2016 12th World Congress on Intelligent Control and Automation (WCICA)
June 12-15, 2016, Guilin, China

A Slope Detection Method Based on 3D LiDAR Suitable for Quadruped Robots

Xiangrui Meng, Zhiqiang Cao, Leijie Zhang, Shuo Wang, Chao Zhou

Abstract—**This paper introduces a slope detection method based on point cloud data from 3D LiDAR for quadruped robots in unknown environments. For quadruped robots, they need to adjust their gaits according to different slope angles to avoid some potential dangers. 3D LiDAR is used to gather point cloud data, which is superior to 2D LiDAR in speed and accuracy. In this paper, a slope detection method using bilateral filtering and RANSAC algorithms is discussed. The experiments of slope detection are fulfilled with the consideration of different angles of slopes and different orientations to the slopes, and the results demonstrate that errors of angle estimation is small.**

I. INTRODUCTION

Environment perception and obstacle detection are basic and important issues for mobile robots when they move in the unknown environments. Researches on environment perception are abundant for wheeled robots and tracked robots, but for quadruped robots, it is still challenging due to the fact that the legs of quadruped robots may be stuck in holes or stumbled by bumps and they cannot walk steadily on steep slopes. LiDAR and vision sensors are widely used for mobile robots. The complexity of stereo vision algorithms may prevent the quadruped robots from running in real-time, which attracts some researchers focusing on environmental sensing through LiDAR sensors. Traditional methods [1-3] usually use a single-layer LiDAR sensor, which needs a pan-tilt device to acquire 3D point cloud data. However, it takes a large amount of time to get enough 3D point cloud data, which cannot meet the requirement of on-line processing for mobile robots. Meanwhile, the data features of single-layer LiDAR and multi-layer LiDAR are not all the same, which should be considered for data processing. In order to ensure the real-time processing based on LiDAR sensor for mobile robots, 3D LiDAR sensors provide an effective solution.

Wurm *et al.* [4] proposed a method for mobile robots navigation by detecting vegetarians in unstructured environments. It trains SVM and LDA algorithms, and the experiments show that it can classify the terrain accurately. Weiss *et al.* [5] discussed Micro-Electro-Mechanical Systems based on 3D LiDAR sensors, which is better than traditional kinds of sensors like vision or single-layer sensors. RANSAC is utilized to estimate the ground and k-d tree and fast nearest neighbor search algorithms is adopted to detect maize row. However, its detection rate is not ideal during experiments, and the method cannot classify different crops. Ravankar *et al.* [6] proposed a map generation method in a cluttered and noisy environment. It uses clustering and morphological erosion techniques in the spatial and Hough domains, and straight-line maps are generated by SVD or a Hough transform of the centroids. Zhang *et al.* [7] proposed a method for 3D object recognition, which uses GentleBoost algorithm for training and classifying based on BA image. Besides, it can exclude ill-suited objects through semantics information. Drews-Jr. *et al.* [8] proposed a framework for novelty detection and segmentation in robotic maps using Gaussian Mixture Models (GMMs). It compares two alternative criteria to detect changes, including a greedy technique based on the Earth Mover's Distance and a structural matching algorithm. The experiments show that if the region of interest is located appropriately, the detection rate is well. Jung *et al.* [9] present a human detection algorithm and an obstacle avoidance algorithm for a marathoner service robot. It discussed the detection of human bodies by SVDD, and a proportional controller for marathoner following. The rate of following failure is only 3%. Sarkar *et al.* [10] proposed an offline method for maps building of indoor environments using line segments. A new formulation for identifying is constructed, and all line segments that represent the same plane in the environment can be merged into one. The experiments show that the method can build good maps in a variety of indoor environments. Gonzalez-Ruiz *et al.* [11] considered a map building method of obstacles including occluded ones for a team of mobile robots. The approach maps the visible parts of the environment using occupancy grid mapping method and map the parts that cannot be sensed directly by the laser scanners based on wireless channel measurements. And the latter mapping uses Bayesian compressive sensing and total variation minimization. So it can map a complex occluded structure with a team of mobile robots sensing system.

In this paper, we discuss a slope detection method for quadruped robots with a 3D LiDAR sensor. The paper is organized as follows. Section II states the slope detection problem for quadruped robots. The slope detection method using bilateral filtering and RANSAC algorithms is described in Section III in detail. In Section IV, experimental results of slope detection are given. Section V concludes the paper.

This work is supported in part by the National Natural Science Foundation of China under Grants 61233014, 61421004, and in part by the National High Technology Research and Development Program of China (863 Program) under Grant 2015AA042201, and in part by the national defense basic research program under Grant B132011××××, and in part by the Beijing National Science Foundation under Grants 4152054, 4161002.

Xiangrui Meng, Zhiqiang Cao, Leijie Zhang, Shuo Wang, Chao Zhou are with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, CAS, Beijing 100190, China. Zhiqiang Cao is the corresponding author. (e-mail: {mengxiangrui2014, zhiqiang.cao, zhangleijie2014, shuo.wang, chao.zhou}@ia.ac.cn)

978-1-4673-8414-8/16/$31.00 ©2016 IEEE

II. Problem Statement

When quadruped robots move in the unstructured environments, they may encounter slopes. Different slopes imply that the robots should generate different path planning and gaits in order to adapt to or avoid the slopes. For example, they cannot step onto steep slopes. The slope detection module based on 3D LiDAR is shown in Fig. 1. The point cloud data from the 3D LiDAR sensor mounted on the quadruped robot is sent to the slope detection module. The coordinate transformation and calibration are then executed. After the filtering and plane fitting are conducted based on the bilateral filtering algorithm and RANSAC algorithm, the estimated angle and the position of the slope will be calculated, which shall be sent to the central processing unit of the quadruped robot for better path planning and gait adjustments. The following part focuses on the detail procedures of slope detection module.

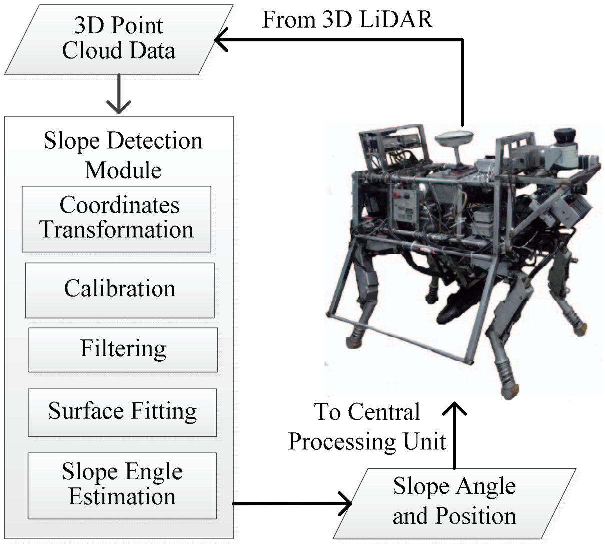

Fig. 1. Slope detection module based on 3D LiDAR for quadruped robot

Coordinate systems should be constructed firstly in order to solve the slope detection problem, including laser coordinate system and world coordinate system. The 3D LiDAR is supposed to be mounted on a height at about *h* meter with a tilt angle *α* between the LiDAR's upright direction and the vertical direction, as shown in Fig. 2. And the LiDAR is placed towards the ground. The laser coordinate system $O_lX_lY_lZ_l$ is established with its origin being the center point $O_l(x_l, y_l, z_l)$ of the 3D LiDAR sensor. The angle between *Y*-axis of $O_lX_lY_lZ_l$ and the moving direction of the robot is *α*, and *X*-axis of $O_lX_lY_lZ_l$ is perpendicular to the moving direction of the robot.

For a specific point *p*, its coordinate in the laser coordinate system is (x^l_p, y^l_p, z^l_p), and it can be computed in the laser coordinate system as follows.

$$\begin{cases} x^l_p = D_{lt}\sin(\pi-\beta)\sin\omega \\ y^l_p = D_{lt}\sin(\pi-\beta)\cos\omega \\ z^l_p = D_{lt}\cos(\pi-\beta) \end{cases} \quad (1)$$

where D_{lt} is the distance between the original O_l and the target point *p*, *β* is the angle between O_lZ_l and the line from O_l to *p*, *ω* is the azimuth angle between the current scanning line and the moving direction of the robot.

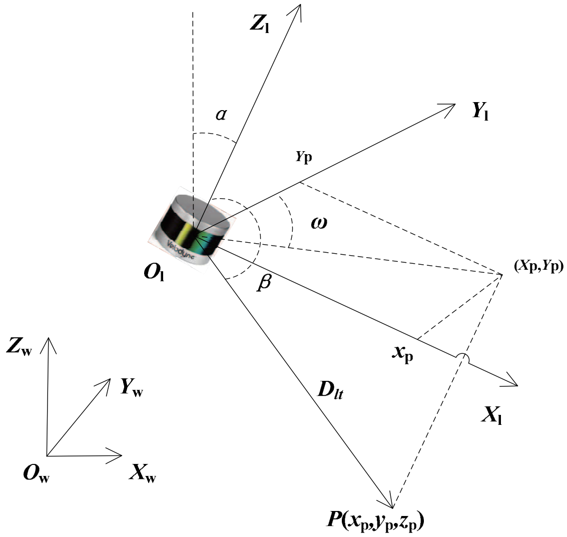

Fig. 2. Laser coordinate system and world coordinate system

As the robot is moving, point cloud data represented in laser coordinate system will be transformed into world coordinate system. $O_wX_wY_wZ_w$ is the world coordinate system. In order to transform the coordinates from laser coordinate system to the world coordinate system, the distance that the robot has covered is a necessary parameter, so acceleration and velocity of the robot are also needed to compute the distance. Thus, the coordinate of the specific point *p* mentioned before in the world coordinate system is given as follows.

$$\begin{pmatrix} x^w_p \\ y^w_p \\ z^w_p \\ 1 \end{pmatrix} = \begin{bmatrix} \cos\alpha & 0 & -\sin\alpha & s^w_{posx} \\ 0 & 1 & 0 & s^w_{posy} \\ \sin\alpha & 0 & \cos\alpha & -h \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x^l_p \\ y^l_p \\ z^l_p \\ 1 \end{pmatrix} \quad (2)$$

$$\begin{aligned} s^w_{pos} &= f(x^w_{pos}, y^w_{pos}, z^w_{pos}) = g(s^w_{posx}, s^w_{posy}, s^w_{posz}) \\ &= g(s^w_{posx}(a^w_{posx}, v^w_{posx}, t), s^w_{posy}(a^w_{posy}, v^w_{posy}, t), \quad (3) \\ &\quad s^w_{posz}(a^w_{posz}, v^w_{posz}, t)) \end{aligned}$$

where s^w_{pos} means the distance the robot has covered from the original point, *pos* is the current position of the robot, *f(x,y,z)* is function describes relation between the current position $q(x^w_{pos}, y^w_{pos}, z^w_{pos})$ and the distance s^w_{pos}, and so does *g(x,y,z)*. And s^w_{posx}, s^w_{posy}, s^w_{posz} are the distance of x^w, y^w and z^w component of s^w_{pos}, respectively, a^w_{posx}, a^w_{posy}, a^w_{posz}, are the x^w, y^w and z^w component of acceleration of the robot, v^w_{posx}, v^w_{posy}, v^w_{posz} are the x^w, y^w and z^w component of velocity of the robot, *t* is the runtime of the robot.

In the following section, filtering and plane fitting will be introduced to acquire the normal vector of the plane, which paves the foundation of slope angle estimation.

III. Slope Detection And Angle Estimation Using 3D Lidar

In this section, filtering is done first. The plane in the environment can be extracted by RANSAC algorithm, and the angle of the plane is used to decide whether there is a slope or not. The normal vector of the extracted plane is estimated in order to calculate the angle between the plane and the ground. And the angle of the surface is estimated. If the angle is estimated to 0° or 90°, we can assume that there is ground or a wall in front of the robot.

A. Data preprocessing using Bilateral Filtering

The point cloud data acquiring from LiDAR cannot be used directly because of noise, which will increase the uncertainty of estimation. The reasons for noises are various. Some of them are data reflected from tiny obstacles or holes and bumps in the environment and others are the measurement errors which cannot be avoided. After filtering, every specific point will be revised by its neighborhood data, and the original features should be preserved. In this case, this paper uses bilateral filtering algorithm [12, 13] to reduce the impact of the noise, which can be called edge-preserving filtering algorithm. An advantage of this algorithm is that the smoothing result is related not only to the locations of the point and those in its neighboring zone, but also to their values.

In order to calculate the new value of the point, the location kernel $\Phi_{\sigma d}$ and the value kernel $\Phi_{\sigma r}$ should be defined first:

$$\Phi_{\sigma_d} = \exp(-\frac{(i-k)^2+(j-l)^2}{2\sigma_d^2}) \tag{4}$$

$$\Phi_{\sigma_r} = \exp(-\frac{\left\|f(i,j)-f(k,l)\right\|^2}{2\sigma_r^2}) \tag{5}$$

For a point p, its new value and original value are defined as BF_p and I_p, respectively, and we have:

$$\mathrm{BF}_p = \frac{1}{W_p}\sum_{q\in S} I_p \Phi_{\sigma_d}(\|p-q\|)\Phi_{\sigma_r}\left(\left\|I_p - I_q\right\|\right) \tag{6}$$

where p denotes the target point, q means a neighborhood point of p, S is the neighborhood of p, and W_p is the normalization factor which is given by

$$W_p = \sum_{q\in S} \Phi_{\sigma_d}(\|p-q\|)\Phi_{\sigma_r}\left(\left\|I_p - I_q\right\|\right) \tag{7}$$

B. Slope Detection Based on RANSAC

Usually, there are two kinds of slope detection methods, one is straight line fitting, and the other is plane fitting. The former is easy to be affected by some obstacles whereas the latter is more likely to detect real slopes, including ground or walls. In this paper, the Random Sample Consensus Algorithm (RANSAC) [14] is used for plane fitting is used in this paper.

In order to detect the slope in the environment, we can define the plane equation as:

$$\rho = x\cos\alpha + y\cos\beta + z\cos\gamma \tag{8}$$

where ρ is the normal vector of the plane, which value is distance between the plane and the original point. α, β, γ are angles between the plane and three axis, respectively. Apart from certain number of points that fit the plane, the point dataset also has large amount of outliners, which should be excluded.

The process of extracting the normal vector is as follows. First of all, the algorithm selects three points randomly from the dataset to solve the equation (8). The plane and its parameters are computed using this minimal sample subset. Then, the algorithm will check whether other points from the dataset fit the fitting plane or not. And if the point is not within the error threshold that defines the maximum deviation, it will be considered as an outlier. The consensus set is built by the set of inliers obtained for the fitting plane. Afterwards, the plane is estimated using all the points from the consensus set. If there are sufficient points classified into inliers, the fitting plane will be considered good enough, or else, the procedure will repeat until it reaches the maximum iterations.

The number of required iterations can be calculated by probabilistic method. If r is defined as the probability of selecting a suitable model, equation (9) means that when the algorithm never selects a set of n points which all are inliers and the probability is 1-r:

$$1-\mathrm{r} = (1-w^n)^k \tag{9}$$

with:

$$\mathrm{w} = \frac{n(\text{inliners})}{n(\text{points})} \tag{10}$$

where w is the probability of choosing an inlier when a single point is selected, n is the minimum number of inliers, k is the number of iterations. So the number of required iterations is:

$$k = \frac{\log(1-\mathrm{r})}{\log(1-w^n)} \tag{11}$$

In this case, if we assume that the outliers are 50% of the dataset, and the probability to get a suitable plane is 0.95, then k can be computed by equation (11) and its value is about 23. However, the iteration times may be much larger than that in order to get a better fitting plane in the experiments.

Finally, the angle between the slope and the ground can be calculated by equation (12):

$$\varphi = \arccos(\rho \cdot \mathrm{u}_{xoy} / |\rho| \cdot |\mathrm{u}_{\mathrm{xoy}}|) \tag{12}$$

where u_{xoy} is the normal vector of $X_\mathrm{w}O_\mathrm{w}Y_\mathrm{w}$ plane in the world coordinate system.

IV. Experiments

The LiDAR sensor adopted in this paper is Velodyne VLP-16 LiDAR sensor. Unlike general single-layer LiDAR, this LiDAR supports 16 channels, a 360° horizontal field of view and a 30° vertical field of view, which make it suitable for detecting unknown environments. Distance, azimuth angles and intensity values are data measured by the 3D LiDAR.

As RGBD devices are more and more popular these years, the PCL (Point Cloud Library) [15] also develops quickly. It is an open source 3D point cloud data processing library, which is a part of ROS (Robot Operating System) initially. The PCL uses PCD (Point Cloud Data) file format, while Velodyne LiDARs use PCAP file format to store point cloud data. These involved data structures or file formats are different data representations of point cloud, but should be considered at first for the purpose of data acquisition. The point cloud library uses VTK (The Visualization Toolkit) for 3D visualization. The following experiments are fulfilled under a visualization platform, which is built under visual studio 2012 with PCL 1.8.0.

A. Slope Detection and Angle Estimation

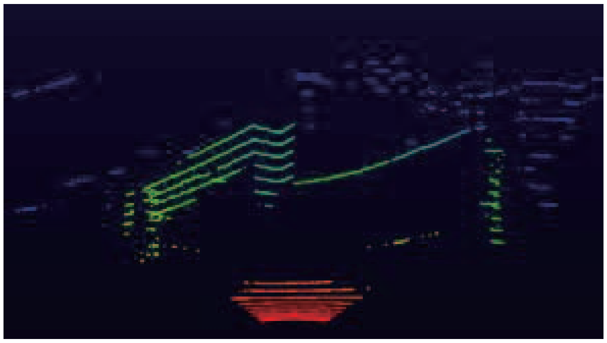

Fig. 3. The visualization of the environment (The color relates to the distance between the points and the LiDAR sensor. The warmer the color is, the shorter the distance is)

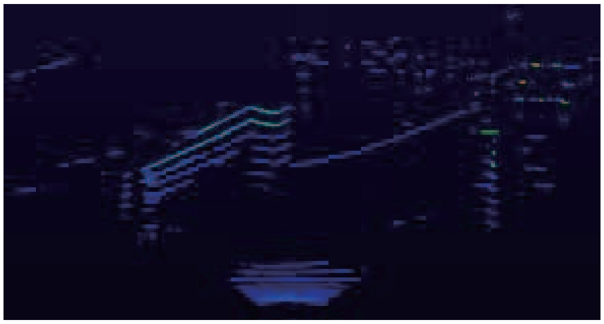

Fig. 4. Intensity values of the points in the environment (The similar color of the point means the similar intensity value)

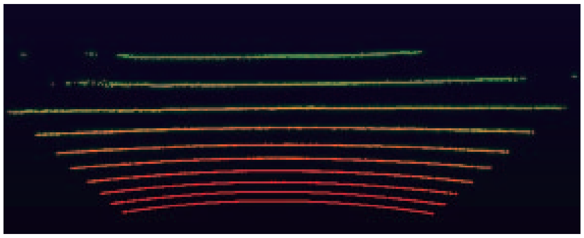

Fig. 5. Region of slope

The visualization of the environment including a slope is as shown in Fig. 3. The lines in the picture indicate the multi-layer point cloud data. And the different colors mean different distances between the target points and the 3D LiDAR sensor. The point cloud data should be preprocessed by bilateral filter algorithm which needs not only positions of the points but also the intensity values of the points. And the intensity values of the point cloud data are illustrated in Fig. 4. After filtering, RANSAC is executed to find an optimal plane in the environment while the outliers will be excluded. And the region of plane is got and illustrated in Fig. 5, which indicates a potential slope. The normal vector of the plane is as shown in Fig. 6.The angle of the slope is calculated by equation (12) with the normal vector generated by RANSAC. To analyze the accuracy of the slope detection method discussed above, the slopes of 10°, 15°, 20° and 25° are detected respectively. This experiment is repeated for 10 times for each specific angle of slope.

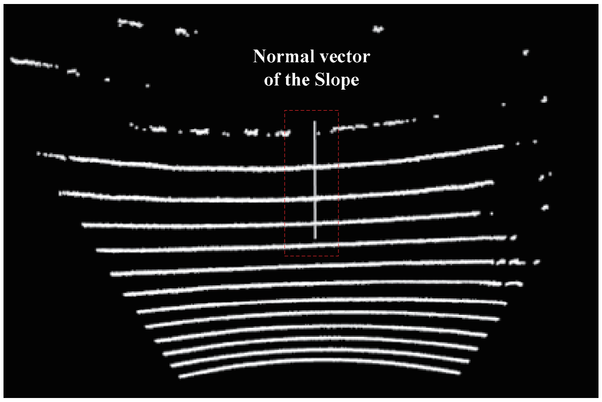

Fig. 6. Extracted slope surface and its normal vector

In Tab.1, $E(\theta)$ is the mean value of angle estimation, and *D(θ)* is variance of these values. We can see that the errors of angle estimation are below ±2°, and the variances of estimated values are small.

Tab.1 Slope detection with different angles

NO.	Actual Angles	E(θ)	D(θ)
1	10°	10.41°	1.018
2	15°	16.06°	1.238
3	20°	20.99°	0.946
4	25°	26.59°	0.258

B. Different Orientation of Slope Detection

The robot with a 3D LiDAR may not face the entrance of the slope directly, but often face the side of the slope. The normal vector of the slope also can be estimated by RANSAC in this case, as shown in Figs. 7-8. The corresponding slope angles for Fig. 7 and Fig. 8 are 20° and 25°, respectively.

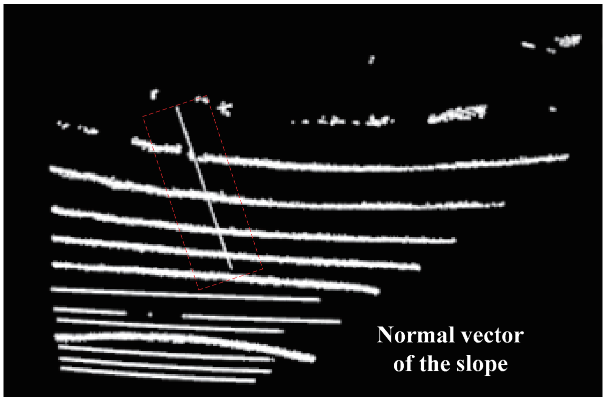

Fig. 7. The estimated normal vector (detected from left side of the slope)

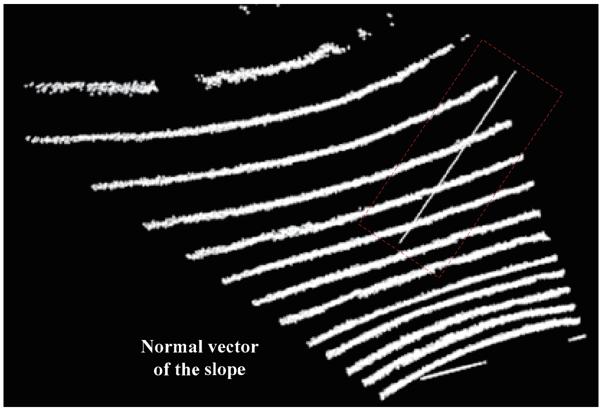

Fig. 8. The estimated normal vector (detected from right side of the slope)

Besides, for a given slope, some experiments are conducted with different orientations. The experimental results are shown in Tab. 2, where 10 different observation orientations are executed for each slope. It is shown that the mean values of the estimated slope angles are close to actual ones and their variances are acceptable. This experiment illustrates that the algorithm has rotational invariance to some extent.

Tab. 2. Slope detection from different orientations

NO.	Actual Angles	E(θ)	D(θ)
1	20°	21.47°	0.215
2	25°	26.58°	0.229

V. Summary And Future Work

Slopes detection is a necessary procedure for quadruped robots, because they should adjust the gaits considering the angles of the slopes. This paper discusses the slope detection method based on 3D LiDAR using RANSAC algorithm. The plane is fitted from the point cloud data and its normal vector is estimated. Then the angle of the slope is calculated by dot product between the normal vector ρ and the $X_wO_wY_w$ normal vector. Results show that the error of angle estimation is below ±2°, and the variance of estimated angles is small.

In the future work, slopes in the more complex environments will be analyzed and detected using improved algorithms. And terrain maps will be constructed using the 3D LiDAR with the information of detected slopes.

References

[1] S. Hu, C. Chen, A. Zhang, W. Sun, and L. Zhu, “A Small and Lightweight Autonomous Laser Mapping System without GPS,” *Journal of Field Robotics*, 30(5), pp. 784-802, 2013

[2] Y. An, Y. Zhuang, H. Hu, and F. Yan, “Two-dimensional laser-based environment exploration and recognition for service robots,” *Transactions of the Institute of Measurement and Control*, 35(8), pp. 1068-1084, 2013

[3] Q. Qiu, T. Yang, and J. Han, “A new real-time algorithm for off-road terrain estimation using laser data,” *Science in China Series F: Information Sciences*, 52(9), pp. 1658-1667, 2009

[4] K.M. Wurm, H. Kretzschmar, R. Kümmerle, C. Stachniss, and W. Burgard, “Identifying vegetation from laser data in structured outdoor environments,” *Robotics and Autonomous Systems*, 62(5), pp. 675-684, 2014

[5] U. Weiss, and P. Biber, “Plant detection and mapping for agricultural robots using a 3D LIDAR sensor,” *Robotics and Autonomous Systems*, 59(5), pp. 265-273, 2011

[6] A.A. Ravankar, Y. Hoshino, A. Ravankar, L. Jixin, T. Emaru, and Y. Kobayashi, “Algorithms and a Framework for Indoor Robot Mapping in a Noisy Environment using Clustering in Spatial and Hough Domains,” *International Journal of Advanced Robotic Systems*, 12(1), 2015

[7] Y. Zhuang, X. Lin, H. Hu, and G. Guo, “Using Scale Coordination and Semantic Information for Robust 3-D Object Recognition by a Service Robot,” *IEEE Sensors Journal*, 15(1), pp. 37-47, 2015

[8] P. Drews, P. Núñez, R.P. Rocha, M. Campos, and J. Dias, “Novelty detection and segmentation based on Gaussian mixture models: A case study in 3D robotic laser mapping,” *Robotics and Autonomous Systems*, 61(12), pp. 1696-1709, 2013

[9] E. Jung, J.H. Lee, B. Yi, J. Park, S. Yuta, and S. Noh, “Development of a Laser-Range-Finder-Based Human Tracking and Control Algorithm for a Marathoner Service Robot,” *IEEE/ASME Transactions on Mechatronics*, 19(6), pp. 1963-1976, 2014

[10] B. Sarkar, P.K. Pal, and D. Sarkar, “Building maps of indoor environments by merging line segments extracted from registered laser range scans,” *Robotics and Autonomous Systems*, 62(4), pp. 603-615, 2014

[11] A. Gonzalez-Ruiz, A. Ghaffarkhah, and Y. Mostofi, “An Integrated Framework for Obstacle Mapping With See-Through Capabilities Using Laser and Wireless Channel Measurements,” *IEEE Sensors Journal*, 14(1), pp. 25-38, 2014

[12] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand, “Bilateral Filtering: Theory and Applications,” *Foundations and Trends® in Computer Graphics and Vision*, 4(1), pp. 1-75, 2008

[13] C. Tomasi, and R. Manduchi, “Bilateral filtering for gray and color images,” in *Sixth International Conference on Computer Vision*, 1998, pp. 839-846

[14] M.A. Fischler, and R.C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, 24(6), pp. 381-395, 1981

[15] R.B. Rusu, and S. Cousins, “3D is here: Point cloud library (PCL),” in *IEEE International Conference on Robotics and Automation*, 2011, pp. 1-4