Noname manuscript No.
(will be inserted by the editor)

Gaussian-process-based Real-time Ground Segmentation for Autonomous Land Vehicles

Tongtong Chen · Bin Dai · Ruili Wang · Daxue Liu

Received: date / Accepted: date

Abstract Ground segmentation is a key component for Autonomous Land Vehicle (ALV) navigation in an outdoor environment. This paper presents a novel algorithm for real-time segmenting three-dimensional scans of various terrains. An individual terrain scan is represented as a circular polar grid map that is divided into a number of segments. A one-dimensional Gaussian Process (GP) regression with a non-stationary covariance function is used to distinguish the ground points or obstacles in each segment. The proposed approach splits a large-scale ground segmentation problem into many simple GP regression problems with lower complexity, and can then get a real-time performance while yielding acceptable ground segmentation results. In order to verify the effectiveness of our approach, experiments have been carried out both on a public dataset and the data collected by our own ALV in different outdoor scenes. Our approach has been compared with two previous ground segmentation techniques. The results show that our approach can get a better trade-off between computational time and accuracy. Thus, it can lead to successive object classification and local path planning in real time. Our approach has been successfully applied to our ALV, which won the championship in the 2011 Chinese Future Challenge in the city of Ordos.

Keywords Autonomous Land Vehicle · Ground Segmentation · Gaussian Process · 3D Point Cloud

T. Chen · B. Dai · D. Liu
College of Mechatronic Engineering and Automation,
National University of Defense Technology, Hunan, P.R. China
E-mail: chentongtong5208@gmail.com

B. Dai
E-mail: bindai.cs@gmail.com

D. Liu
E-mail: daxue_l@yahoo.com.cn

R. Wang
School of Engineering and Advanced Technology, Massey University,
Auckland, New Zealand
E-mail: r.wang@massey.ac.nz

Mathematics Subject Classification (2000) MSC 68T45 · MSC 68T40 · MSC 60G15

1 Introduction

Navigation in an unknown outdoor environment is a challenging task for an Autonomous Land Vehicle (ALV). It is one interesting topic in the areas related to sensors, measurements, and control. The research outcomes of this topic have wide applications. For example, an ALV can be used in military, where tasks are too risky for soldiers. Since ground segmentation is a key component for ALV navigation in an outdoor environment, it has been studied recently in the robotic research area. In order to arrive at a predefined destination safely, a robot needs identifying a traversable ground and various obstacles in the area. Then, the robot can plan a right path toward its destination while avoiding collisions. As ground segmentation is a critical pre-processing step in obstacle classification, the performance of ground segmentation will directly affect the successive obstacle classification that especially is important to local path planning and position estimation in dynamic environments. A typical dynamic urban scene is shown in Fig. 1.

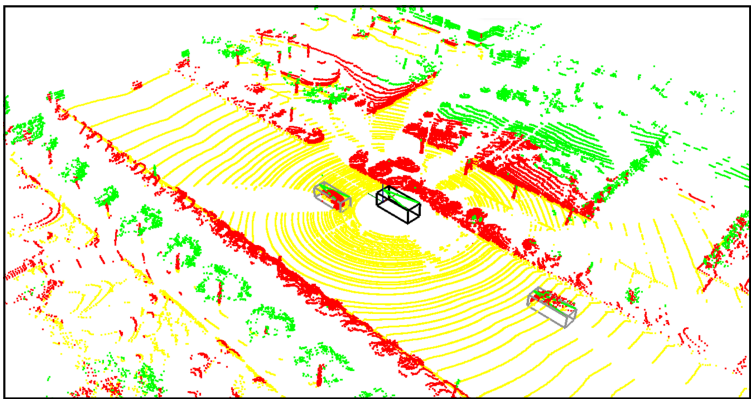


Fig. 1 A typical segmentation result of one scan collected by Velodyne LIDAR (Light Detection And Ranging) in an urban environment. The yellow, red, green pixels represent the ground, obstacles, and overhanging structures, respectively. The middle cube is our ALV, and the other two cubes are the vehicles detected based on ground segmentation. The green arrow represents the driving direction

With the cameras and range scanning devices becoming standard equipment in an ALV, much research has been carried out over the last few years concerning ground segmentation. As color cameras are mass-produced and comparatively inexpensive, many research groups use them as the main sensors for their robots or AVLs. The methods can be divided into two groups: the methods based on the

monocular vision [7, 9, 28, 32, 33] and the stereo vision [8, 14, 20]. These methods usually use texture features and color features to segment the ground. Thus, they are highly sensitive to the illumination and shadows.

Compared to cameras, laser rangefinders contain mobile parts in their design, which makes them complex and expensive. However, they can provide accurate range information, which makes them popular and widely used in the domain of robots.

Douillard et al. [4] presented a method based on the fast Fourier transform for segmenting three-dimensional scans of underwater unstructured terrains in the frequency domain. The lower frequency components represent the underlying ground. The bandwidth is automatically determined using peak detection. To the best of our knowledge, this is the only work that segments three-dimensional scans in the frequency domain.

In the spatial domain, many ground segmentation approaches have been proposed, which can roughly be divided into three subgroups: grid cell-based methods [6, 29], line-based methods [10, 13, 26] and surface-based methods [5, 11, 17, 23, 31].

Thrun et al. [29] presented a ground segmentation method based on the Min-Max elevation map, which was widely used by the teams at the 2007 DARPA (Defense Advanced Research Projects Agency) Urban Challenge competition. The height difference is computed between the maximum and minimum heights of the returns falling in the cell. The cells whose height differences are less than a predefined threshold are identified as ground. This approach originally could not solve the under-segmentation problem, which was later addressed in reference [6]. The approach in [6] extracts the ground surface with a mean elevation map. The largest gradient between the cell and its neighbors is retained as the cell's gradient. If the gradient is less than a threshold, the cell belongs to the ground. The largest set of connected ground cells becomes the ground surface. The grid cell-based methods only use the local information rather than the global ground continuity information to segment the ground. Thus, the performance is so sensitive that it can be easily affected by observation noises and the extrinsic calibration of a sensor.

Stamos et al. [26] presented an online ground segmentation algorithm based on the coarse classification method in reference [10]. It first detects a sequence of horizontal points that can be potentially on the ground within each scanline, and then verifies them as the actual ground points across the scanlines. The algorithm can generate a robust estimation of the ground points, along with the height of the ground at each scanline. Himmelsbach et al. [13] described a fast method for segmentation of large-size long range 3D point clouds for a high-speed ALV. The approach divides the polar grid map into many segments, and uses the line extraction algorithms in [19] to represent the ground surface in each segment. The method splits the segmentation problem into many simple fitting problems with lower complexity. It can get a real-time performance, but it is not suitable for a rough terrain. This issue will be addressed in this paper.

The surface-based method performs ground segmentation using LIDAR (LIght Detection And Ranging) data based on the notion of ground continuity. The commonly used approaches are based on the 3D grid maps. Reference [17] presented an extension of Gaussian process regression technique for terrain modelling. To deal with the preservation of the structural feature, the approach employs a non-stationary covariance function and presents a new method to local kernel adaption based on Elevation Structure Tensor (EST). Based on reference [17], Plagemann

et al. [23] modelled a terrain using a GP regression with a non-stationary covariance function. Also, in order to further increase the efficiency of the model, a non-iterative model is derived. Another GP prior is put into the local kernel's parameters to calculate the local kernel at any point. In [5], a novel iterative approach, Gaussian Process Incremental Sample Consensus (GP-INSAC), was presented to estimate the probabilistic, continuous ground surface for spare 3D data that are cluttered by the non-ground objects. The algorithm maintains the approximate capability of GP terrain modelling and endows it with the capability of outlier rejection. Thus, it can represent all kinds of terrains. However, it only processes the data off-line. The approach in [11] uses online kernel-based learning in Reproducing Kernel Hilbert Space (RKHS) to estimate a continuous ground surface over the area of interest, which also provides the surface's upper and lower bounds. The visibility information is exploited to increase accuracy and the computational time is decreased by gradient-based optimization. These methods can usually get good ground segmentation results, but cannot be applied in real-time applications, which are essential for an ALV.

In this paper, we present a real-time ground segmentation approach based on Gaussian process regression in the polar grid map for an ALV equipped with Velodyne HDL-64E Scanning LIDAR system [3]. Our method constructs a circular polar grid map, and uses a one-dimensional Gaussian process regression with the non-stationary covariance function proposed by Paciorek and Schervish in [21] to allow for local adaptation to the undulation of the ground in each segment of the map. The adaptation is achieved by calculating the local gradient in each segment. Our method is inspired by references [13] and [5]. Comparing with reference [13], this paper exploits continuous Gaussian process regression instead of the piecewise line fitting in every segment, which makes our approach more robust to rough terrains. Relative to reference [5] that modelled the global ground in the three-dimensional grid map using Gaussian Process Incremental Sample Consensus with stationary covariance function, our method divides a large-scale two-dimensional ground segmentation problem in the three-dimensional grid map into many one-dimensional Gaussian process regression problems with lower complexity in the segments of the polar grid map, which will increase the computational efficiency greatly. Also, a non-stationary covariance function is adopted to increase the accuracy of segmentation.

The main contributions of this paper are two-fold. The most significant one is that a complicated, large-scale two-dimensional ground segmentation problem can be divided into many simple one-dimensional Gaussian process regression problems in the segments of the polar grid map. Thus, the LIDAR data can be processed online. The second contribution is the locally adaptive Gaussian process regression, in which the length-scale l_i is adapted using the 1D gradient information in the segment of the polar grid map. Also, we formulate the pseudo log marginal likelihood to learn the hyper-parameters of the Gaussian process regression with non-stationary covariance function using several typical terrain structures. With the learnt hyper-parameters, our algorithm can adapt to most urban and countryside environments.

The rest of the paper is structured as follows. In the next section, the three-dimensional Velodyne LIDAR data acquisition and how to construct the polar grid map are described. Section 3 describes the ground segmentation approach based on Gaussian process regression with non-stationary covariance function in detail.

Some experimental results are showed in Section 4. Our algorithm are compared with two previous ground segmentation techniques on the public dataset in [16] and the data collected by our ALV in different outdoor scenes. The results show a promising performance of our algorithm. Finally, conclusions and an outline of the future work are given in Section 5.

2 Data Acquisition and Polar Grid Map Representation

Velodyne HDL-64E S2 is an improved high definition LIDAR scanner designed for ALV navigation, mapping, surveying and other uses. With its full 360 degree horizontal Field Of View (FOV) by 26.8 degree vertical FOV, the HDL-64E S2 provides significantly more environmental information up to 120 meters than previously available, and with over 1.3 million points per second output at the frame rate of 10HZ. The HDL-64E S2 outputs UDP Ethernet packets, each of which contains a data payload of 1206 bytes that represent distance, azimuth, and intensity information respectively [15]. Unlike the references [10, 26], in this paper, the points that are collected within one scan (required 0.1 second) are treated as if they are collected at the same time. The approach in [18] is adopted to transform the original points to the LIDAR coordinate system. Through simple translation, these points can be transformed into the vehicle's local coordinate system, in which the results of ground segmentation can be used for local path planning directly. We assume that the vehicle's local coordinate system is tied to its centre with y-axis pointing directly forward, and z-axis pointing upward. Formally, a scan collected at time point t is represented by the set $P_t = \{p_1, p_2, \ldots, p_l\}$, where $p_i = (x_i, y_i, z_i)^T$ is given by its Euclidean coordinates with respect to the vehicle's local coordinate system.

The scan contains so many points that we need to represent them in a right way to increase the computational efficiency. Instead of establishing the complex relationships based on a kernel function among the grid cells [5, 11, 17, 23, 31] or calculating the height differences [29] in the elevation map, this paper follows reference [13] to construct a polar grid map. There are at least three merits in our algorithm: (1) the polar grid map retains all the data to avoid the under-segmentation problem; (2) the polar grid map fits well to the physical model of LIDAR. The data can be partitioned in a way that allows the ground points to be segmented by many simple one-dimensional local regressions, which will greatly improve the computational efficiency; (3) the polar grid map is suitable to describe both flat and sloped terrains as well as the transition between both [13]. In order to keep the completeness and continuity of the paper, we will describe how to construct the polar grid map for our ALV briefly.

Given a scan $P_t = \{p_1, p_2, \ldots, p_l\}$ collected at time point t with respect to the vehicle's local coordinate system with origin O, we firstly partition these 3D points into a circular polar grid map with radius R_{def}, which will be divided into M segments, as shown in Fig. 2.

The point whose horizontal distance to origin O exceeds R_{def} will be omitted. The residual points are mapped into segments according to their angles with the positive y-axis of the coordinate system. Point p_i will be assigned into segment $S(p_i)$ according to Eq. 1:

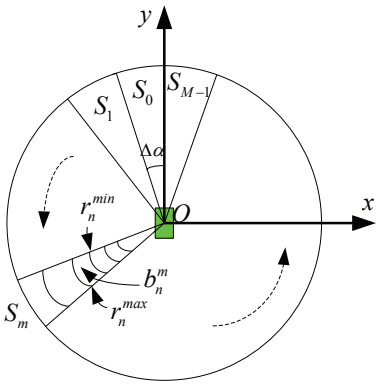


Fig. 2 A polar grid map, which is evenly split into M segments counter-clockwise. Every segment is divided into N bins unevenly. Obviously, the range of a bin that is close to the LIDAR centre is smaller, while the range of a bin that is far away from the centre is relatively large. The middle green rectangle represents our ALV

$$S(p_i) = \big[\frac{Ang(x_i, y_i)}{\Delta\alpha}\big], \tag{1}$$

where [.] is the rounding function, $Ang(x_i, y_i)$ represents the horizontal angle between the positive y-axis and Op_i, $\Delta\alpha$ represents the angle that every segment covers. All points that are mapped into the same segment m can form a set P_m [13]:

$$P_m = \{p_i \in P_t | S(p_i) = m\}. \tag{2}$$

Every segment will be divided into N bins to discrete the range components of the points. The n^{th} bin b_n^m in segment m covers the range from r_n^{min} to r_n^{max}. Point $p_i = (x_i, y_i, z_i)^T$ is mapped into bin b_n^m if and only if its radial coordinate r_i satisfies the inequality in Eq. 3:

$$r_n^{min} < r_i = \sqrt{x_i^2 + y_i^2} \leq r_n^{max} \wedge p_i \in P_m. \tag{3}$$

Since data from the LIDAR samples the ground unevenly, quickly becoming sparse at longer ranges and containing large voids because of occlusions, the range that every bin covers varies in this paper, according to the bin's distance to the LIDAR. All the 3D points that are mapped into the n^{th} bin of segment m can form a set $P_{b_n^m}$. For every set $P_{b_n^m}$, there is a one-to-one set $P'_{b_n^m}$ of 2D points [13]:

$$P'_{b_n^m} = \left\{p'_i = \left(r_i, z_i\right)^T | p_i \in P_{b_n^m}\right\}. \tag{4}$$

Our goal is to split a complex two-dimensional ground segmentation problem into many simple one-dimensional regressions. Thus, we choose the 2D point $p'_i \in P'_{b_n^m}$ with the lowest z_i in every bin of segment m to form the candidate ground point set PG_m:

$$PG_m = \left\{p'_i | p'_i \in P'_{b_n^m}, z_i = \min(H_n^m), n = 1 \ldots N\right\}, \tag{5}$$

where H_n^m is the set of z coordinates of all the points that are mapped into bin b_n^m. Up to now, all the 3D points have been mapped into $P'_{b_n^m}$. For every segment m, the set PG_m can be used to model the ground. After the reference height of the ground in every segment is obtained, it can be used to give each point in the corresponding segment a label: ground or obstacle, according to its relative height with the reference height of the ground.

3 Ground Segmentation as a 1D Regression Problem

In robotics, Velodyne HDL-64E S2 is a popular sensor, as it provides accurate and high frequency 3D measurements. After the pre-processing step above, these 3D raw points are transformed into a polar grid map that will be divided into M segments. For every segment of the map, in this section, we will construct a continuous 1D ground model that yields predictive distributions of ground heights at arbitrary locations of the segment. This can split the complex two-dimensional ground modelling problem into many one-dimensional local regression problems to increase the computational efficiency.

For segment m, the process of constructing the local ground model can be described as follows. Given a two-dimensional point set $PG_m = \left\{(r_i, z_i)\right\}_{i=1}^n$ of n location samples, $r_i, z_i \in \mathbb{R}$ can be contaminated by obstacle points. Our task is to get rid of the outliers and use the residual samples set D_m to build the ground model $p_m(z_*|D_m, r_*)$ for segment m, where r_* is the test input (radial coordinate) in segment m, and z_* is the predictive height of the ground at input r_*. Since Gaussian process regression has powerful approximate ability, it has been widely used in robotics recently [1, 5, 12, 17, 23, 30, 31]. In this paper, a one-dimensional Gaussian process regression will be used to build the ground model in every segment of the polar grid map, rather than to build a continuous two-dimensional ground model of the whole 3D elevation map directly.

3.1 Gaussian Process Regression

A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution [24]. It is a non-parametric continuous representation that provides a powerful basis for modelling spatially correlated and possibly uncertain data. Given a 2D training set $D = \left\{(r_i, z_i)\right\}_{i=1}^n$ of n samples, according to the definition of the Gaussian process, the joint distribution of these training samples can be expressed as:

$$p(Z|R) \sim N(\mu, K), \tag{6}$$

where $R = [r_1, \ldots, r_n]^T, Z = [z_1, \ldots, z_n]^T$, μ is the mean vector, and K is the covariance matrix. For notational simplicity, the mean vector μ is often taken to be zero [24]. The covariance matrix K models the relationships between the random variables. It is specified in terms of a covariance function k and the noise variance σ_n^2:

$$K(r_i, r_j) = k(r_i, r_j) + \sigma_n^2 \delta_{ij}, \tag{7}$$

where δ_{ij} is a Kronecker delta, which is one iff $i = j$ and zero otherwise.

The commonly used stationary, isotropic covariance function k is squared-exponential (SE) covariance function. It has the following form [24]:

$$k(r_i, r_j) = \sigma_f^2 \exp\left(-\frac{(r_i - r_j)^2}{2l^2}\right), \tag{8}$$

where l is the length-scale, and σ_f^2 is the signal covariance. These free parameters constitute the hyper-parameters of the Gaussian process. Gaussian process regression uses the fact that the joint distribution of the training outputs Z and the output z_* at the test input r_*, according to the definition of Gaussian process, is:

$$\begin{bmatrix} Z \\ z_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(R,R) & K(R,r_*) \\ K(r_*,R) & K(r_*,r_*) \end{bmatrix}\right). \tag{9}$$

Thus, the key predictive equation for Gaussian process regression takes the following form:

$$\begin{aligned} \bar{z_*} &= K(r_*, R)K^{-1}Z \\ V[z_*] &= K(r_*, r_*) - K(r_*, R)K^{-1}K(R, r_*) \end{aligned}, \tag{10}$$

where $\bar{z_*}$ and $V[z_*]$ are the mean and covariance of the output z_* respectively, K^{-1} is the matrix inversion of K, and $K = K(R) = \big(K(r_i, r_j)\big)_{1 \le i,j \le n}$. The k^{th} element of $K(r_*, R) \in \mathbb{R}^{1 \times n}$ is $K(r_*, r_k)$, which can be calculated using Eq. 7.

The main drawback of squared-exponential covariance function is the assumption that the length-scale is constant over the whole input space [23]. In order to adapt to various terrains, we should choose a long length-scale for the flat ground and a short length-scale for the rough ground.

In this paper, in order to increase the accuracy of the segmentation, the non-stationary, isotropic covariance function proposed by Paciorek and Schervish [21] is used to solve the regression problem for each segment. For one-dimensional case, it takes the following form:

$$k(r_i, r_j) = \sigma_f^2 \left(l_i^2\right)^{\frac{1}{4}} \left(l_j^2\right)^{\frac{1}{4}} \left(\frac{l_i^2 + l_j^2}{2}\right)^{-\frac{1}{2}} \exp\left(-\frac{2(r_i - r_j)^2}{l_i^2 + l_j^2}\right), \tag{11}$$

where each input location r_i has an individual length-scale l_i. The covariance between two outputs z_i and z_j is calculated by averaging between the two length-scales at the corresponding input locations r_i and r_j [22]. In this way, the covariance function of the model can describe the local characteristics of the ground better, to some extent.

In this paper, the length-scales are adapted with the 1D gradient information in the segment of the polar grid map. For every segment m, the incremental algorithm in [19] is used to extract the lines L_m, and the line whose absolute gradient is greater than 10 degrees is omitted. That is because the ground whose absolute gradient is greater than 10 degrees is considered to be an obstacle for an ALV in this paper. For each input r_i in PG_m, we find the line in L_m that is closest to it in terms of the minimum distance to one of the line's endpoints, and then assign

input r_i the absolute value of the gradient $g(r_i)$ of the corresponding line. The local length-scale l_i for input r_i is calculated as follows:

$$l_i = \begin{cases} a * \lg(\frac{1}{|g(r_i)|}) & \text{if } |g(r_i)| > g_{def} \\ a * \lg(\frac{1}{|g_{def}|}) & \text{otherwise} \end{cases}, \tag{12}$$

where a is just a scale parameter that is greater than 0, and can be learnt in Section 3.3; $|g(r_i)|$ is bounded by g_{def} to prevent l_i to be infinite in the flat ground, g_{def} is a predefined parameter that is greater than 0.

3.2 Ground Segmentation based on GP Regression

For segment m, a candidate 2D ground point set PG_m can be obtained, which can be contaminated by obstacle points. However, the typical Gaussian process regression assumes that all data of set PG_m are ground points with few outliers. In this paper, the ground segmentation problem can be re-formulated as obtaining one regression model with the ability of outlier rejection for each segment. In this paper, iterative learning is adopted to build our local ground model that has both the powerful approximate ability and outlier rejection ability in every segment of the polar grid map. The detailed algorithm is shown in Algorithm 1.

Algorithm 1 Ground Segmentation

Input: $P_t = \{p_1, p_2, \ldots, p_l\}, M, N, B, T_s, T_r, a, \sigma_f, \sigma_n, t_{model}, t_{data}$
Output: label of $p_i, i = 1, \ldots, l$

1: $\left(PG, P'_{b_n^m}\right)$=PolarGridMap$(P_t, M, N)$;
2: **for** $i = 1 : M - 1$ **do**
3: L_i=fitline(PG_i, a);
4: $s_{new} = s_p = \emptyset$;
5: s_{new}=seed(PG_i, B, T_s);
6: **while** size$(s_{new}) > 0$ **do**
7: $s_p = s_p \cup s_{new}$;
8: $model$=GPR$(s_p, L_i, \sigma_f, \sigma_n)$;
9: $test$=$PG_i - s_p$;
10: s_{new}=eval$(model, test, t_{data}, t_{model})$;
11: **end while**
12: **for** $j = 0 : N - 1$ **do**
13: segment$\left(model, P'_{b_i^j}, T_r\right)$;
14: **end for**
15: **end for**

The algorithm starts with a scan of 3D point clouds P_t, and outputs the label of every point: ground or obstacle. The algorithm mainly contains six steps: polar grid map representation, line fitting, seed estimation, Gaussian process regression with non-stationary covariance function, new seed evaluation and point-wise segmentation.

The polar grid map representation is corresponding to the function $polargridMap$ in line 1. The polar grid map is partitioned into M segments, each of which is divided into N bins. The detailed process is shown in Section 2.

Line fitting is corresponding to the function $fitline$ in line 3. All the line extraction algorithms in [19] can be used. We use the incremental algorithm in this paper, because the points in set PG_i have been sorted according to their distances to the LIDAR. Each point in set PG_i is assigned a local length-scale calculated by Eq. 12 (See Fig. 3).

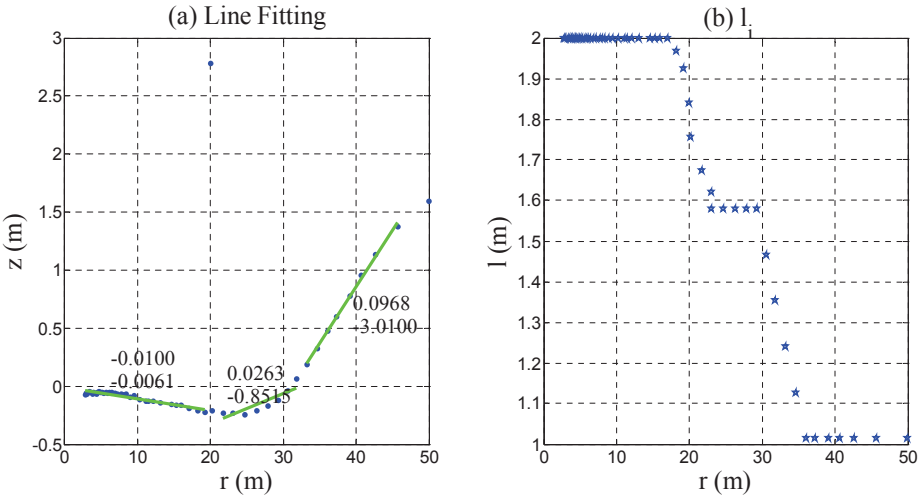


Fig. 3 The results of the line extraction in one segment. The green lines represent the ground in the segment, and the numerals are the gradients and interceptions of the corresponding lines (a). The pentacles are the length-scales at the input locations, which are processed by an average filter ($a = 1$) (b)

The third step is seed estimation in line 5. Its goal is to find the initial ground points as the training data. In this paper, the points whose absolute heights are less than T_s within a fixed radius B of the sensor are chosen as the initial seeds. Since the absolute heights of the points are used as a criterion to choose the seeds in this function, the accuracy of extrinsic calibration of the LIADR needs to meet some requirements. If there are no initial seeds, our algorithm will be of no avail. Generally, $B = 30m$ and $T_s = 30cm$ are chosen because of the climbing obstacle capability of our ALV.

Obtaining the ground model based on Gaussian process regression (line 8) is the goal of the fourth step. Equations 7 and 11 are used to calculate the covariance matrix between the pairs of seeds s_p. If the horizontal distance between two seeds is less than one meter, only one seed will be kept and the other one will be deleted. Thus, there will not be too many seeds in each segment. The small size of the covariance function increases the computational efficiency greatly.

The new seed evaluation is corresponding to the function $eval$ in line 10. Equations 7 and 11 can be used to calculate $K(r_*, r_*)$ and $K(r_*, s_p)$ for every test point (r_*, z_*). Equation 10 can then be used to calculate the mean and variance of the output z_* at the test input location r_*. In this paper, the rulers (in Eq. 13) proposed by Douillard [5] are adopted to evaluate the attribute of the input points:

the new seeds or outliers. The parameter t_{model} is the threshold of the variance $V[z]$ of the output z_*, while t_{data} specifies the normalized distance between the real output z_* and the expected mean $\bar{z}$ of the output at the test input r_*.

$$\begin{aligned} V[z] &\leq t_{model} \\ \frac{|z_* - \bar{z}|}{\sqrt{\sigma_n^2 + V[z]}} &\leq t_{data} \end{aligned} \quad . \tag{13}$$

For the test point (r_*, z_*), it is classified as the new seed if and only if it satisfies both of the inequalities in Eq. 13. Otherwise, it is an outlier. Starting from the initial seeds (line 5), the seeds are accumulated per iteration until no more new seeds are found (lines 6 - 11). Figure 4 shows the results of the iterations in the first segment of the scene shown in Fig. 10.

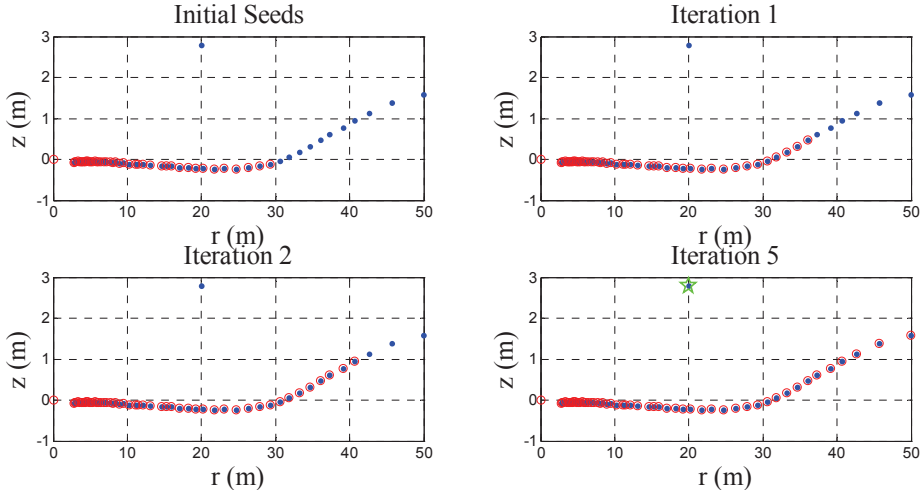


Fig. 4 Cross section of a segment in a polar grid map. The blue points are the likely ground points, while the red circles represent the seeds after per iteration. The point (the pentacle in the bottom-right figure) that is the return of a wire across the road is classified as the outlier

The final step is the point-wise segmentation, which is corresponding to the function *segment* in line 13. For segment i, the final seeds, which are found through the iterations above, can be used to model the local ground. For the j^{th} bin b_j^i in segment i, $(r_j^{min} + r_j^{max})/2$ is used as the radial coordinate of bin b_j^i. Thus, for $r_j^i = (r_j^{min} + r_j^{max})/2$, Eq. 10 can be used to predict the mean height of the ground $\bar{z}_j^i$ in bin b_j^i. This mean ground height $\bar{z}_j^i$ of bin b_j^i is defined as the reference ground height H_{ij}. For every point p'_k in the set $P'_{b_j^i} = \{p'_k = (r_k, z_k) | p_k \in P_{b_j^i}\}$, if its relative height $|z_k - H_{ij}|$ is less than T_r, p'_k is classified as a ground point. Otherwise, it is an obstacle point. Since the point-to-point mapping between set $P_{b_j^i}$ and set $P'_{b_j^i}$ is one-to-one, the 3D point p_k and the 2D point p'_k have the same label. Now, all the labels of the 3D points in the scan can be obtained.

3.3 Learning Hyper-parameters

So far, we have described our method assuming that we had known the hyper-parameters $\theta = \{\sigma_f, a, \sigma_n\}$. However, in practice, they cannot be obtained in advance. In this section, we formulate the regression task in every segment as a supervised learning problem, and focus on how to learn these parameters.

Given a training set $\Im = \{r^m, z^m\}_{m=1}^M$, each segment is a training example of the form (r^m, z^m), where each $r^m = \{r_1^m, \ldots, r_{n_m}^m\}$ is a sequence of n_m radial coordinates in segment m, each $z^m = \{z_1^m, \ldots, z_{n_m}^m\}$ is a corresponding sequence of the heights of the ground, and n_m is the number of points in segment m. In this paper, we assume that these examples are conditionally independent, and try to seek these hyper-parameters by maximizing the pseudo log marginal likelihood [24]:

$$\sum_{m=1}^{M} \log p(z^m|r^m, \theta) = -\frac{1}{2}\sum_{m=1}^{M}(z^m)^T K_m^{-1} z^m - \frac{1}{2}\log\Big(\prod_{m=1}^{M}|K_m|\Big) - \frac{\log 2\pi}{2}\sum_{m=1}^{M} n_m, \tag{14}$$

where K_m is the covariance matrix for the noisy z^m values in segment m, and can be calculated using Eq. 7.

To set the hyper-parameters by maximizing Eq. 14, we seek the partial derivatives of the pseudo log marginal likelihood with respect to the hyper-parameters. We obtain:

$$\frac{\partial}{\partial\theta_i}\Big(\sum_{m=1}^{M}\log p(z^m|r^m,\theta)\Big) = \frac{1}{2}\sum_{m=1}^{M} tr\left(\left(\alpha_m\alpha_m^T - K_m^{-1}\right)\frac{\partial K_m}{\partial\theta_i}\right), \tag{15}$$

where $tr()$ represents the trace of the matrix, $\alpha_m = K_m^{-1}z_m$, and $\frac{\partial K_m}{\partial\theta_i}$ is a matrix of elementwise derivatives. The specific formulas of $\frac{\partial K_m}{\partial\theta_i}$ are as follows:

$$\begin{aligned}
\frac{\partial K_m(r_i,r_j)}{\partial\sigma_f} &= 2\sigma_f\left(d_i^2\right)^{\frac{1}{4}}\left(d_j^2\right)^{\frac{1}{4}}\left(\frac{d_i^2+d_j^2}{2}\right)^{-\frac{1}{2}}\exp\left(-\frac{2(r_i-r_j)^2}{a^2(d_i^2+d_j^2)}\right)\\
\frac{\partial K_m(r_i,r_j)}{\partial a} &= \sigma_f^2\left(d_i^2\right)^{\frac{1}{4}}\left(d_j^2\right)^{\frac{1}{4}}\left(\frac{d_i^2+d_j^2}{2}\right)^{-\frac{1}{2}}\exp\left(-\frac{2(r_i-r_j)^2}{a^2(d_i^2+d_j^2)}\right)\left(\frac{4(r_i-r_j)^2}{a^3(d_i^2+d_j^2)}\right),\\
\frac{\partial K_m(r_i,r_j)}{\partial\sigma_n} &= 2\sigma_n\delta_{ij}
\end{aligned} \tag{16}$$

where $d_i = \lg(\frac{1}{|g(r_i)|})$. A gradient based optimizer can be adopted to solve the problem.

In this paper, ten labelled scans of street scenes collected in an urban area of Boston [16] and some manually labelled scans are used to learn the hyper-parameters. Every scan is divided into 180 segments. Some typical segments such as the uphill roads, downhill roads, flat roads, rough roads and so on are chosen to form the training set $\Im$. Figure 5 shows 9 typical segments in the training set $\Im$.

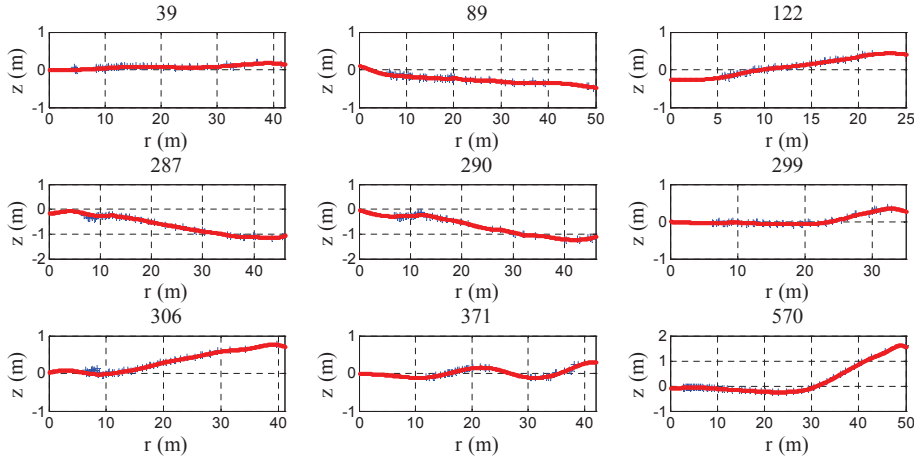


Fig. 5 9 typical segments chosen from the training set. The blue pluses are training points in the segments. The red lines represent the terrains in the segments with the learnt parameters ($\sigma_f^2 = 0.0528, \sigma_n^2 = 0.0012, a = 6.2978$)

3.4 Segmentation of Obstacle Points Based on Ground

Having labelled all points of a scan, both the obstacles such as vehicles, pedestrians, stumps on the road and the overhanging structures such as tree canopy and wires are classified as obstacles that an ALV cannot travel across. This is a typical under-segmentation problem. However, with the method in this paper, it can be easily solved. Since local path planning of our ALV only depends on a 2.5D grid map, we map the 3D obstacle points in the polar grid map into the grid map with a resolution of 0.2 meter (the reason we use 0.2 meter is that the width of the tyres of our ALV is about 0.2 meter). Symbol P_{ij} is used to represent the set of the obstacle points that are mapped into the grid cell c_{ij}.

$$P_{ij} = \left\{ p_k | i \leq \frac{x_k}{0.2} < i+1 \cap j \leq \frac{y_k}{0.2} < j+1 \cap p_k \in obs \right\}, \tag{17}$$

where x_k, y_k are the x coordinate and y coordinate of point p_k in the vehicle's local coordinate system, correspondingly, and the label of the point is obstacle. For all points in P_{ij}, we sort them from low to high according to their relative heights with the ground. For any two adjacent points p_m, p_n in P_{ij}, z_m, z_n are their relative heights with the ground, respectively. If z_m is greater than z_n, and the distance $|z_m - z_n|$ is above a given threshold T, $|z_m - z_n| > T$, then the points whose relative heights are equal or larger than z_m in P_{ij} are overhanging points. The other points are real obstacle points. The threshold T is chosen in terms of the height of our ALV, $T = 1.8m$.

4 Experimental Evaluation and Analysis

The goal of the experimental evaluation is to demonstrate the usefulness of the ground segmentation algorithm proposed in Section 3.2. In order to evaluate the

real-time performance and the accuracy of the algorithm proposed in this paper, two other algorithms proposed in [5, 13] are adopted to be compared with. We implement them in C++ as well as the point cloud library (PCL) [25], and conduct several contrastive experiments both in urban and countryside environments. Our experimental platform is a modified Toyota Land Cruiser, as shown in Fig. 6, equipped with a Velodyne HDL-64E S2, Three SICK LMS 291, and a NovAtel SPAN-CPT GPS-aided inertial navigation system (INS). The Velodyne laser scanner rotates with a speed of 10 HZ. Thus, every scan contains up to 0.13 million points. At the same time, in order to evaluate their accuracy intuitively, a quantitative performance evaluation is conducted on the dataset provided in [16]. In all the experiments, we use a notebook computer with an Intel P8600 CPU with 2.4 GHz using double cores and 2 GB main memory.

Fig. 6 Our ALV used for the experiments in various environments. It is equipped with some cameras, lasers, and positioning sensor for perception

Our algorithm and the algorithm in [13] are both based on the polar grid map. In the process of constructing the polar grid map of diameter 100 meters, we set the segment size $M = 180$. Every segment is divided into 160 bins, $N = 160$. To simplify computation, the range that the bins that are close to the LIDAR within 20 meters cover is about 0.2 meter along radial direction and the others cover about 0.5 meter along the radial direction. The algorithm in [5] is based on the three-dimensional grid map, while in this paper, the grid map covers an area of 100 by 100 meters with the horizontal resolution of 0.4 meter and the vertical resolution of 0.2 meter.[1] The content of every grid cell is the 3D mean of the points in it. In order to increase the computational efficiency, the data compression is performed that the data are decimated by keeping 1 in every 30 of the grid means.

[1] Seemingly, the grid map is larger than the polar grid map. However, the data from LIDAR samples the ground unevenly, quickly becoming sparse at longer ranges. Thus, the number of points in a grid map is only a litter larger than that in a polar grid map. According to our statistics, the number of points in the grid map is only 0.52 percent more than that in the polar grid map.

4.1 Runtime

As the algorithm is applied to the ALV, the real-time performance of the ground segmentation algorithm is our focus. We have evaluated the proposed algorithm online for numerous scans acquired while autonomously driving in the city of Ordos. The route is in a complex suburban environment that contains some uphill roads, downhill roads, flat roads and so on, as shown in Fig. 7. The algorithms in [5, 13] are also implemented off-line using the same scans. Figure 8 shows the results of these algorithms.

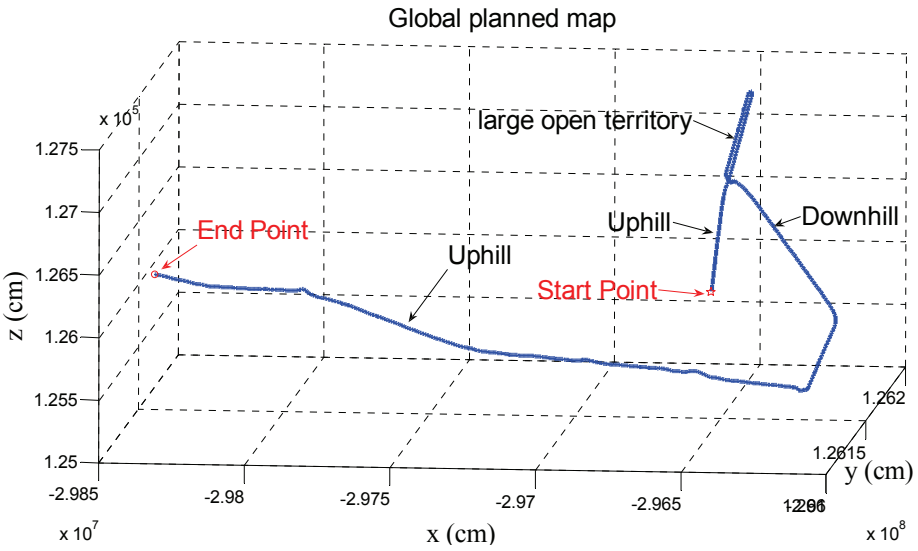


Fig. 7 The global planned map in the city of Ordos acquired from the position sensor (INS)

Most of the time, with an average of 74.93 ms per scan, our algorithm performs in real time. But, in Interval A, the runtime of every scan is larger than 100 ms. That is because the ALV locates in a large open territory (as shown in Fig. 7) and the ground on the right of ALV is uphill. In these corresponding segments of the polar grid map, the iterative times (lines 6 - 11 in Algorithm 1) increase, so does the time. The same phenomenon happens to the algorithm in [5]. From Fig. 8, one can find that our algorithm is slower than the algorithm in [13] with an average of 35.12 ms per scan, while it is also much faster than the algorithm in [5] with an average of 678.77 ms per scan. Even though, our algorithm can get a real-time performance, because the refresh rate of the scan is 100 ms per scan.

4.2 Accuracy

The dataset provided in [16] contains ten labelled scans of street scenes collected in an urban area in Boston. Each point has a label such as ground, car, house and so

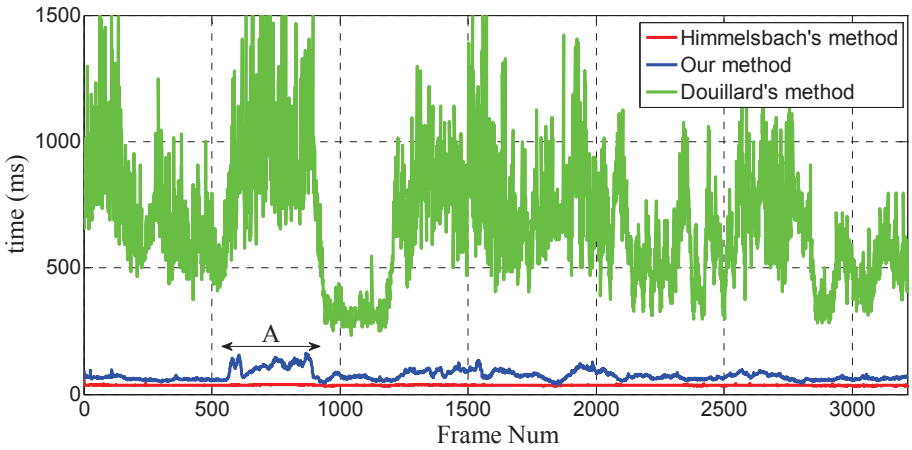


Fig. 8 Runtime of our algorithm (blue) compared to that of [13] (red) and [5] (green) for 3200 scans acquired in the city of Ordos

on. We only focus on two kinds of laser points: ground points and obstacle points in this paper. All the non-ground points (the laser points that belong to car, house, person, tree, street sign, fence and other) are gathered into the obstacle class. All ten scans of the laser points are mapped into the polar grid map with diameter of 100 meters and the three-dimensional grid map that covers an area of 100 by 100 meters, respectively. The results of ground segmentation are detailed in Table 1. All the algorithms can get good performances. That is because these scans are collected in the urban streets where the terrains are very flat (some scans where the terrains are rough will be shown in the next subsection). Our algorithm can achieve an accuracy of 0.9767, which is 1.16 percents higher than the algorithm in [13], but 0.17 percent lower than the one in [5]. As known, it can be a disaster for an ALV to misclassify the non-ground points as ground, because this will easily result in a collision with the obstacle. From Table 1, one can see that our algorithm misclassifies the least obstacle points as ground. Thus, our algorithm is the safest one among the three algorithms, to some extent.

Table 1 Confusion matrix of the three algorithms

	Our algorithm		Algorithm in [5]		Algorithm in [13]	
Inferred / True	ground	non-ground	ground	non-ground	ground	non-ground
ground	192251	8268	196060	2570	189780	10728
non-ground	6822	439919	11479	440589	11844	434901
accuracy	0.9767		0.9784		0.9651	

4.3 Segmentation in Various Environments

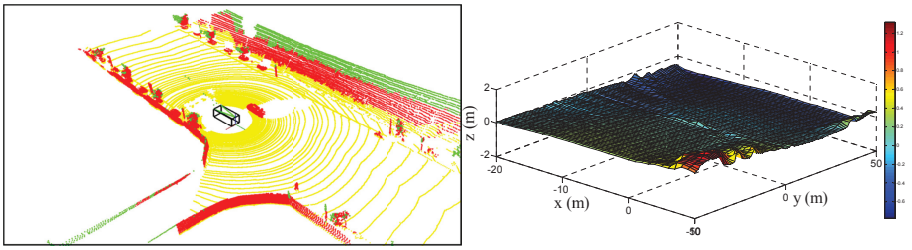
In order to evaluate the usefulness of the proposed algorithm to our ALV, these algorithms are compared by many scans acquired from some urban and countryside environments. Example results are shown in Figs. 9, 10, 11, and 12, which show both the ground segmentation results and the ground surfaces obtained using these algorithms. These scenes include a highway with some traffic, a field road with some road-blocks and two campus roads with a steep slope and a T-junction, respectively. In our implementation, the parameters in our algorithm are set to $\sigma_n^2 = 0.0012, \sigma_f^2 = 0.0528, a = 6.2978, T_{model} = 0.04, T_{data} = 3, T_r = 0.3$. These parameters are learnt by maximizing the pseudo log marginal likelihood in Section 3.3.

Figures 9, 10, 11, and 12 show some results of the proposed ground segmentation algorithm in four different scenes compared to those obtained with the algorithms in [13] and [5], respectively. Every figure contains two columns: the left column shows the segmentation results of our algorithm, the algorithm in [5], and the algorithm in [13], while the other column shows the ground surfaces obtained with the corresponding algorithms. In the left column of the figures, the yellow, red, green points represent the ground, obstacles and overhanging structures, respectively. Since the algorithms in this paper and [13] are based on the polar grid maps that have been partitioned into many segments, we can only build a local ground model in every segment. The global ground surfaces obtained with our algorithm and the algorithm in [13] are simply made up by (the combination of) all the local ground models in the segments.[2]

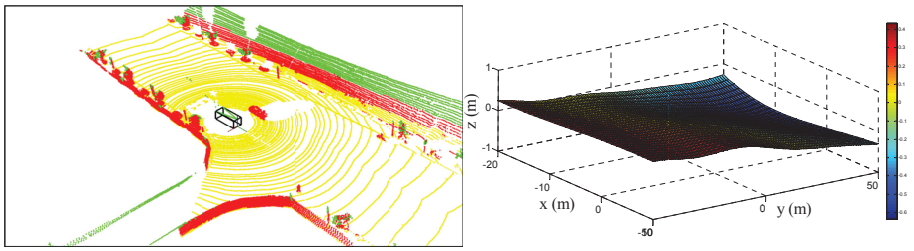
Figure 9 shows a highway scene with a vehicle, a cyclist and a pedestrian. The left column shows the segmentation results of the algorithms, while the right one shows the ground surfaces in the scene obtained using the corresponding algorithms. Our algorithm (as shown in Fig. 9a) and the algorithm in [5] (as shown in Fig. 9b) can get good performances, while some ground points (the red points in the blue quadrangle) are misclassified as obstacles by the algorithm in [13] (as shown in Fig. 9c). That is because the vehicle occludes with some ground points, which violates smooth transitions between pairs of successive lines (In [13], the ground is represented by lines in every segment). The ground surface obtained using the algorithm in [13] is shown in the right image of Fig. 9c. The flat ground surface in the red quadrangle is misclassified as a bump, which is higher than the ground surface around. Thus, the real ground points in the red quadrangle are misclassified as negative obstacles.

Figure 10 shows a campus scene with an 8.7% slope. Our algorithm and the algorithm in [5] classify the slope accurately, as shown in the left column, while the algorithm in [13] cannot (the red and green points in the blue quadrangle of the left image of Fig. 10c). That is because the root mean squared error of the line fitting for such a steep slope exceeds a predefined threshold. The algorithm in this paper can accumulate the points on the slope as the seeds through the iterations

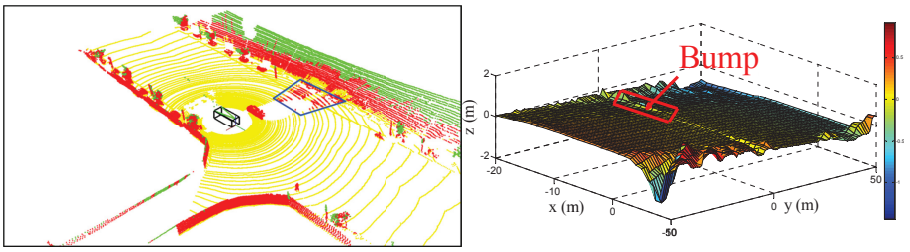
[2] The detailed procedure is as follows: A look-up table between the grid cells in the grid map and the bins in the polar grid map is established. The reference ground height in every bin can be calculated with the local ground model in the corresponding segment. Through the look-up table, the reference ground height in every grid cell in the grid map can be obtained. All the reference ground heights in the grid cells make up the global ground surface. Note that there is not a specific formula about the global ground model in this paper.



(a) The segmentation result (in the left image) and the ground surface (in the right image) obtained with our algorithm



(b) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [5]

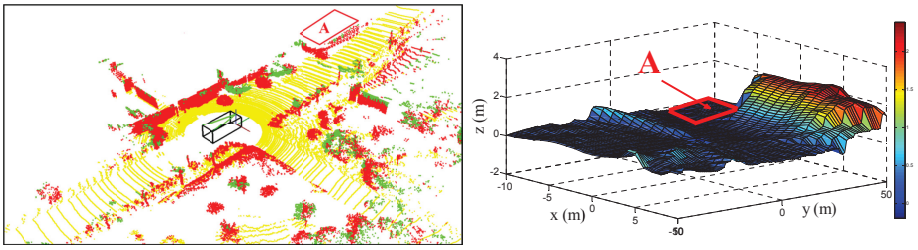


(c) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [13]

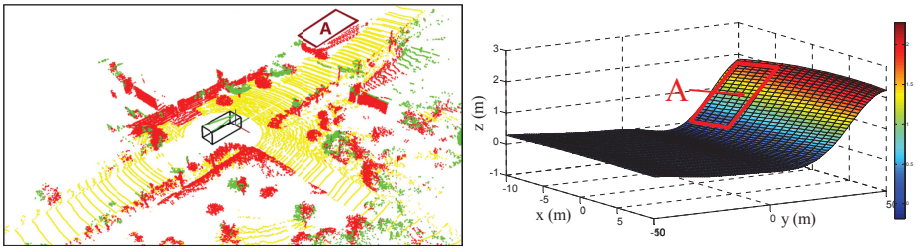
Fig. 9 Our segmentation result in (a) of a highway scene, with a vehicle, a cyclist and a pedestrian, compared with the results of the algorithms in [5] in (b), and [13] in (c). In the left column of the figure, the ground points, obstacles, and overhanging structures are colored yellow, red, and green pixels, respectively. The right column shows the ground surfaces obtained with the corresponding algorithms. Figures 10, 11, 12 and 13 are the same

as shown in Fig. 4. Thus, the local ground in every segment can be modelled accurately. The similar iterative processes are implemented in the algorithm in [5] too.

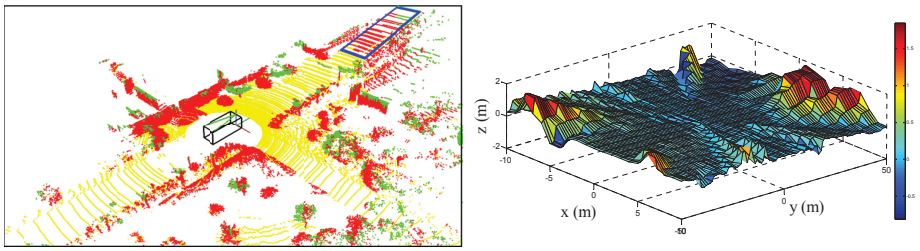
Nevertheless, there are also some differences between our algorithm and the algorithm in [5]. Comparing the ground surfaces in Figs. 10a and 10b, one can find that the ground surface in Region A is classified as a flat ground in Fig. 10a while Region A is classified as a slope in Fig. 10b. The main reason is the difference of the distribution of the ground seeds. There are no laser returns in the corresponding



(a) The segmentation result (in the left image) and the ground surface (in the right image) obtained with our algorithm



(b) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [5]

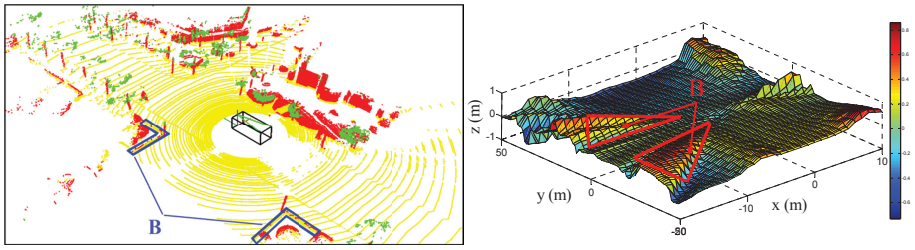


(c) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [13]

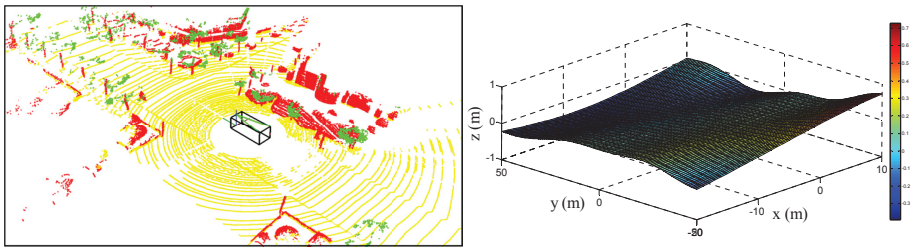
Fig. 10 Our segmentation result in (a) of a campus scene with an 8.7% slope, compared with the results of the algorithms in [5] in (b), and [13] in (c)

Region A (Quad. A) in the left image of Fig. 10b. Our algorithm divides Region A into many segments and gets the local ground models in the segments. Only the seeds, which are on the flat ground in these segments, are employed. When the global ground model is calculated using the algorithm in [5], the seeds both on the flat ground and the slope (the seeds on the slope are found through the iterative processes) can be employed to predict the ground surface in Region A. The mean prediction (in Eq. 10) of the height of the ground surface is a linear combination of the heights of the seeds. Thus, our algorithm classifies Region A as a flat ground, while the algorithm in [5] classifies it to be a slope.

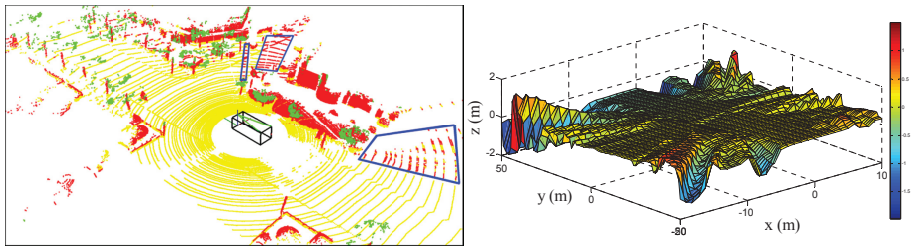
Figure 11 shows another campus scene with a T-junction. Detecting a long range intersection is very important to local path planning. The algorithms in this paper and [5] can get good performances (in Figs. 11a and 11b). That is



(a) The segmentation result (in the left image) and the ground surface (in the right image) obtained with our algorithm



(b) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [5]

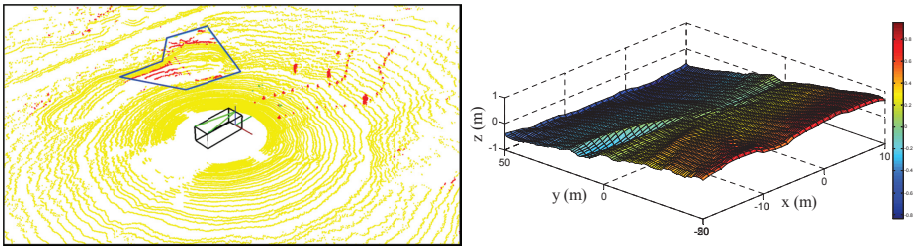


(c) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [13]

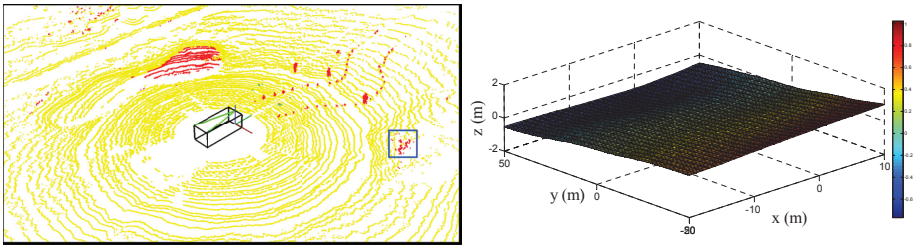
Fig. 11 Our segmentation result in (a) of a campus scene with a T-junction in front of the ALV, compared with the results of the algorithms in [5] in (b), and [13] in (c)

because these algorithms have seeds on the ground of the intersection. However, the algorithm in [13] recognizes the intersection as the obstacles as shown in the blue quadrangles of the left image of Fig. 11c, which makes the successive intersection detection impossible [2]. Compared with the result of the algorithm in [5], our segmentation result still has some defects. The stair steps are classified as ground in the blue polygons B in the left image of Fig. 11a.

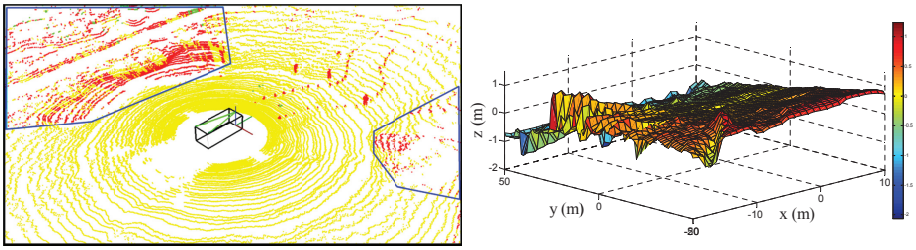
Our algorithm is insensitive to the slowly rising obstacles such as stair steps. Though the height of each stair step is often less than 30 centimeters, it is an obstacle for the ALV in an urban environment. In the segments, which contain the obstacles like the stair steps, the minimum heights in the bins increase gradually. Since there are only a few seeds on the flat ground in each segment, our algorithm



(a) The segmentation result (in the left image) and the ground surface (in the right image) obtained with our algorithm



(b) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [5]



(c) The segmentation result (in the left image) and the ground surface (in the right image) obtained with the algorithm in [13]

Fig. 12 Our segmentation result in (a) of the countryside scene with some road-blocks, compared with the results of the algorithms in [5] in (b), and [13] in (c)

may accumulate the points on the stair steps as the seeds through the iterative processes. The stair steps will be misclassified as ground surface. Thus, the height of the ground surface in the segment will increase (e.g., Region B shown in the right image of Fig. 11a). The algorithm in [5] can get a better result in this situation. That is because it uses many seeds on the global flat ground surface. The points on the stair steps can hardly become the seeds through the iterative processes in the algorithm in [5]. The predicted ground surface, which is shown in the right image of Fig. 11b, is very flat, and the stair steps are classified as obstacles accurately.

The countryside scene is shown in Fig. 12. There are many road-blocks in front of our ALV which are put into type S on the flat ground to form the curbs of the road. All the three algorithms can separate the road-blocks from the ground, because they all have the abilities to describe a flat road. However, as for a rough

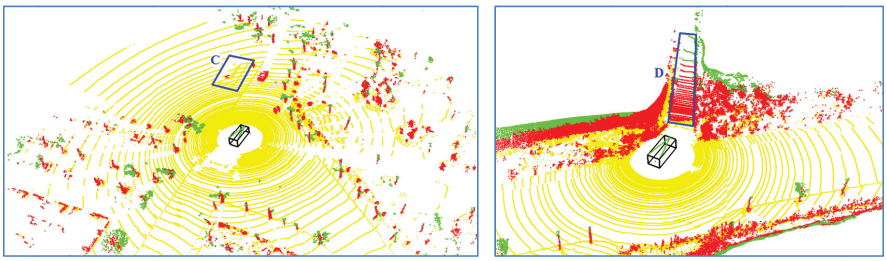


Fig. 13 Examples where the performances of our algorithm are poor. The left image shows a campus scene with a car, some persons, some trees and two stair steps (the bottom stair step is 0.15 meter high and the other is 0.2 meter) that are in Quad. C, our algorithm misclassifies the stair steps as ground, but segments the car, persons, trees and overhanging structures accurately. In the right image, our ALV locates at the intersection of a trunk road and a countryside uphill road. The uphill road is misclassified as obstacles in Quad. D

road, they have different performances. The algorithm in [13] cannot adapt to a rough road, many ground points are classified as obstacles (i.e. the red points in the blue polygon in the left image of Fig. 12c). That is because the rough ground surface makes the root mean square error of the line fitting in the segments large. As there are many seeds on the flat ground, the algorithm in [5] can get a flat ground surface. It can detect the negative obstacles accurately, while misclassify the slowly rising hillside as obstacles (as shown in the blue quadrangle in the left image of Fig. 12b). Because our algorithm is insensitive to the hillsides, it can classify them as ground accurately.

From the right columns of Figs. 9, 10, 11, and 12, one can see that the ground surface obtained by the algorithm in [5] is the smoothest one, the second is our algorithm, and the last is the algorithm in [13]. In [5], the algorithm that is based on two-dimensional Gaussian process regression uses the seeds in the whole map to model the global ground. Thus, it keeps the continuity of the ground in the whole map. Our method models the local ground in every segment. It only uses the seeds in each segment. However, less seeds make our algorithm insensitive to the stair steps and hillsides, while less seeds also make our algorithm get a real-time performance. Our algorithm can only keep the continuity in each segment, not between the two adjacent segments. The algorithm in [13] uses the line fitting method to model the local ground in the segments. Only the relative continuity in each segment can be kept. At the same time, the occlusion and rough terrain will greatly influence the results of the line fitting.

Figure 13 presents some examples where the performances of our algorithm are poor to demonstrate in which environment situations our algorithm is ineffectiveness. In the left image of Fig. 13, our algorithm misclassifies the stair steps as ground (Region C in the left image), because our algorithm divides the polar grid map into many segments. Each segment contains only a few seeds, which makes our algorithm insensitive to the slowly rising obstacles, such as the stair steps. Our algorithm will accumulate the points on the stair steps as seeds to model the local ground surface, and misclassify the stair steps as ground. The other drawback of our algorithm is shown in the right image of Fig. 13. Our ALV locates at the inter-

section of a trunk road and a countryside uphill road (a 12% slope), and faces the uphill road. As previously mentioned, we only choose the points whose absolute heights are less than 0.3 meter within a fixed radius 30 meters of the sensor as the initial seeds. However, in this scene, the slope is so steep that there are no initial seeds at all, which makes our algorithm of no avail.

Though our algorithm can be improved further, it had been successfully used in our ALV, which won the champion in the 2011 Chinese Future Challenge that was hold in the city of Ordos, China, where the ALV needed to finish the tasks such as avoiding the static obstacles, merging into vehicles, U-turns and so on. With the algorithm in this paper and path planning algorithm in [27], our ALV finished the route of ten kilometers autonomously in the urban environment of Ordos in twenty minutes, without any human intervention.

5 Conclusion and Future Work

This paper presents a novel real-time ground segmentation algorithm for an ALV, which combines the advantages of the both algorithms in [5] and [13]. The algorithm splits the complex two-dimensional ground segmentation problem into many one-dimensional regression problems to increase the computational efficiency, and keeps the continuity of the ground in every segment. It can get a better accuracy compared with the algorithm in [13] and the real-time performance compared with the algorithm in [5]. Our algorithm is useful to the successive object classification and local path planning in real time.

In the future, we will investigate how to keep the continuity between the adjacent segments based on the algorithm in this paper. Also, we will bring in the direction of the normal at each point when choosing the initial seeds, because the normal of the ground is nearly upward in an urban environment. If the normal of a point is not upward, it will not be chosen as an initial seed. This approach may increase the accuracy and robustness of our algorithm, to some extent.

Acknowledgment

The work is support in part by National Nature Science Foundation of China under Grant No.61075043. Thanks all persons in laboratory who had joined the competition and worked together to won the champion in the third Chinese Future Challenge.

References

1. Abuhashim, T., Sukkarieh, S.: Incorporating geometric information into gaussian process terrain models from monocular images. In: Internation Conference on Intelligent Robots and Systems (IROS), pp. 4162–4168 (2012)
2. Chen, T., Dai, B., Liu, D., Liu, Z.: Lidar-based long range road intersection detection. In: Sixth International Conference on Image and Graphics, pp. 754–759 (2011)
3. Chen, T., Dai, B., Liu, D., Zhang, B., Liu, Q.: 3d lidar-based ground segmentation. In: Asian Conference on Pattern Recognition, pp. 446–450 (2011)
4. Douillard, B., Nourani-Vatani, N., Johnson-Roberson, M., Williams, S., Roman, C., Pizarro, O., Vaughn, I., Inglis, G.: Fft-based terrain segmentation for underwater mapping. In: The 2012 Robotics : Science and Systems Conference (RSS) (2012)

5. Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, A., Morton, P., Frenkel, A.: On the segmentation of 3d lidar point clouds. In: International Conference on Robotics and Automation (ICRA), pp. 2798–2805. Shanghai, China (2011)
6. Douillard, B., Underwood, J., Melkumyan, N., Singh, S., Vasudevan, S., Brunner, C., Quadros, A.: Hybrid elevation maps : 3d sruface models for segmentation. In: International Conference on Intelligent Robots and Systems (IROS), pp. 1532–1538. Taiwan (2010)
7. Filitchkin, P., Byl, K.: Feature-based terrain classification for littledog. In: Internation Conference on Intelligent Robots and Systems (IROS), pp. 1387–1392 (2012)
8. Gu, J., Cao, Q., Huang, Y.: Rapid traversability assessment in 2.5d grid-based map on rough terrain. International Journal of Advanced Robotic Systems **5**(4), 389–394 (2008)
9. Guo, Y., Song, A., Bao, J., Tang, H., Cui, J.: A combination of terrain prediction and correction for search and rescue robot autonomous navigation. International Journal of Advanced Robotic System **6**(3), 207–214 (2009)
10. Hadjiliadis, O., Stamos, I.: Sequential classification in point clouds of urban scenes. In: Fifth International Symposium on 3D Data Processing, Visualization and Transmission (2010)
11. Hadsell, R., Bagnell, J., Huber, D., Hebert, M.: Non-stationary space-carving kernels for accurate rough terrain estimation. International Journal of Robotics Research (IJRR) **29**(8), 981–996 (2010)
12. Hemakumara, M., Sukkarieh, S.: Non-parametric uav system identification with dependent gaussian processes. In: International Conference on Robotics and Automation (ICRA), pp. 4435–4441 (2010)
13. Himmelsbach, M., Hundelshausen, F., Wuensche, H.: Fast segmentation of 3d point clouds for ground vehicles. In: IEEE Intelligent Vehicles Symposium (IV), pp. 560–565. USA (2010)
14. Hu, T., Wu, T., Xu, X.: Stereo matching using weighted dynamic programming on a single-direction four-connected tree. Computer Vision and Image Understanding (CVIU) (2012)
15. Inc, V.L.: HDL 64E S2 Uesr's Manual. Morgan Hill, CA 95037 (2008)
16. Lai, K., Fox, D.: Object recognition in 3d point clouds using web data and domain adaptation. International Journal of Robotics Research (IJRR) (2010)
17. Lang, T., Plagemann, C., Burgard, W.: Adaptive non-stationary kernel regression for terrain modeling. In: Robotics : Science and systems Conference (RSS) (2007)
18. Muhammad, N., Lacroix, S.: Calibration of a rotating multi-beam lidar. In: International Conference on Intelligent Robots and Systems (IROS), pp. 5648–5653. Taiwan (2010)
19. Nguyen, V., Martinelli, A., Tomatis, N., Siegwart, R.: A comparison of line extraction algorithms using 2d laser rangefinder for indoor mobile robotics. In: International Conference on Intelligent Robots and Systems (IROS), pp. 1929–1934. Canada (2005)
20. Ortigosa, N., Morillas, S., Peris-Fajarnes, G.: Obstacle-free pathway detection by means of depth maps. Journal of Intelligent and Robotic Systems **53**, 115–129 (2010)
21. Paciorek, C., Schervish, M.: Nonstationary covariance function for gaussian process regression. In: Processings of the Conference on Neural Information Processing Systems (NIPS) (2004)
22. Plageman, C., Kersting, K., Burgard, W.: Nonstationary gaussian process regression using point estimates of local smoothness. In: The European Conference on Machine Learning (ECML), pp. 204–219. Belgium (2008)
23. Plagemann, C., Mischke, S., Prentics, S., Kersting, K., Roy, N., Burgard, W.: Learning predictive terrain models for legged robot locomotion. In: International Conference on Intelligent Robots and Systems (IROS), pp. 3545–3552 (2008)
24. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. The MIT Press, England (2006)
25. Rusu, R., Cousins, S.: 3d is here: point cloud library (pcl). In: International Conference on Robotics and Automation (ICRA), pp. 1–4 (2011)
26. Stamos, I., Hadjiliadis, O., Zhang, H., Flynn, T.: Online algorithms for classification of urban objects in 3d point clouds. In: International Conference on 3D Imaging, Modeling, Processing, Visualization and transmission (3DIMPVT), pp. 332–339 (2012)
27. Sun, Z., Chen, Q., Nie, Y., Liu, D., He, H.: Ribbon model based path tracking method for autonomous land vehicle. In: Internation Conference on Intelligent Robots and Systems (IROS), pp. 1220–1226 (2012)
28. Sung, G., Kwak, D., Lyou, J.: Neural network based terrain classification using wavelet features. Journal of Intelligent and Robotic Systems **59**, 269–281 (2010)

29. Thrun, S., et al.: Stanley: The robot that won the darpa grand challenge. Journal of Filed Robotics (JFR) pp. 661–692 (2006)
30. Vasudevan, S., Ramos, F., Nettleton, E., Whyte, H.: Heteroscedastic gaussian processes for data fusion in large-scale terrain modeling. In: International Conference on Robotics and Automation (ICRA), pp. 3452–3459 (2010)
31. Vasudevan, S., Ramos, F., Nettleton, E., Whyte, H.: Non-stationary dependent gaussian processes for data fusion in large-scale terrain modeling. In: International Conference on Robotics and Automation (ICRA), pp. 1875–1882. Shanghai, China (2011)
32. Wang, M., Zhou, J., Tu, J., C.L., L.: Learning long-range terrain perception for autonomous mobile robots. International Journal of Advenced Robotic Systems **7**(1), 55–66 (2010)
33. Xiao, L., Dai, B., Wu, T., Fang, Y.: Unstructured road segmentation based on spare representation. In: National Conference on Image and Graphic. Changchun, China (2012)