machines

MDPI

Article

A Robust Gaussian Process-Based LiDAR Ground Segmentation Algorithm for Autonomous Driving

Xianjian Jin [1,2,*], Hang Yang [1], Xin Liao [3], Zeyuan Yan [1], Qikang Wang [1], Zhiwei Li [1] and Zhaoran Wang [1]

[1] School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200072, China; messes999@gmail.com (H.Y.); zeyuan_yan@163.com (Z.Y.); qikang_wang@163.com (Q.W.); zhiweili990618@163.com (Z.L.); wangzhaoran003@163.com (Z.W.)
[2] Shanghai Key Laboratory of Intelligent Manufacturing and Robotics, Shanghai University, Shanghai 200072, China
[3] School of Mechanical Engineering, Shijiazhuang Tiedao University, Shijiazhuang 050043, China; sinna_liaoxin@stdu.edu.cn
* Correspondence: xianjianjin@shu.edu.cn

Abstract: Robust and precise vehicle detection is the prerequisite for decision-making and motion planning in autonomous driving. Vehicle detection algorithms follow three steps: ground segmentation, obstacle clustering and bounding box fitting. The ground segmentation result directly affects the input of the subsequent obstacle clustering algorithms. Aiming at the problems of over-segmentation and under-segmentation in traditional ground segmentation algorithms, a ground segmentation algorithm based on Gaussian process is proposed in this paper. To ensure accurate search of real ground candidate points as training data for Gaussian process, the proposed algorithm introduces the height and slope criteria, which is more reasonable than the use of fixed height threshold for searching. After that, a sparse covariance function is introduced as the kernel function for calculation in Gaussian process. This function is more suitable for ground segmentation situation the radial basis function (RBF). The proposed algorithm is tested on our autonomous driving experimental platform and the public autonomous driving dataset KITTI, compared with the most used RANSAC algorithm and ray ground filter algorithm. Experiment results show that the proposed algorithm can avoid obvious over-segmentation and under-segmentation. In addition, compared with the RBF, the introduction of the sparse covariance function also reduces the computation time by 37.26%.

Keywords: autonomous driving; ground segmentation; sparse covariance; Gaussian process regression

Citation: Jin, X.; Yang, H.; Liao, X.; Yan, Z.; Wang, Q.; Li, Z.; Wang, Z. A Robust Gaussian Process-Based LiDAR Ground Segmentation Algorithm for Autonomous Driving. *Machines* **2022**, *10*, 507. https://doi.org/10.3390/machines10070507

Academic Editor: Antonios Gasteratos

Received: 19 May 2022
Accepted: 22 June 2022
Published: 23 June 2022

1. Introduction

1.1. Background

Autonomous driving technology is rapidly developing, which has been attributed to advances in the sensor, communications and computer industries [1–7]. As the first step in autonomous driving, the results of environmental perception systems directly affect the subsequent motion planning and decision making. An environment perception system based on 3D-LiDAR aims to extract the point cloud of the target object from the disordered input point cloud, which includes ground segmentation, obstacle point cloud clustering and bounding box fitting [8–11]. The results of ground segmentation are irreversibly inputted to subsequent steps for processing in the environmental perception system, and affected the clustering and bounding box fitting process [12,13]. Therefore, the accuracy of ground segmentation is crucial for the environment perception system and autonomous driving technology.

In ground segmentation algorithms, the accuracy of ground segmentation is usually compromised by two incorrect segmentation situations: over-segmentation and under-segmentation [1,5]. Over-segmentation means that obstacle points that do not belong to the ground are divided into the ground points. This type of incorrect segmentation will

reduce the possibility of obstacle detection, and result in the lack of obstacle points. Under-segmentation means some ground points are divided into obstacle points, and this type of incorrect segmentation will affect the vehicle's judgment of the drivable area. Therefore, the optimization of the ground segmentation algorithm is to avoid these two types of incorrect segmentation situations [9].

1.2. Related Work

Commonly used ground segmentation algorithms are divided into two methods: surface-based methods and line-based methods. The surface-based method considers the shape of the ground, regarding it as a plane or a curved surface. In this method, a certain number of ground candidate points are selected to complete the fitting of the plane or curved surface. Depending on whether the ground is regarded as the plane or curved surface, the fitting algorithm used is different. When the ground is regarded as a plane, the random sampling consistency (RANSAC) algorithm and its improved algorithms are mainly adopted [14–18]. Qiu et al. proposed an iterative fitting method of datum surface based on prior distribution information; this strategy can avoid the concentration of initial frame sampling points in a small range, and effectively improve the efficiency of sampling iteration [15]. Li et al. also chose to optimize the sampling point selection process. They introduced the normal distribution transformation (NDT) to convert a certain number of points into NDT cells, and then divided these into planar cells and non-planar cells. In the subsequent sampling process, only those belonging to planar cells were selected to fit [16]. Obviously, the biggest defect of this type of algorithm is that the fitting accuracy is too poor for the ground with changes. Therefore, it is more reasonable to treat the ground as a surface, and then select enough points to fit the surface. B. Douillard et al. used the Gaussian process incremental sample consensus (GP-INSAC) algorithm to complete the curved surface fitting, which first uses the INSAC algorithm to complete the selection of the initial ground candidate points, and then uses the Gaussian process to complete the ground parameter estimation [17]. In this way, fitting the entire ground plane poses a large computational burden. Therefore, the line-based methods are proposed.

The line-based methods mainly consider the scanning characteristics of LiDAR. In this method, the ground is divided into different segments according to the preset angle, and then each segment is divided into different small bins according to the distance [19–30]. By judging the spatial features or other features of the points in each bin, the reference ground height of each bin is obtained to establish the distinction between ground points and non-ground points. Him et al. examined the height difference and slope in different bins to quickly determine the ground height; this algorithm is simple and efficient, but the fitting effect is strongly affected by the fixed height difference thresholds or slope thresholds, which are too sensitive to height changes, and easily leads to over-segmentation [19]. Chen et al. introduced the Gaussian process into this type of algorithm. Compared with the previous GP-INSAC algorithm to fit the entire ground plane, the Gaussian process is simplified to one dimension and is only used to fit the ground reference line within a segment [20]. Luo et al. calculated the probability that each bin belongs to the ground, and proposed a ground segmentation algorithm based on the occupancy probability grid, which has a good segmentation effect on various types of road surfaces [22]. P. Narks et al. designed a segmented fitting strategy, which has better adaptability to slope conditions [23].

In summary, this paper proposed a robust ground segmentation algorithm based on Gaussian process, which adopts the technical route of line-based method. Firstly, we divide the ground coordinate system into several segments, all points in the Cartesian coordinate system are mapped to a certain bin in a segment. After that, ground real points are searched in each segment according to height and slope criteria, which are used as training data for the Gaussian process. Finally, the reference ground height of each bin in the segment is obtained through Gaussian process regression to complete the ground segmentation. The flow chart of this paper is shown in Figure 1.

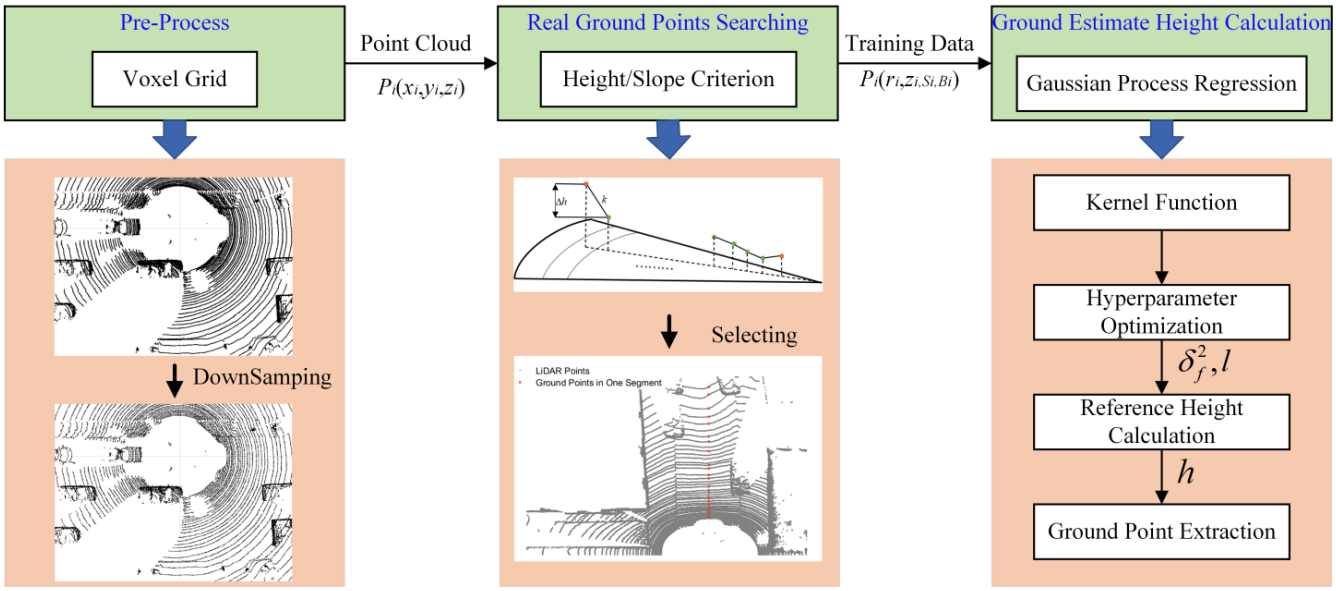


Figure 1. The flow chart of this paper.

There are two main contributions in this paper:

(a) We design height and slope criteria to search for in the initial ground points selection stage. Different from RGF, the proposed algorithm only finds enough ground points for the subsequent training of Gaussian process parameters, so it can have stricter screening conditions. Compared with the RGF algorithm, it can effectively prevent over-segmentation.

(b) We introduce a Gaussian process based on a sparse covariance function for ground reference height estimation. The introduction of sparse covariance makes the description of the ground more reasonable and computationally more efficient.

The remainder of the article is organized as follows. In Section 2, we detail our initial point searching method, and the derivation of the Gaussian process and the definition of the sparse covariance matrix are put in Section 3. Section 4 shows the segmentation result of our algorithm on the KITTI dataset and on the autonomous driving algorithm verify platform, which are compared with the RANSAC algorithm and the algorithm proposed in the literature [19]. The last section summarizes the paper, and gives the insufficiency of the paper and the future development direction.

2. Raw Data Pretreatment and Ground Candidate Points Selection

2.1. Raw Point Cloud Downsample

Usually, a velodyne 64-line LiDAR that scans a frame can obtain more than 100,000 laser points, while a 16-line LiDAR with fewer beams scans a frame to obtain more than 60,000 laser points. Therefore, the acquired raw point cloud requires downsampling, that is, a group of points in a certain space is simplified into one point by calculation, so as to reduce the computational burden of processing subsequent data. In this paper, the commonly used voxel grid downsampling algorithm is directly selected for processing the raw data before the coordinate projection of point cloud, and the result of downsampling is shown in Figure 1. It can be seen that the number of points in the original point cloud is significantly reduced after downsampling, while the height and distance information are not lost. After downsampling, all points will be projected into polar coordinated for processing.

2.2. Point Cloud Coordinate Projection

A frame of point cloud collected by LiDAR at time k is defined as $P_k = \{p_1, p_2 \dots p_n\}$, where n is the ID sequence of all points in current frame, and $p_n = (x_n, y_n, z_n)$ represents

the space coordinates of this point in a Cartesian coordinate system. The method in [19] is adopted to establish a polar coordinate system as shown in Figure 2. The segment S_i is determined by the formula:

$$S_i = \left[\frac{\arctan(x_t, y_t)}{\phi} \right] \quad (1)$$

where $i = \{1, 2, \ldots s\}$, s represents the total number of segments. The ϕ is the angle defined for different types of LiDAR. In Figure 2, bule area is one segment, each segment is divided into multiple bins, which is determined by the distance and represented as green and orange area in Figure 2. Therefore, the point $p_n = (x_n, y_n, z_n)$ in a Cartesian coordinate system can be presented as $p_n = (r_n, z_n, S_i, B_j)$, where r_n represents the distance from the LiDAR, z_n represents the height, S_i represents the ID of segment, and B_j represents the ID of bin, where $j = \{1, 2, \ldots m\}$, m represents the total number of bins in one segment. So far, each point in the Cartesian coordinate system will be projected into a certain bin in a certain segment according to the projection rules. Therefore, we only need to estimate the height of each bin in each segment and then perform height regression to complete the ground segmentation in every segment.

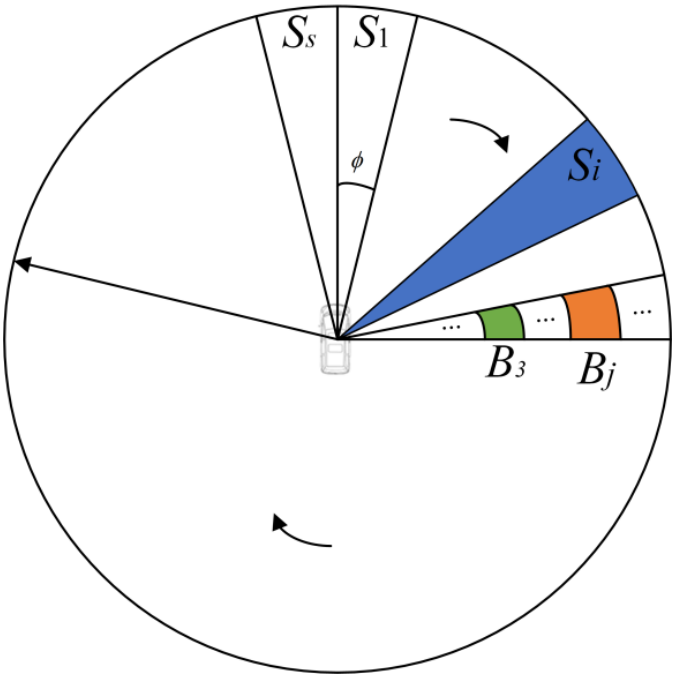


Figure 2. Schematic diagram of point cloud coordinate projection.

2.3. Ground Candidate Point Selection

The purpose of selecting ground candidate points in each segment is to obtain the ground height conference curve, obtain the ground reference height for each bin and use the reference height filter out possible ground points in each bin. In the current bin, if the lowest point is a ground point, then points within a certain distance above this point are likely to be ground points. If the lowest point is not a ground point, then there is no ground point. Therefore, this process must guarantee that all candidate points are real ground points and have a reasonable spatial distribution. In every segment, the difference between the front and rear bins is mainly in the height and slope. Therefore, reasonable height and slope criteria based on different scan model can select points that are most likely to be the ground points.

Figure 3 shows three LiDAR scan models in one segment, and each point is the lowest point of its bin. Figure 3a shows model 1, which is without obstacles in the current segment. In model 1, the height and slope different between the bins is small. Figure 3b shows

model 2, which shows the existence of obstacles with a high height. Generally, such obstacles are vehicles, trees, etc., and the ground plane after the obstacle is not scanned. Figure 3c shows model 3, which shows obstacles with a lower height. Unlike model 2, this type of obstacle is lower, and the ground is still scanned further away from the obstacle. Because the road boundary extraction is not considered in this paper, the points on the steps are still regarded as ground points.

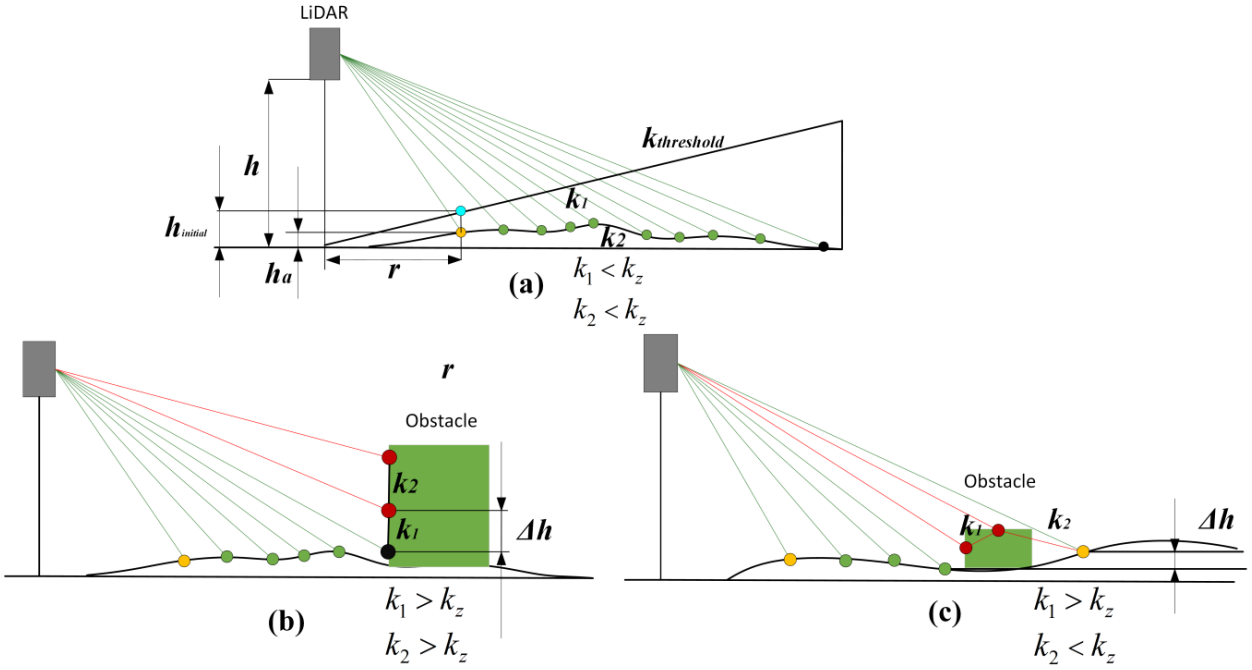


Figure 3. LiDAR scan models of one segment in ground candidate point selection process. (**a**) scan model 1. (**b**) scan model 2. (**c**) scan model 3.

The height and slope criteria are defined as the height difference Δh between the front and rear points in the current segment below the threshold h_z, and the slope below the threshold k_z. Once the criteria are not met, it is determined that an obstacle has been scanned, the details of which are shown in Figure 3b,c. On this basis, the search process of initial ground points is as follows:

1. Determine the first datum point a in current segment, which is the yellow point in Figure 3. The first datum point a needs to meet the height h_a lower than the threshold $h_{initial}$. The $h_{initial}$ is the limit ground value calculated from the LiDAR installation height h and the ground limit slope $k_{threshold}$ under urban working conditions. When the point is higher than the $h_{initial}$, it cannot be the ground point. The bin where the first datum point a located is B_a.
2. Check whether the lowest point in B_{a+1} satisfies the height criterion and slope criterion; if so, the point in B_{a+1} becomes the new datum point. Continue to check the point in the next bin. This process is then repeated until there is a point that does not satisfy the height and slope criteria.
3. When the height and slope criteria are not satisfied at bin B_{a+b}, it indicates that there are obstacles in the current bin. If the height has been trending upward until the end of the traversal, the situation is model 2; otherwise, the situation is model 3.
4. When the situation is model 2, the search is stopped, and the point in bin B_{a+b} is regarded as end point, which is the black point in Figure 3b. When the situation is model 3, set the point where the height drops as a new datum point, and continue to search until the end.

Through the above steps, the ground points can be successfully screened out. However, due to gaps in the LiDAR scan, not every bin has a laser point, and for these missing bins,

we still need to estimate its ground reference height. At the same time, due to the need to ensure that each point is a ground point, the height and slope criteria will set more stringent judgment conditions, which leads to the misjudgment of ground points in some bins as obstacle points. Therefore, a curve fitting algorithm needs to be employed to estimate the ground reference height for each bin in each segment.

3. Gaussian Process Regression Based on Sparse Covariance Function for Ground Segmentation

3.1. Gaussian Process

The Gaussian process is a learning method based on Bayesian statistical knowledge, which combines with the strong ability of the kernel method to process nonlinear information. A typical Gaussian process in function space can be defined as:

$$f(x) \sim GP(m(x), K(x, x')) \tag{2}$$

GP represents a Gaussian process, $m(x)$ and $K(x, x')$ presented the mean function and the covariance function of the Gaussian process respectively, where the more detailed definition is:

$$m(x) = E[f(x)] \tag{3}$$

$$K(x, x') = E[(f(x) - m(x))(f(x') - m(x'))] \tag{4}$$

The E represents math expectation, and define a linear regression model with standard white Gaussian noise ω as:

$$y = f(x) + \omega \tag{5}$$

where ω belongs to $N\left(0, \delta_n^2\right)$, and N represents the gaussian distribution. In fact, the essence of Gaussian process regression is to use the function $f(x)$ obtained by the Gaussian process to find the expected value of y, which can be defined as:

$$E(y) = E[f(x) + \omega] = E[f(x)] = \phi(x)^T E[w] = 0 \tag{6}$$

$$\begin{aligned} cov(y_j, y_i) &= E(y_j, y_i^T) \\ &= E[(f(x_j) + \omega_j)(f(x_i) + \omega_i)] \\ &= E[f(x_j) f(x_i)^T] + E(\omega_j \omega_i) = K(x_j, x_i) + \sigma_n^2 \end{aligned} \tag{7}$$

where the cov represented the covariance function, and the y in Formula (5) can be regarded as a Gaussian distribution, which can be expressed as:

$$y \sim N(0, K(x, x') + \sigma_n^2 I) \tag{8}$$

The I represents the identity matrix. Therefore, the training set y and testing set y' satisfy the following relationship:

$$\begin{bmatrix} y \\ y' \end{bmatrix} \sim N(0, \begin{bmatrix} K(x, x) + \sigma_n^2 I & K(x, x') \\ K(x, x') & K(x', x') \end{bmatrix}) \tag{9}$$

The $K(x, x)$ is the covariance matrix between points which belong to training data, the $K(x, x')$ is the covariance matrix between training data and testing data, and the $K(x', x')$ is the covariance matrix between points which belong to testing data. All of these covariance matrixes can be calculated through covariance function which is selected by prior information. After that, the probability edge distribution from the probability joint distribution can be presented as:

$$y' \left| x, y, x' \right. \sim N(E(y'), cov(y')) \tag{10}$$

$$E(y') = E(y' \left| x, y, x' \right.) = K(x', x)[K(x, x) + \sigma_n^2 I]^{-1} y \tag{11}$$

$$cov(y') = K(x', x') - K(x', x)[K(x, x) + \sigma_n^2 I]^{-1} K(x, x') \tag{12}$$

It can be seen from the Formulas (11) and (12) that the prediction result of Gaussian process regression is completely characterized by its mean function and covariance function. Since the mean value of the Gaussian process regression model is assumed to be 0 for reducing the computational burden, the quality of the final prediction result is related to the selection of the covariance function. However, the calculation of the covariance function is very complicated, and the kernel function can greatly simplify the calculation process.

The commonly used kernel function is the RBF, which can be mapped to infinite dimensions, has few parameters, is easy to choose, and has good anti-interference ability against noise in the data. The expression of RBF k_{SE} is:

$$k_{SE}(d) = \sigma_f^2 \exp(-\frac{d^2}{2l^2}), d = |x - x'| \tag{13}$$

The σ_f^2 in Formula (13) represents the coefficient, and the l represents the scale factor, d represents the Mahalanobis distance between training data and test data. Therefore, it uses a reasonable kernel function to curve fit the training points, and then it completes the estimation and regression of the ground height in each segment, a process called Gaussian process regression (GPR).

3.2. Gaussian Process Regression Based on Sparse Kernel Function for Ground Segmentation

However, in the actual ground segmentation process, the RBF does not perform well, it mainly because of the huge amount of computation it requires. Moreover, considering the characteristics of ground segmentation—that is, in the absence of obstacles, the height of adjacent bins on the ground will not change significantly—we introduce a sparse covariance function to reduce the amount of computation while maintaining the regression accuracy, which is presented as k_{SSE}. The expression for the sparse covariance function can be presented as:

$$k_{SSE}(d) = \begin{cases} \sigma_f^2[\frac{2+\cos(2\pi\frac{d}{l})}{3}(1-\frac{d}{l}) + \frac{1}{2\pi}\sin(2\pi)\frac{d}{l}] & \text{if } d < l \\ 0 & \text{if } d \geq l \end{cases} \tag{14}$$

The difference between introduced sparse covariance function and the RBF is that the Mahalanobis distance is added as a judgment. When the Mahalanobis distance between the samples is greater than the scale factor l, they are independent of each other, which means the kernel function is 0. When the Mahalanobis distance d is less than the scale factor l, the weight of the influence factor can be assigned by the distance between the two points. Since a certain bin is basically only affected by nearby bins, the sparse covariance function is well suited for ground segmentation in a physical significance.

Remark 1. *The pseudo-code of proposed algorithm is shown in the Algorithm 1. For ease of expression, we define several functions. The function LocationTranslation Transaltion represents the process of point cloud coordinate projection in* Section 2.2*, and S_s and B_m represent the total number of segments and bins, respectively. In this paper, S_s is set to 120, and the bin is divided according to the scanning characteristics of LiDAR. Within the scanning range of 20 m, one segment is divided into 100 bins, where the center point distance in adjacent bins is 0.2 m, and in the scanning range of 20–50 m, the remaining regions in the current segment are divided into 60 bins. Therefore, the B_m in this paper is set to 160. The judge function represents the process of ground candidate point selection, which is explained in detail in the* Section 2.3*. The points filtered by the criteria proposed in* Section 2.3 *are stored in the $P_{initial\ ground\ points}$ for Gaussian process calculation. Function model represents the reference ground curve result which calculates by the Gaussian process (GP) in* Section 3.1*. k_{SSE} represents the sparse covariance function introduced by Formula (14) and the GPR represents the ground point regression which uses curve fitting result calculated by Gaussian process in each segment, while the $H_{reference}$ represents reference regression height of the ground, and P_{Ground} represents the final ground point set.*

Algorithm 1 Ground segmentation based on Gaussian process regression with sparse covariance function

Input : $P_k, \delta_f^2, l, \phi, H_{reference}, k_{ssE}$
Output :P_{Ground}
1 : $LocationTranslation : P_k(x,y,z) \Rightarrow P_k(r,z,S,B)$
2 : **for** $i = 1 : S_s$
3 : **for** $j = 1 : B_m$
4 : $P_{initial\ ground\ points} = P_{initial\ ground\ points} \cup judge(P_j(r,z,S_i,B))$
5 : **end**
6 : $model = GP(P_{initial\ ground\ points}, \delta_f^2, l, k_{ssE})$
7 : $P_{Ground} = GPR(model, H_{reference})$
8 : **end**

4. Experiments and Analysis

KITTI dataset is the most commonly used autonomous driving evaluation dataset currently, which collects a large number of pictures, point clouds and IMU data in urban and rural environments. The KITTI dataset provides point cloud data collected by a 64-line LiDAR, and annotated with the vehicle type and its location as real ground. Therefore, we use the KITTI dataset to verify the feasibility of proposed algorithms. The code is implemented by MATLAB 2020a on a laptop with Ryzen 4800H CPU and 16 GB RAM.

In addition to the KITTI dataset, we also use our own autonomous driving experimental verification platform to verify the applicability of the algorithm in the campus environment which contains the slope and intersection conditions. Our environmental perception equipment consists of a Velodyne VLP-16 Lidar and AVT Mako G503C monocular industrial-grade camera, which are mounted on a chassis that can be controlled and all calculation work is done by an industrial computer, which equipped Intel Xeon E3-1275 v5, and NVIDIA GTX 2080s graphics card, 32 GB RAM. The unmanned driving experimental platform is shown in Figure 4.

Figure 4. Autonomous driving algorithm experimental platform.

4.1. Hyperparameter Acquisition

As shown in Formula (14), there are some quantities that need to be obtained through training in advance in the kernel function, which are called hyperparameters. Calculating hyperparameters in real-time for each frame of LiDAR data is a heavy burden on the computer. Therefore, it is more reasonable to calculate it in advance, which adapts to as many ground shapes as possible for Gaussian process. In the sparse covariance function, the hyperparameters are δ_f^2 and l. For curve fitting based on Gaussian process, the scale

factor l affects the complexity of the curve. When l is smaller, the curve is more complex. In addition, σ_f^2 affects the confidence interval, which represents the fault tolerance of the model. Therefore, in the ground reference height curve model in this paper, the physical meaning corresponding to l is the complexity of the ground curve in each segment. For a more convenient expression during next derivation process, all hyperparameters are denoted by $\theta = \{\theta_1, \theta_2, \theta_3 \ldots \theta_n\}$. The most commonly used method is to convert the maximum a posteriori solution of θ into the solution of the maximum marginal likelihood function $p(\theta|y, X)$ by Bayes theorem; the solution processes are as follows:

$$p(\theta|y, X) = \frac{p(y|X, \theta)p(\theta)}{p(y|X)} \tag{15}$$

$$\log p(\theta|y, X) = -\frac{1}{2}y^T(K + \sigma_f^2 I)^{-1}y - \frac{1}{2}\log|K + \sigma_f^2 I| - \frac{n}{2}\log 2\pi \tag{16}$$

$$\frac{\partial L(\theta)}{\partial \theta_i} = -\frac{1}{2}trace(\Sigma^{-1}\frac{\partial \Sigma}{\partial \theta_i}) - \frac{1}{2}y^T\Sigma^{-1}\frac{\partial \Sigma}{\partial \theta_i}\Sigma^{-1}y \tag{17}$$

where the $L(\theta) = \log p(\theta|y, X)$, $\sum = K(x, x) + \delta_f^2 I$. The *trace* represents the trace of the matrix. Then it uses the conjugate gradient method to solve the best value, and finds the direction of the gradient drop, and the value drops to convergence after repeating iterations. We select 500 frame point cloud data in the KITTI dataset to get the best parameters in this paper, which includes different pavements in city roads. Figure 5 is the fitting result of randomly selected six frames of data, which shows that this parameter $\left(\delta_f^2 = 0.159, l = 9.04\right)$ can better adapt to various types of ground with little fluctuation.

4.2. The Verification Effect of the Algorithm on the KITTI Dataset

In the verification experiment based on the KITTI dataset, four datasets were selected for processing, which total more than 1000 frames of point cloud data. The most used RANSAC algorithm and the Ray Ground Filter (RGF) algorithm proposed in [19] were used as comparison. In this paper, we evaluate the performance of ground segmentation using the ability resist over-segmentation and under-segmentation. Currently, there is no labeled open-source dataset which is used to evaluate the ground segmentation effect; many papers rely on visual effects to compare the effects of algorithms [15,22]. Therefore, the ground segmentation effect in this paper will be reflected by the ratio of ground points to all points in a frame of data. Figure 6 show the ratio of ground points of *2011_09_26_drive_0018* and *2011_09_26_drive_0051* respectively, and their common feature is that they all have a certain slope. It can be clearly seen that due to the existence of slopes, the ratio of ground points in the RANSAC algorithm has always been very low, which indicates that the RANSAC algorithm has always been in a state of under-segmentation. The proportion of ground points of RGF and the proposed algorithm is higher than that of the RANSAC algorithm, but the ratio of ground points of the RGF algorithm is always higher than that of the proposed algorithm, so the results need to be further analyzed.

Figure 7 shows the segmentation results for one of the frames in dataset *2011_09_26_drive_0018*, the ground points are shown in red, and the non-ground points are shown in green. It can be seen from Figure 7a that the RANSAC algorithm does not have the ability to describe the ground fluctuation, and obvious under-segmentation occurs at Section A and Section B. The more intuitive comparison between Section C and Section D in Figure 7 is shown in Figure 8, which shows the RGF has obvious over-segmentation where there are obstacles. Table 1 shows that the proportion of ground points after the frame is divided by three types of algorithms. It can also be seen that the RGF algorithm has over-segmented. The reason is the RGF algorithm must set a relatively loose slope threshold and height threshold to ensure enough acquisition, which inevitably leads to the insensitivity of the RGF algorithm to height. The proposed algorithm relies on the strong regression ability of the Gaussian process, and the slope threshold and height threshold can

be set stricter. Therefore, the proposed algorithm can both describe the undulating ground to prevent under-segment and prevent over-segmentation.

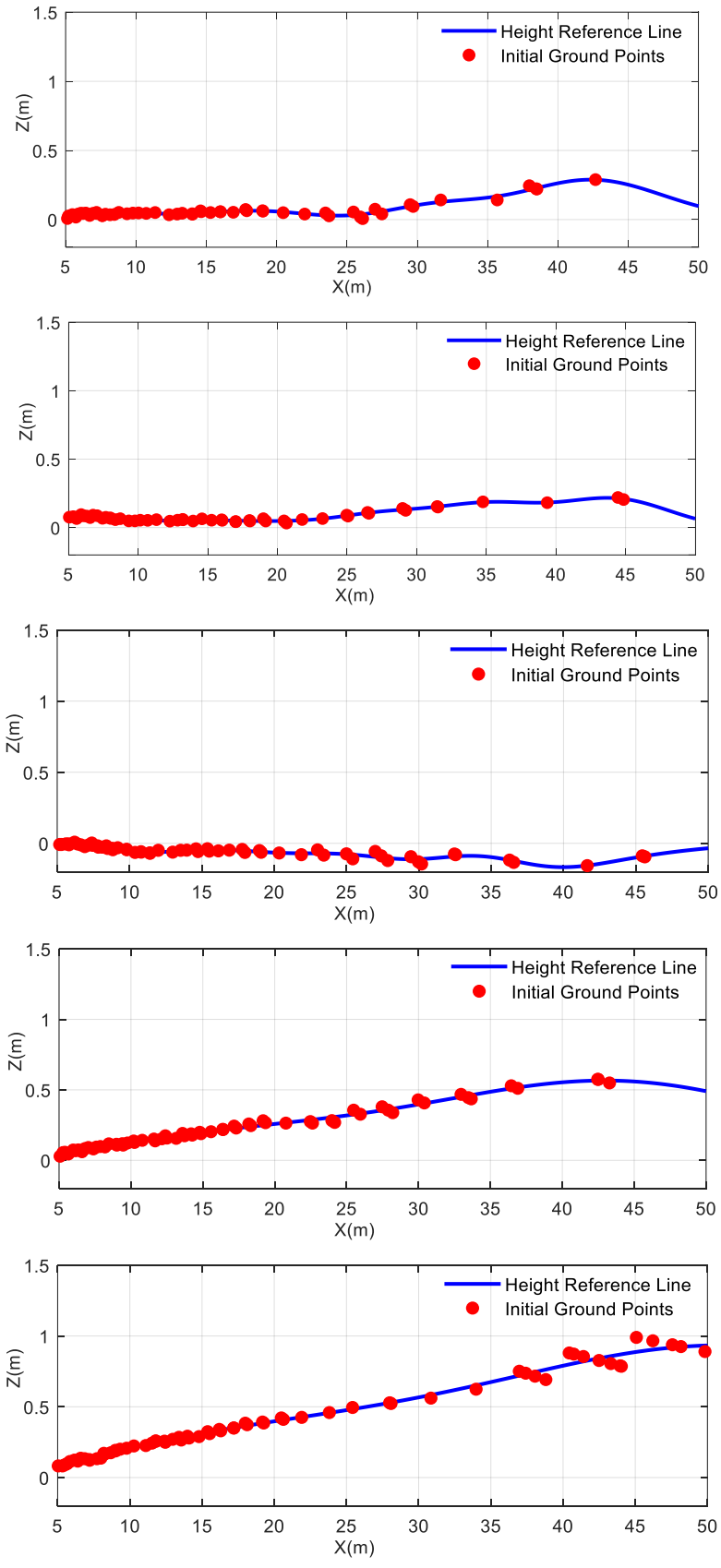


Figure 5. *Cont.*

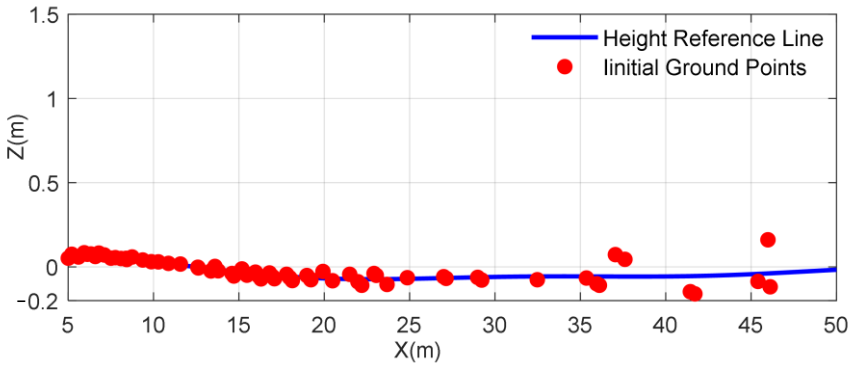


Figure 5. The results obtained by selecting six frame ground points and using the calculated parameters for Gaussian process regression.

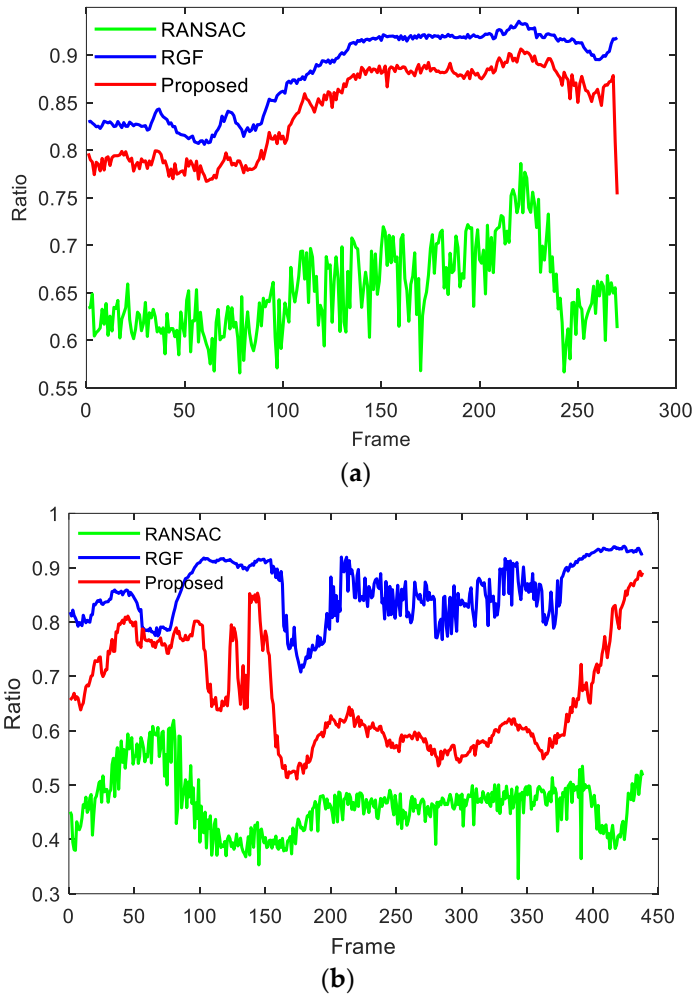
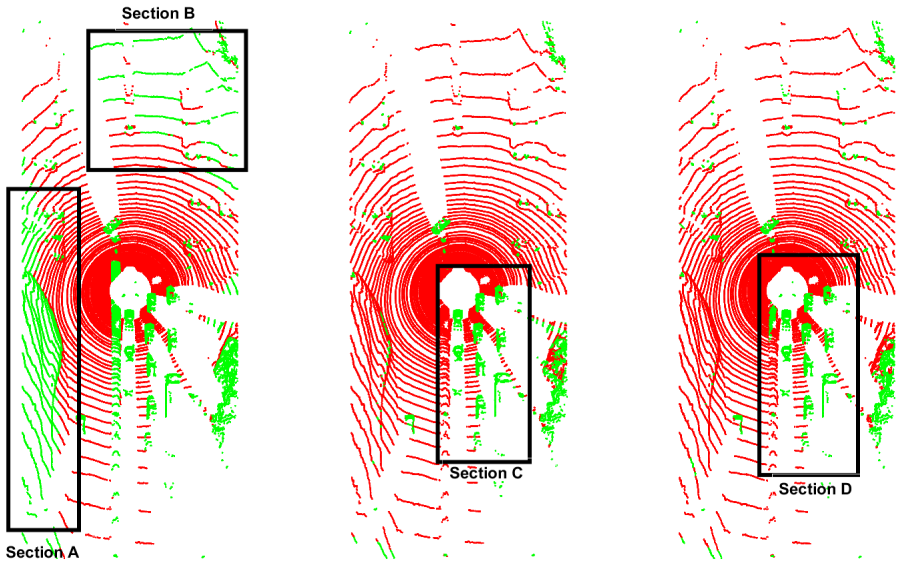


Figure 6. The ratio of ground points in two KITTI dataset. (**a**) The ratio of ground points in *2011_09_26_drive_0018*. (**b**) The ratio of ground points in *2011_09_26_drive_0051*.

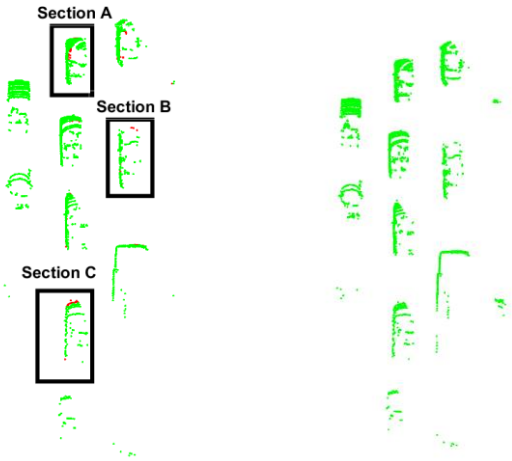
Figure 9 shows the segmentation results for one of the frames in dataset *2011_09_26_drive_0051*. It can be seen the RANSAC algorithm has obvious under-segmentation for the ground with a certain height fluctuation, which means the ground is not a perfect plane. In the real situation, the Section A, Section B and Section C in Figure 9a are with slope; it is hard to describe this kind of ground plane through RANSAC. The result of RBF representation is that it can be well adapted to slope conditions, which benefit from its slope and height threshold settings in each segment. Therefore, in the RGF algorithm, a larger height and slope threshold needs to be used for ground segmentation. This is why

significant over segmentation occurs in areas with obstacles, such as shown in Section D and Section E. From the proposed algorithm segment result, it can be clearly seen that its ground description ability is better, and the ground presents a certain undulation, which is very important to avoid over-segmentation. The ground reference height in each bin is derived from the result of Gaussian process regression.



(**a**) (**b**) (**c**)

Figure 7. Comparison of segmentation result of three types of algorithms in dataset *2011_09_26_drive_0018*. (**a**) RANSAC algorithm. (**b**) RGF algorithm. (**c**) Proposed algorithm.

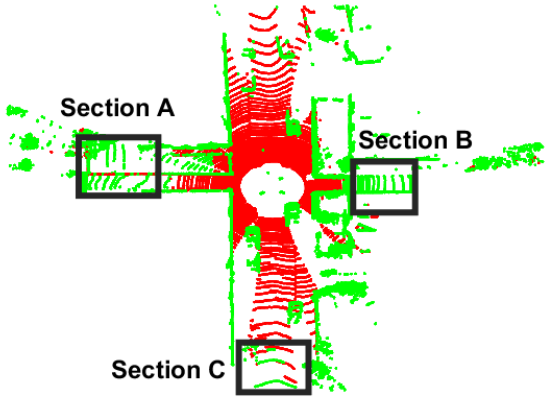


(**a**) (**b**)

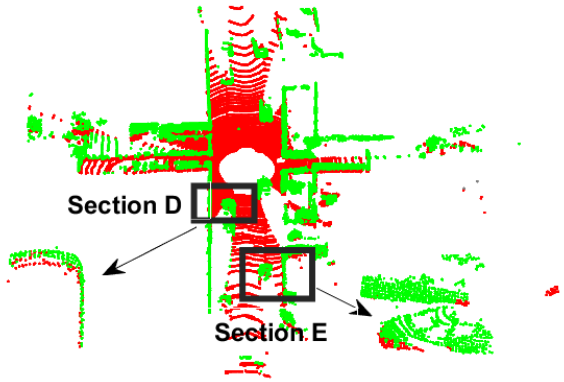
Figure 8. The enlarged comparison of Section C and Section D in Figure 7. (**a**) Section C in Figure 7. (**b**) Section D in Figure 7.

Table 1. Statistics of ground points and non-ground points in Figure 7.

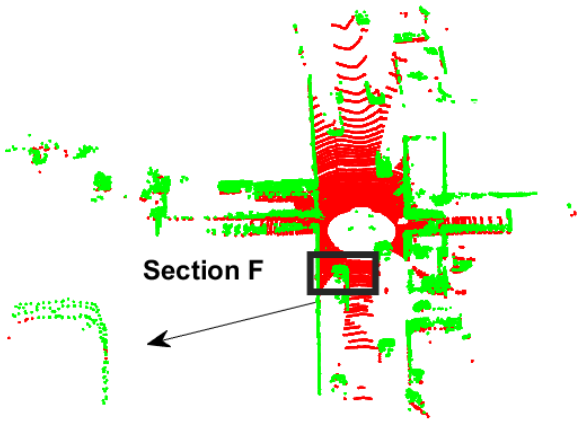
Method	Ground Points	Non-Ground Points	Ratio of Ground Points
RANSAC	23,316	13,219	63.82%
RGF	30,306	6213	82.99%
Proposed	29,396	7133	80.47%



(**a**)



(**b**)



(**c**)

Figure 9. Algorithm segmentation effect comparison in KITTI dataset *2011_09_26_drive_0051*. (**a**) RANSAC algorithm. (**b**) RGF algorithm. (**c**) Proposed algorithm.

Table 1 shows the ratio of ground points in the segmentation results of the three types of algorithms in Figure 7, and Table 2 shows the ratio of ground points in the segmentation results of the three types of algorithms in Figure 9. The number of ground points obtained by the RGF algorithm are more than the actual number of ground points, which means RGF shows obvious over-segmentation, while RANSAC shows obvious under-segmentation.

Table 2. Statistics of ground points and non-ground points in Figure 9.

Method	Ground Points	Non-Ground Points	Ratio of Ground Points
RANSAC	15,844	37,869	29.56%
GPF	20,545	31,064	39.91%
Proposed	16,416	278,561	37.08%

The time-consuming result comparison is shown in Figure 10. Since the RANSAC algorithm simply fits a random plane, it takes the shortest time, with an average time of 0.0358 s. In addition, the RGF algorithm takes into account certain ground fluctuations; the average time-consuming is 0.1511 s, which is slightly higher than the RANSAC algorithm. Compared with the RGF algorithm, the proposed algorithm has one more step, so it is reasonable that the time consumption is higher than that of the RBF algorithm. Meanwhile, the experimental conclusions of the literature [17] and literature [20] proved that the higher time consumption is acceptable and can achieve the real-time performance of autonomous driving. The average time consumption of the algorithm using the RBF function is 0.3483 s, while the average time consumption of the proposed algorithm is 0.2184 s, and the time consumption is reduced by 37.26%. This proved that the Gaussian process using sparse covariance runs more efficiently than Gaussian process using RBF kernel function, which means that compared with the traditional RBF kernel function, the introduced sparse covariance function can effectively reduce the time consumption.

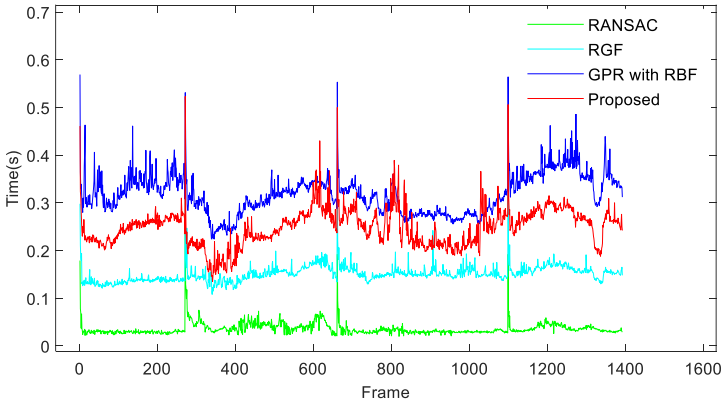


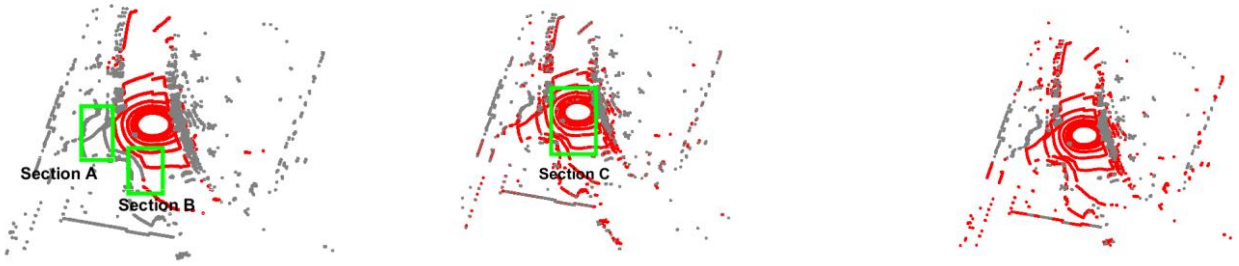
Figure 10. Time consumption comparison of four algorithms on four datasets.

4.3. The verification Effect of the Algorithm on the Autonomous Driving Algorithm Verification Platform

Similarly, we selected two typical scenes of slope and intersection for segmentation and comparison in the campus environment, which can test the ability of the ground segmentation algorithm to describe the ground. Figure 11 shows the segmentation results in the scene with slopes, and Figure 12 shows the segmentation results at the intersection.

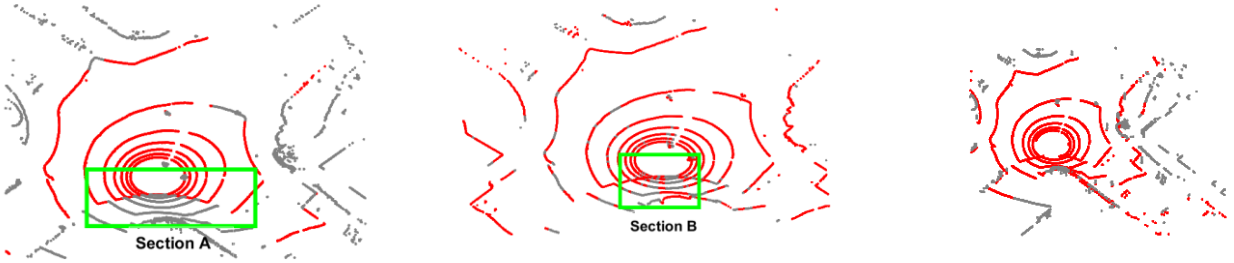
From the segmentation results of the RANSAC algorithm in Figure 11 it can be seen that because RANSAC does not have the ability to describe the slope, there is an obvious under-segmentation phenomenon in Section A and Section B. For the RGF algorithm, although the slope can be described, the segmentation failed in Section C. This is because the RGF algorithm only relies on the height and slope criteria, and the robustness is poor. Once the lowest point in a bin does not meet the criteria, there is no reference height in the bin, resulting in inaccurate estimation.

Figure 12 shows the segmentation effect in the intersection scene. Similarly, the RANSAC algorithm has obvious under-segmentation at section A, and for the RGF algorithm, Section B is also under-segmented due to the threshold setting.



(**a**) (**b**) (**c**)

Figure 11. The segmentation effects of various algorithms in the case of slope. (**a**) RANSAC algorithm. (**b**) RGF algorithm. (**c**) Proposed algorithm.



(**a**) (**b**) (**c**)

Figure 12. The segmentation effects of various algorithms in the case of intersection. (**a**) RANSAC algorithm. (**b**) RGF algorithm. (**c**) Proposed algorithm.

Table 3 shows the proportion of ground points in the segmentation results of the three types of algorithms in Figure 12. It can also be seen that RANSAC has serious under-segmentation. Comparing the RGF algorithm and the proposed algorithm, it can be found that in the case of no over-segmentation, the proposed algorithm obtains more ground points, accounting for 63.51%.

Table 3. Statistics of ground points and non-ground points in Figure 11.

Method	Ground Points	Non-Ground Points	Ratio of Ground Points
RANSAC	9605	7187	57.2%
RGF	10,168	6624	60.55%
Proposed	10,665	6127	63.51%

5. Conclusions

This paper proposes a robust ground segmentation algorithm for LiDAR based on Gaussian process to reduce situations of over-segmentation and under-segmentation. The ground reference height is accurately estimated by performing Gaussian process in each segment. Firstly, we use the height criteria and the slope criterion to search for real candidate ground points in each segment, and then use the candidate ground points as training data for Gaussian process to calculate the ground reference height of each bin. Finally, the extraction of ground points is done in each segment. Validation experiments on KITTI dataset and autonomous driving platform proved that the proposed algorithm has better resistance to over-segmentation and under-segmentation than traditional RANSAC algorithm and RGF.

However, the algorithm proposed in this paper still has some deficiencies. The first is that although a certain ground continuity is considered in the segment, the ups and downs of the ground are still affected by adjacent segments. Second, the algorithm needs to calculate the hyperparameters when the road situation has a significant change, this

means the algorithm has poor adaptability to different types of ground. In fact, we could calculate the hyperparameter during each frame process, but this would bring heavy computation burden. Therefore, future work needs to further explore the adaptive problem of hyperparameters to balance the adaptability and real-time performance of the algorithm.

Author Contributions: Conceptualization, X.J., H.Y., X.L., Z.Y., Q.W., Z.L. and Z.W.; supervision, X.J.; conception and design, X.J. and H.Y.; collection and assembly of data, X.J., H.Y. and Z.Y.; manuscript writing, X.J. and H.Y.; funding, X.J. and X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by National Science Foundation of China (51905329), Hebei Province Foreign Special Talent Introduction Plan Project (2022).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, Y.; Ma, L.; Zhong, Z.; Liu, F.; Chapman, M.A. Deep learning for lidar point clouds in autonomous driving: A review. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 3412–3432. [CrossRef] [PubMed]
2. Pendleton, S.D.; Andersen, H.; Du, X.; Shen, X. Perception, Planning, Control, and Coordination for Autonomous Vehicles. *Machines* **2017**, *5*, 6. [CrossRef]
3. Mukhtar, A.; Xia, L.; Tang, T. Vehicle detection techniques for collision avoidance systems: A review. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2318–2338. [CrossRef]
4. Ye, Y.; Fu, L.; Li, B. Object detection and tracking using multi-layer laser for autonomous urban driving. In Proceedings of the 2016 IEEE International Conference on Intelligent Transportation Systems (ITSC), Riode Janeiro, Brazil, 1–4 November 2016.
5. Wang, G.J.; Wu, J.; He, R.; Yang, S. A point cloud-based robust road curb detection and tracking method. *IEEE Access* **2019**, *7*, 24611–24625. [CrossRef]
6. Jin, X.; Wang, J.; Yan, Z.; Xu, L.; Yin, G.; Chen, N. Robust Vibration Control for Active Suspension System of In-Wheel-Motor-Driven Electric Vehicle via μ-Synthesis Methodology. *J. Dyn. Syst. Meas. Control* **2022**, *144*, 051007. [CrossRef]
7. Dai, Y.; Lee, S.G. Perception, planning and control for self-driving system based on on-board sensors. *Adv. Mech. Eng.* **2020**, *12*. [CrossRef]
8. Dieterle, T.; Particke, F.; Patino-Studencki, L.; Thielecke, J. Sensor data fusion of LIDAR with stereo RGB-D camera for object tracking. In Proceedings of the 2017 IEEE Sensors, Glasgow, UK, 29 October–1 November 2017.
9. Li, Y.; Ibanez-Guzman, J. Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems. *IEEE Signal Process. Mag.* **2020**, *37*, 50–61. [CrossRef]
10. Zhao, C.; Fu, C.; Dolan, J.M.; Wang, J. L-Shape Fitting-based Vehicle Pose Estimation and Tracking Using 3D-LiDAR. *IEEE Trans. Intell. Veh.* **2021**, *6*, 787–789. [CrossRef]
11. Sualeh, M.; Kim, G.W. Dynamic Multi-LiDAR Based Multiple Object Detection and Tracking. *Sensors* **2019**, *19*, 1474. [CrossRef] [PubMed]
12. Kim, D.; Jo, K.; Lee, M.; Sunwoo, M. L-shape model switching-based precise motion tracking of moving vehicles using laser scanners. *IEEE Trans. Intell. Transp. Syst.* **2017**, *19*, 598–612. [CrossRef]
13. Zhang, X.; Xu, W.; Dong, C.; Dolan, J.M. Efficient L-shape fitting for vehicle detection using laser scanners. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017.
14. Jin, X.; Yang, H.; Li, Z. Vehicle Detection Framework Based on LiDAR for Autonoumous Driving. In Proceedings of the 2021 5th CAA International Conference on Vehicular Control and Intelligence (CVCI), Tianjin, China, 29–31 October 2021.
15. Qiu, J.; Lai, J.; Li, Z.; Huang, K. A lidar ground segmentation algorithm for complex scenes. *Chin. J. Sci. Instrum.* **2020**, *41*, 244–251.
16. Li, L.; Yang, F.; Zhu, H.; Li, D.; Li, Y.; Tang, L. An Improved RANSAC for 3D Point Cloud Plane Segmentation Based on Normal Distribution Transformation Cells. *Remote Sens.* **2017**, *9*, 433. [CrossRef]
17. Douillard, B.; Underwood, J.; Kuntz, N.; Vlaskine, V.; Quadros, A.; Morton, P.; Frenkel, A. On the segmentation of 3D LIDAR point clouds. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011.
18. Rummelhard, L.; Paigwar, A.; Nègre, A.; Laugier, C. Ground estimation and point cloud segmentation using spatiotemporal conditional random field. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017.
19. Himmelsbach, M.; Hundelshausen, F.V.; Wuensche, H.J. Fast segmentation of 3D point clouds for ground vehicles. In Proceedings of the 2010 IEEE Intelligent Vehicles Symposium, La Jolla, CA, USA, 21–24 June 2010.

20. Chen, T.; Dai, B.; Wang, R.; Liu, D. Gaussian-Process-Based Real-Time Ground Segmentation for Autonomous Land Vehicles. *J. Intell. Robot. Syst. Theory Appl.* **2014**, *76*, 563–582. [CrossRef]
21. Chen, T.; Dai, B.; Liu, D.; Song, J. Sparse Gaussian process regression based ground segmentation for autonomous land vehicles. In Proceedings of the 27th Chinese Control and Decision Conference (2015 CCDC), Qingdao, China, 23–25 May 2015.
22. Luo, Z.; Mohrenschildt, M.; Habibi, S. A Probability Occupancy Grid Based Approach for Real-Time LiDAR Ground Segmentation. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 998–1010. [CrossRef]
23. Narksri, P.; Takeuchi, E.; Ninomiya, Y.; Morales, Y.; Akai, N.; Kawaguchi, N. A slope-robust cascaded ground segmentation in 3D point cloud for autonomous vehicles. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018.
24. Jiménez, V.; Godoy, J.; Artuñedo, A.; Villagra, J. Ground Segmentation Algorithm for Sloped Terrain and Sparse LiDAR Point Cloud. *IEEE Access* **2021**, *9*, 132914–132927. [CrossRef]
25. Shen, Z.; Liang, H.; Lin, L.; Wang, Z.; Huang, W.; Yu, J. Fast Ground Segmentation for 3D LiDAR Point Cloud Based on Jump-Convolution-Process. *Remote Sens.* **2021**, *13*, 3239. [CrossRef]
26. Chu, P.M.; Cho, S.; Park, J.; Fong, S.; Cho, K. Enhanced ground segmentation method for Lidar point clouds in human-centric autonomous robot systems. *Hum.-Cent. Comput. Inf. Sci.* **2019**, *9*, 17. [CrossRef]
27. Zermas, D.; Izzat, I.; Papanikolopoulos, N. Fast segmentation of 3D point clouds: A paradigm on LiDAR data for autonomous vehicle applications. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017.
28. Zhang, Y.; Wang, J.; Wang, X.; Dolan, J.M. Road-segmentation-based curb detection method for self-driving via a 3D-LiDAR sensor. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 3981–3991. [CrossRef]
29. Huang, W.; Liang, H.; Lin, L.; Wang, Z.; Wang, S.; Yu, B.; Niu, R. A Fast Point Cloud Ground Segmentation Approach Based on Coarse-To-Fine Markov Random Field. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–14. [CrossRef]
30. Melkumyan, A.; Ramos, F. A sparse covariance function for exact Gaussian process inference in large datasets. In Proceedings of the IJCAI International Joint Conference on Artificial Intelligence, Pasadena, CA, USA, 11–17 July 2009.