding redundant calculation and greatly improving the speed of the overall algorithm.

Secondly, by transforming the 3D point cloud into a bird's-eye view image, our method reduces the data dimension and simplifies the drivable region planning problem into a 2D image processing problem. Through the innovative design of the algorithm, the reasonable planning of the drivable region is realized.

Thirdly, the proposed method can be applied to mobile robots and autonomous driving, and can assist the robot or autonomous driving vehicle in path planning and control decision without significantly increasing the computing time.

REFERENCES

1. Tian Y, Ping L, Wang X, et al. Pedestrian detection aided by deep learning semantic tasks[J]. 2015.
2. Hu Q, Paisitkriangkrai S, Shen C, et al. Fast detection of multiple objects in traffic scenes with a common detection framework[J]. IEEE Transactions on Intelligent Transportation Systems, 2015, 17(4): 1002-1014.
3. Creusen, I. M., L. Hazelhoff, and P. H. N. D. With. "Color transformation for improved traffic sign detection." Image Processing (ICIP), 2012 19th IEEE International Conference on IEEE, 2012.
4. Chen, T., Chen, Z., Shi, Q., & Huang, X.. (2015). Road marking detection and classification using machine learning algorithms. 2015 IEEE Intelligent Vehicles Symposium (IV). IEEE.
5. Dong, Y., Xiong, J., Li, L., & Yang, J.. (2012). Robust lane detection and tracking for lane departure warning. 2012 International Conference on Computational Problem-Solving (ICCP). IEEE.
6. Zou B, Wang L. Based on the Detection and Extraction of the Passable Area of Lidar Road[J]. Automation and Instrumentation, 2018.
7. Wang J, Kong B, Wang C. An approach of real-time road boundary detection based on HDL-64E lidar[J]. Journal of Hefei University of Technology (Natural Science Edition), 2018, 41(8):1029-1034.
8. Na, Kiin, B. Park, and B. Seo. "Drivable space expansion from the ground base for complex structured roads." 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC) IEEE, 2016.
9. Lyu Y, Bai L, Huang X. Real-time road segmentation using lidar data processing on an fpga[C]/2018 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2018: 1-5.
10. Zhang, Y., Wang, J., Wang, X., & Dolan, J. M. (2018). Road-segmentation-based curb detection method for self-driving via a 3d-lidar sensor. IEEE Transactions on Intelligent Transportation Systems, 1-11.
11. Fernandes, R., Premebida, C., Peixoto, P., Wolf, D., & Nunes, U. (2014). Road detection using high resolution LIDAR. IEEE Vehicle Power & Propulsion Conference.
12. Caltagirone, L., Scheidegger, S., Svensson, L., & Wahde, M.. (2017). Fast lidar-based road detection using fully convolutional neural networks.