

Toward Accurate Road Detection in Challenging Environments Using 3D Point Clouds

Jaemin Byun, Beom-Su Seo, and Jihong Lee

In this paper, we propose a novel method for road recognition using 3D point clouds based on a Markov random field (MRF) framework in unstructured and complex road environments. The proposed method is focused on finding a solution for an analysis of traversable regions in challenging environments without considering an assumption that has been applied in many past studies; that is, that the surface of a road is ideally flat. The main contributions of this research are as follows: (a) guidelines for the best selection of the gradient value, the average height, the normal vectors, and the intensity value and (b) how to mathematically transform a road recognition problem into a classification problem that is based on MRF modeling in spatial and visual contexts. In our experiments, we used numerous scans acquired by an HDL-64E sensor mounted on an experimental vehicle. The results show that the proposed method is more robust and reliable than a conventional approach based on a quantity evaluation with ground truth data for a variety of challenging environments.

Keywords: Road detection, 3D point clouds, intelligent vehicle, MRF model, 3D LiDAR sensor.

I. Introduction

Interest in technology for autonomous navigation has dramatically increased along with the development of intelligent vehicles. In particular, more accurate environment recognition is required for safe driving on complex inner city roads and highways. Many studies are thus handling obstacle and drivable space detection for road environments. Many past researches have studied the detection of drivable spaces and obstacles using a camera, radar, and 2D LiDAR. Recent approaches have looked at transferring 2D data processing into 3D information through the use of a high-performance 3D LiDAR sensor and stereo camera [1]. In 2007, in the DARPA Urban Challenge, some teams, for the first time in the history of the challenge, used a 3D LiDAR sensor for drivable space and obstacle detection.

Recently, there have been a number of studies using 3D point cloud data [2]–[5]. However, most of the methods incorporated in these studies are focused on ground extraction on, ideally, a flat road and not on sloping or undulating roads, which are often found in more complex road environments.

We now move on to explain about previous related works for road and obstacle detection with 3D LiDAR. One of the most widely used methods for road and obstacle detection with 3D LiDAR is that of 3D point cloud projection. In this method, 3D point clouds are projected onto an assumed or estimated ground plane. Then, coordinates whose y-value (height) exceeds a given threshold are found. The ground plane is then represented by a grid, in which each cell contains only one representative value, such as an average, a maximum, or a minimum z-value of the number of points in a cell [6]–[7]. One of the advantages of this is that several sensors can be easily fused, thereby ensuring that any resulting mapping is straightforward.

Manuscript received Nov. 10, 2013; revised Mar. 1, 2015; accepted Mar. 19, 2015.

This work was supported by the IT R&D program of MSIP/IITP (B0101-15-0134, Development of Decision Making/Control Technology of Vehicle/Driver Cooperative Autonomous Driving System Based on ICT).

Jaemin Byun (corresponding author, bbhan97@gmail.com) and Beom-Su Seo (bseo@etri.re.kr) are with the IT Convergence Technology Research Laboratory, ETRI, Daejeon, Rep. of Korea.

Jihong Lee (jihong@cnu.ac.kr) is with the Department of Mechatronics Engineering, Chungnam National University, Daejeon, Rep. of Korea.

Many teams participating in the DARPA Urban Challenge successfully applied this method.

However, the difficulty of road detection remains in sloped terrains or situations with a large rolling/pitch angle of the host vehicle. Both J. Leonard and others [8] and M. Himmelsbach and others [9] describe a method that identifies points in a point cloud that are likely to be on the ground. The method then attempts to fit a ground model through those ground points. In addition, other points above the ground model are dealt with as obstacle points.

B. Douillard and others [10] proposed a strategy that utilizes ground models of non-constant resolution, either providing a continuous probabilistic surface or a terrain mesh built from the structure of a range image. F. Moosmann and others [11] proposed a graph-based approach to segment the ground and objects from 3D LiDAR scans using a novel generic criterion based on local convexity measures. D. Anguelov and others [12] and Y. Lu and C. Rasmussen [13] apply machine learning approaches to the segmentation process with 3D point clouds — the results of which show good performances.

The theory of Markov random fields (MRFs) can be considered through characterizing mutual influences among entities, such as points, cells, or nodes, under a conditional MRF distribution. In a supervised learning framework, an MRF is trained to label points with a *difference* label based on point features. However, inferring difference labels cannot be done in real time for a large number of points. C. Guo and others [14] used a graph-based approach for a 2D road representation of 3D point clouds with respect to the road topography. The approach describes the gradient cues of the road geometry to construct an MRF and implements a belief propagation (BP) algorithm to classify the road environment into four categories — a reachable region, a drivable region, an obstacle region, and an unknown region. However, their approach uses only a gradient value for labeling, and thus it sometimes cannot distinguish the ground from the roof of the

vehicle. M. Li and Q. Li [15] proposed a method called Four Directions Scan Line Gradient Criterion (4DSG), in which the gradient value of the height is calculated using neighboring points. 4DSG can reflect not only the flatness of pavements but also the distinguishing features of a point cloud on curbs in four directions. J. Bohren and others [4] proposed a method in which road points can also be detected based on the reflectivity of the ground in Velodyne scans. However, such a method only worked well under good conditions.

Unlike previous works, the research described in this paper addresses the categorization of six varying road types — FR, UR, DR, SR, MR, BR, and IR — including not only normal roads but also practical challenging roads that may be encountered during autonomous navigation (see Fig. 1). In addition, we propose a robust method of road detection under these complex environments. The outline of our work is shown in Fig. 2. The proposed approach differs from previous related work. The main contributions are as follows:

- An overall pipeline (flow and process) and proper combination (of process) for accurate road detection in an



Fig. 1. Experimental environment.

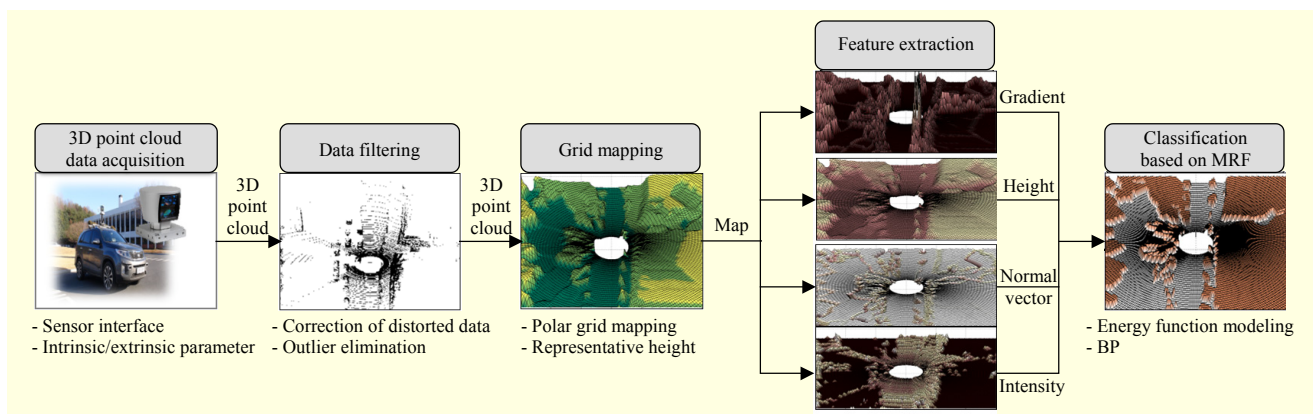


Fig. 2. Flow of overall detection method.

acceptable processing time.

- Guidelines for the best selection of the proper features using 3D geometrical information (3DGI) (the gradient value, the average height, the normal vectors, and the intensity value) from 3D point clouds in undulated and sloped roads such as downhill/uphill roads.
- A new formulation of the road detection problem based on MRF modeling through the selection of the proper likelihood term, as well as the determination of different regions with different classes through loopy belief propagation (LBP).
- The construction of a dataset for quantitative evaluation of various real roads, which is something that is not available in any existing public dataset, is used to show the performance of our work and that of relative works.

In Section I, we gave an outline of other relevant works, followed by a detailed description of our approach. Next, we give a detailed description of 3D point cloud representation and projection, in Section III. Section III describes how to select and extract the features. Section IV explains the method of classification based on MRF modeling. Then, experimental results are given in Section V. Finally, Section VI offers some concluding remarks regarding our works.

II. 3D Point Cloud Representation and Projection

1. Correction of Distorted 3D Point Cloud

The Velodyne LiDAR mounted to the top of the vehicle shown in Fig. 3 has 64 lasers to cover different vertical angles and provide a 360-degree field of view of the surrounding environment at a rate of more than 1.3 million points per second. The LiDAR returns spherical point coordinates and thus needs to transform these into points fit for a Cartesian space. For the transformation, we have to consider calibration parameters, such as distance correction factor (Δr), distance value (r) returned by a laser, vertical angle ($\Delta\phi$), rotational correction angle ($\Delta\theta$), vertical correction angle (ϕ), vertical/horizontal offset (r_v and r_h , respectively) [16].

The 3D point cloud computation required to transform the spherical point coordinates to those that are fit for a Cartesian space is as below

$$\mathbf{p} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} (r + \Delta r) \cdot \cos(\phi + \Delta\theta) \cdot \cos(\Delta\phi) + r_h \cdot \cos(\phi) \\ (r + \Delta r) \cdot \sin(\phi + \Delta\theta) \cdot \cos(\Delta\phi) + r_h \cdot \sin(\phi) \\ d \cdot \sin(\phi + \Delta\theta) - r_v \end{bmatrix}^T. \quad (1)$$

When using a Velodyne LiDAR mounted to the top of a car, we have to consider that the scanner takes a negligible amount of time to complete one full rotation; thus, the observed 3D point clouds are distorted by the motion of the vehicle. For

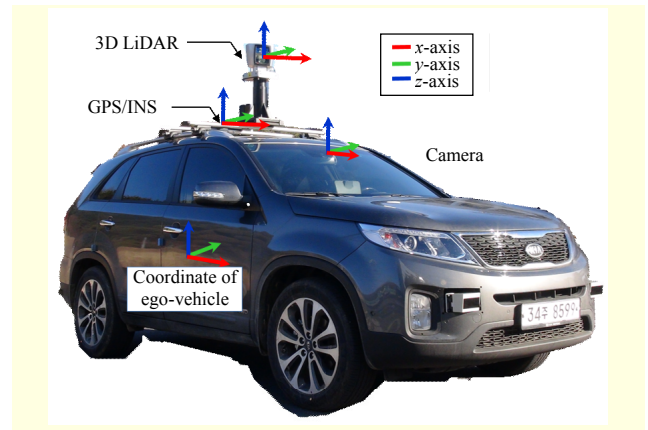


Fig. 3. Experimental vehicle.

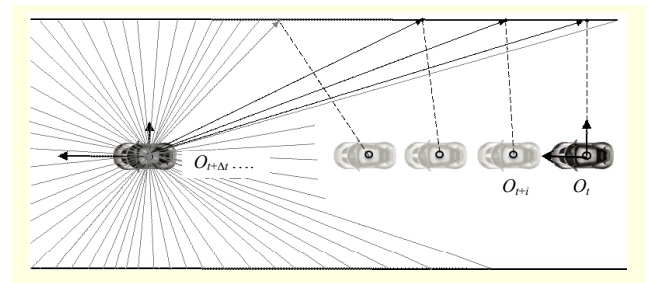


Fig. 4. Correction of distorted 3D point cloud data by estimation of vehicle motion.

instance, if the speed of the vehicle were 100 km/h (27.8 m/s), then, for each rotation of the Velodyne LiDAR (a single scan), it would be distorting the information of the 2.78 m just previously travelled, owing to the fact that the scanner is rotating with a frequency of 10 Hz (0.1 s). Furthermore, we should consider the case where the vehicle passes over a speed bump with a large rolling/pitch angle; in such a case, the road immediately in front of the vehicle is then classified as an obstacle, owing to the downward pitch of the vehicle at the time. We are able to solve these problems by estimating the ego-motion of the vehicle. To correct the aforementioned distortions, we utilize a GPS/INS unit that provides highly accurate motion information of the vehicle.

The laser measurement for each i th scan during one sensor rotation is referenced with respect to the vehicle's position and orientation, O_{t+i} , from the start of the sensor revolution, and is afterward transformed such that the coordinates are referenced with respect to $O_{t+\Delta}$ at the end of the revolution, as shown in Fig. 4. For a transformation with rotation $\tilde{\mathbf{R}}$ and translation $\tilde{\mathbf{t}}$ for all data, the undistorted coordinates $\mathbf{p}_{O_{t+\Delta}}^i$ of point $\mathbf{p}_{O_t}^i$ referenced with respect to $O_{t+\Delta}$ can be calculated as follows:

$$\mathbf{p}_{O_{t+\Delta}}^i = \tilde{\mathbf{R}}^i (\mathbf{p}_{O_t}^i - \tilde{\mathbf{t}}^i). \quad (2)$$

2. Polar Grid Map Representation

We can obtain over 1.3 million 3D point cloud points per second from a 3D LiDAR sensor such as the Velodyne 3D LiDAR for the recognition of the environment surrounding a vehicle, unlike the 2D information obtained from a 2D LiDAR and camera. A lot of time is required to compute all of these points for proper recognition. It is therefore necessary to use a method that can efficiently handle such computations. Such methods can be roughly categorized into four conventionally used types. The first is an elevation map that represents information in 2.5D using a horizontal plane constructed from rectangular grid cells that contain 3D point clouds; the 3DGI can be described through the elevation value corresponding to each cell.

Another method uses a *voxel* to represent a value on a regular grid in three-dimensional space. Various 3D structures can be described. However, this cannot guarantee a reduction in data size, because a lot of empty cells exist depending on how the density and resolution of the voxel are determined. In addition, this type of representation can consume large amounts of memory and has high complexity. Another method is to build a *range image* that can be generated from point cloud data and a given sensor position. However, this method has yet to be applied to the problem of road detection. This implies that it is more useful to recognize obstacles on a road than a road region.

The last method is a mesh grid mapping that is reconstructed in the form of a graph by connecting each node along the indices of the laser diode and the direction of the shot. The advantage of this is that there is a minimal loss of 3D data obtained from the sensor. On the other hand, the spatial relationships between a point and those points that are neighbors to it need to be confirmed, because the characteristics of these neighborhood regions are never the same. Finally, among the four methods described, our work is implemented using a modified version of the elevation map method. For this reason, the goal of our work focuses on the detection of a road using a horizontal plane, because the voxel method can lead to greater complexity and a higher cost in terms of computation. In addition, when constructing a mesh grid, we need to consider the spatial relationship at the cell level for segmentation with MRF modeling, which assumes spatial continuity constraints. Our proposed method therefore uses a projection for the construction of a 2.5D elevation map with 3D point clouds based on polar coordinates, unlike the Cartesian coordinate normally used by considering the scanning manner of a 3D LiDAR sensor. This sensor shoots 64 laser points at different tilt angles and rotates 360 degrees in such a way that point clouds that are close to the center of the

sensor have a higher density. When making a polar grid, this fact can efficiently make a map by minimizing the number of empty cells. The representative value at each node is an elevation value that is the difference between the max-min z -coordinate values among the points in the same cell.

III. Feature Extraction

This section describes how to select and extract the necessary features for distinguishing between a drivable area and a non-drivable area based on MRF. The recognition of drivable and non-drivable spaces should be considered with a geometrical structure of the environment surrounding the vehicle. To do so, past works use a height value for the detection of a road/obstacle through a comparison between the height value and its predetermined threshold value. However, this method may misinterpret a part of the road as an obstacle when the vehicle passes a downhill/uphill or sloped road. M. Li and Q. Li [15] proposed a method of 4DSG 4DSG not only can reflect the flatness of the pavement but also the distinguishing features of a point cloud on curbs in four directions. F. Moosmann and others [11] introduced a generic criterion based on local convexity measures for the segmentation of 3D LiDAR data in non-flat urban environments. C. Guo and others [14] focused on a gradient cue instead of the absolute height information, since it can reflect essential differences between drivable and non-drivable regions. This approach is similar to our work. However, under-segmentation can, unfortunately, occur because it uses only the gradient value; for example, the flat roof of a car cannot be distinguished from a road surface.

In this paper, we propose a novel feature vector (3DGI) that can describe the characteristics of a local road surface in a 3D environment to detect a region of a road using four types of local features: the magnitude of the gradient; the height value; the cosine similarity of the normal vector and the vector in the z -direction of the vehicle; and the intensity value at each node, as shown in Fig. 5. The combination of gradient value and height value is more robust in classification than using only the gradient value. For example, it distinguishes between the roof of a vehicle and a road by the height value, as mentioned before. The cosine similarity can distinguish features such as a wall with a vertical structure and a road. In addition, the intensity value at each node is useful information for the segmentation process; for example, the property of a material such as asphalt (see Fig. 5(d)). The feature vector of the i th node on the previously introduced map can be expressed as follows:

$$\mathbf{d}_i = [d_i^g, d_i^h, d_i^n, d_i^i]^T, \quad D = \{\mathbf{d}_1, \dots, \mathbf{d}_m\}, \quad (3)$$

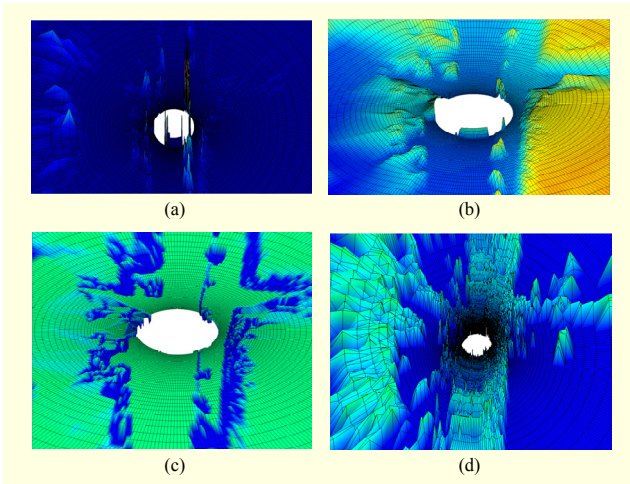


Fig. 5. Display of 3DGI features: (a) gradient value, (b) height value, (c) cosine similarity, and (d) intensity value.

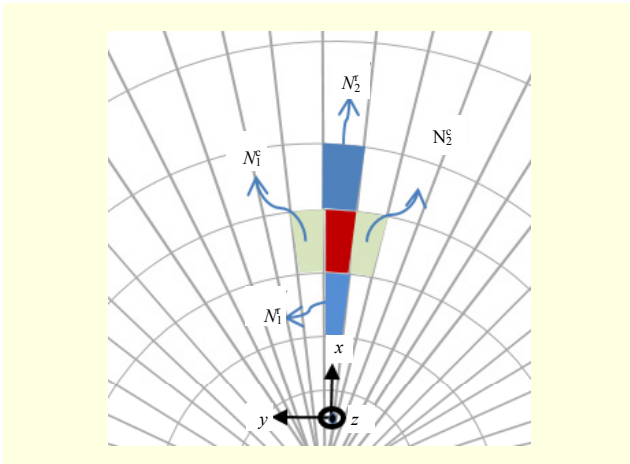


Fig. 6. Search of neighborhood nodes for gradient value.

where d_i^g is the magnitude of the gradient, d_i^h is the height, d_i^n is the similarity value of the normal vector, d_i^i is the intensity value, D is a set of feature vector, and m is the number of cells. First, for a calculation of the magnitude of the gradient, it is necessary to search the four-directed neighborhood nodes along the radial and circular axes, which are then the nearest points that have observation data from the current point in both directions, as shown in Fig. 6. We denote them as N_1^r , N_2^r , N_1^c , and N_2^c , where c indicates the circular direction, and r is the radial direction. The aforementioned magnitude of the gradient can be expressed as follows:

$$d_i^g = \sqrt{(d_i^{gr})^2 + (d_i^{gc})^2}. \quad (4)$$

The gradient of the radial direction is written as

$$d_i^{gr} = \frac{h(N_2^r) - h(N_1^r)}{\|N_2^r - N_1^r\|}. \quad (5)$$

Next, the gradient of the circular direction can be computed as

$$d_i^{gc} = \frac{h(N_2^c) - h(N_1^c)}{\|N_2^c - N_1^c\|}, \quad (6)$$

where $h(*)$ is the height value of its point. Second, the height of each grid cell is computed by median filtering using the current value and the value of the neighborhood nodes.

$$d_i^h = \max_{k \in I} Z(k) - \min_{k \in I} Z(k), \quad (7)$$

where $Z(k)$ is the z -coordinate value of points in the k th cell. Third, the estimated normal vector of i can be approximated normal to the neighborhood surface by performing a principal component analysis (PCA) on the neighborhood covariance matrix [17]. The eigenvector corresponding to the smallest eigenvalue gives an estimate of the normal vector. The estimated normal vector provides a good approximation of the surface normal, but the results obtained using a PCA are, in general, not consistently oriented. However, taking advantage of the known viewpoint, we can flip the normal vector that is not pointing toward this viewpoint. To do so, we flip all normal vectors that form an angle $\theta > 90^\circ$ between a normal vector and view vector, which is a vector from the center of the vehicle to the center point of the vehicle sensor. The cosine similarity can be described using the normal vector \mathbf{v}_n and vertical vector \mathbf{v}_z , which indicates the z -axis direction in the coordinate of the ego-vehicle.

$$d_i^n = \cos \theta = \frac{\mathbf{v}_n \cdot \mathbf{v}_z}{\|\mathbf{v}_n\| \|\mathbf{v}_z\|}. \quad (8)$$

When this value of similarity is zero, the point has a possibility not to be a part of the ground, but rather a vertical structure or obstacle. In addition, the intensity value can represent the characteristics of a material or surface; thus, this value is useful for distinguishing different objects.

IV. Classification Based on MRF Modeling

1. Formulation of Road Detection Based on MRF Model

In this paper, MRF modeling is used for extracting a road area with previous introduced features, \mathbf{d}_i , such as the gradient value, height, normal vector, and intensity. MRF theory provides a convenient and consistent way of modeling context-dependent entities such as image pixels and correlated features. This is achieved through characterizing mutual influences among such entities using a conditional MRF distribution. For the aforementioned reasons, MRF has been widely employed to solve vision problems. The segmentation based on MRF is to assign a label from label set L to each of the sites in S

according to the feature values. Thus, a label, f_i , is assigned from the set $L = \{road, non-road\}$ to site $i \in S$, where the elements in S index the map. Set f is

$$f = \{f_1, \dots, f_m\}. \quad (9)$$

According to Bayes' theory, when both the prior distribution and the likelihood function of a pattern are known, the best that can be estimated from these sources of knowledge is Bayes' labeling. The maximum a posteriori probability (MAP) solution, as a special case in the Bayes' framework, is

$$f^* = \operatorname{argmax}_f P(f | D). \quad (10)$$

In other words, a regularization solution is obtained by minimizing the energy of the form

$$E(f) = \sum_{i \in S} E_1(f_i) + \lambda \sum_{i \in S} \sum_{i' \in S, i' \neq i} E_2(f_i, f_{i'}), \quad (11)$$

where $E_2(f_i, f_{i'})$ is the cost of assigning labels f_i and $f_{i'}$ to two neighboring nodes; this is referred to as the discontinuity cost. Here, $E_1(f_i)$ is the cost of assigning label f_i to node i , which is referred to as the data cost, and λ controls the balance between $E_1(f_i)$ and $E_2(f_i, f_{i'})$. Next, we need to compute the posterior distribution from the prior and the likelihood. Actually, there have been no researches based on MRF to recognize a road area using 3D point information. We therefore try to provide a comparison of the results using previous similar techniques for modeling the cost function in a vision application.

First, we consider the Boykov and Jolly (BJ) segmentation algorithm, which was proposed in [18] for a monochrome image, in which a user imposes hard constraints by drawing a few strokes. To define the likelihood term, $E_1(f_i)$, BJ computed the foreground and background gray-level distributions. These were normalized to be used as foreground probability $P_1(i)$ and background probability $P_2(i)$ and provide the likelihood term with negative log-likelihoods as

$$E_1(f_i) = -\ln P_1(\mathbf{d}_i). \quad (12)$$

They defined the smoothing term as

$$E_2(f_i, f_{i'}) = \exp\{-\beta(\mathbf{d}_i - \mathbf{d}_{i'})^2\}, \quad (13)$$

where parameter β adjusts the sensitivity for the intensity difference. When the constant $\beta = 0$, the smoothing term is simply the well-known Ising interaction, and it is usually far more effective to set $\beta > 0$ and relax the smoothness tendency.

The Lazy Snapping algorithm [19], proposed by Y. Li and others, clusters the RGB intensities of pixels on the dataset using the k -means method. The centroids of the foreground and background clusters are denoted as $\{D_n^1\}$ and $\{D_m^2\}$, respectively, and n and m are the cluster indices. Computing the

minimum distance from d_i to the foreground clusters as $\text{Dist}_i^1 = \min_n \|\mathbf{d}_i - D_n^1\|$, $\text{Dist}_i^2 = \min_m \|\mathbf{d}_i - D_m^2\|$, they defined the likelihood as

$$E_1(f_i) = \frac{\text{Dist}_i^f}{\text{Dist}_i^1 + \text{Dist}_i^2}, \quad (14)$$

$$E_2(f_i, f_{i'}) = \frac{1}{\|d_i - d_{i'}\| + 1}, \quad (15)$$

where Dist_i^f is the distance from the center of the cluster to the corresponding f value; the number of clusters k is determined by an experiment based on the performance criteria ($k = 15$ in our work).

Finally, we enhance the discrimination between a road/non-road using the above feature vectors and learning from the training data to extract the effective information for the segmentation. A support vector machine (SVM) [20] with a kernel trick provides one of the schemes for carrying out this task, which generates a more discriminative nonlinear classification boundary between a road and non-road. SVM is a powerful machine learning technique for binary classification, supported by a strong theoretical foundation and excellent empirical success.

We are given X training data $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M\}$ ($\mathbf{x}_i \in R$) and their labels $\{y_1, y_2, \dots, y_M\}$ ($y_i \in \{Road, Non-Road\}$). The SVM output function can be expressed as

$$f_{\text{SVM}}(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + \mathbf{b}, \quad (16)$$

where superscript T represents the transpose operator, and \mathbf{w} and \mathbf{b} are the weight vector and bias, respectively, which should be determined appropriately through learning. Input vector \mathbf{x} is classified according to f_{SVM} . Geometrically, \mathbf{w} indicates the normal vector of a hyperplane. All vectors lying on one side of the hyperplane are classified as a road, and all vectors on the other side are classified as a non-road. We construct the likelihood term based on the probabilistic outputs of SVM (P-SVM). Note that the training data are the proposed feature vectors. We restrict the values of the likelihood and smoothing terms in the cost function to the range $[0, 1]$.

$$E_1(f_i) = -\log(P(\mathbf{d}_i | f_i)), \quad (17)$$

$$P(\mathbf{d}_i | f_i) = \frac{1}{1 + \exp(a_0 + a_1 f_i)}. \quad (18)$$

The logistic function is useful for converting the SVM outputs, such as the probability, into values between 0 and 1. The coefficients a_0 and a_1 should be optimally tuned for the road and non-road training data, respectively.

$$P(\mathbf{d}_i | f_i) = -\log\left(\frac{1}{1 + \exp(a_0 + a_1 f_i)}\right). \quad (19)$$

To ensure the local consistency, we define the smoothness cost function as

$$E_2(f_i, f_{i'}) = \min(|f_i - f_{i'}|, \varepsilon), \quad (20)$$

where ε is a truncation value.

2. Road Classification by LBP

The max-product BP algorithm works by passing messages around the graph defined by the four connected grids. Each message is a vector of the dimension given by the number of possible levels. Let $m_{ii'}^t$ be the message that node i sends to a neighboring node i' at time t . When using negative log probabilities, all entries in $m_{ii'}^0$ are initialized to zero, and at each iteration, new messages are computed in the following way:

$$m_{ii'}^t(f_{i'}) = \min_{f_i} \left(E_2(f_i, f_{i'}) + E_1(f_i) + \sum_{i \in S} \sum_{j \in S, j \neq i} m_{ji}^{t-1}(f_i) \right). \quad (21)$$

After T iterations, a belief vector is computed for each node as follows:

$$\mathbf{b}_{i'}(f_{i'}) = E_1(f_{i'}) + \sum_{i \in S} \sum_{j \in S, j \neq i} m_{ji}^T(f_{i'}). \quad (22)$$

Finally, the label f^* that minimizes $b_{i'}(f_{i'})$ individually at each node is selected [21].

V. Experiments

1. Experimental Setup

This section presents the experimental results of our work. To show the validation of our work, we collected real 3D point cloud data using experimental vehicles equipped with a 3D LiDAR sensor, such as a Velodyne sensor, under various road environments, such as a flat road, uphill, downhill, and sloped road, as shown in Fig. 3.

The software development environment consists of an ROS and OPRoS framework [22]. We employ the PCL library [23] for pre-processing with a 3D point cloud and use C++ language for an implementation of our work.

We then validate the proposed method by a quantity evaluation. In addition, we compare the performance of our approach and previous related works ([14], [9], and [24]) for various challenging road environments. In the first experiment, the procedure of the proposed method is sequentially performed, as shown in Fig. 2. First, 3D point cloud points obtained by a Velodyne sensor are used to compose a grid map based on a polar coordinate system performance. The best accuracy rate of detection was achieved at $\varepsilon = 0.5$. In the second experiment, for the best selection of the likelihood

terms (Gaussian distribution, k -means clustering, P-SVM), a comparison of the performance according to what is selected is performed. This was accomplished with a hand-labeled ground truth on three kinds of road types. In the case of the first scenario, like the uphill road, the result of classification with a Gaussian distribution method shows that a road/non-road is not perfectly separated owing to the large gap in the various heights. When the k -means clustering method is applied, the value of k is most influential. To determine the optimal k value in our problem, we tried to conduct an experiment on the performance variance according to the k value. Because a lesser value does not provide a good result, and a value larger than 15 requires too much time, the performance of classification cannot be largely enhanced; thus, we finally determined that the proper value of k is 15 based on our experiments. However, this method also shows that a false detection occurs in some parts of the vehicle's surroundings. We can see that the proposed method based on P-SVM provides more excellent results than other methods (Gaussian or k -means) in two scenes. For SVM, we use a radial basis function model as the kernel function. To obtain parameters C and γ , we use cross-validation and grid-search by LibSVM [20]. Their best accuracy was achieved at $C = 2$ and $\gamma = 8$. Unlike other methods, this method shows good classification results, not only for a median strip but also for a vehicle and ground area in front of the vehicle. Similar results can be seen in the case of a separated road (SR) type. We can see that some parts of the vehicle are misclassified as a region of road. In conclusion, we can see a reduction in the rate of mismatching using the P-SVM method compared with other methods.

2. Results and Discussion

A. Quantitative Evaluations

To evaluate the quantitative performance of our road detection method, we used various real road types including those previously mentioned; that is, FR, UR, DR, MR, and IR. Because the few published datasets that exist are limited in terms of the number of frames and utilized road environments, we collected over 100 frames on the different aforementioned road types under a test environment (over 2 km), as can be seen in Fig. 1. We built up our dataset through hand labeling using a 3D point cloud editing tool for the ground truth on *road* and *non-road* areas in each frame. The quantitative evaluations with our dataset are based on three measurements, accuracy rate (AR), precision rate (PR), and recall rate (RR). The three measurements are defined as follows:

$$\text{Accuracy rate (\%)} = \frac{N_{TN} + N_{FP}}{N_{TP} + N_{FN} + N_{FP} + N_{TN}} \times 100, \quad (23)$$

Table 1. Performance of proposed method and related works [14], [9], and [24].

Method	Proposed approach			C. Guo et al. [14]			M. Himmelsbach et al. [9]			Silk method [24]		
	AR	PR	RR	AR	PR	RR	AR	PR	RR	AR	PR	RR
(a) FR	92.3	92.1	97.8	89.8	92.4	86.6	89.2	98.6	98.3	90.0	94.7	82.6
(b) UR	92.0	97.4	93.0	86.4	96.9	86.7	88.7	95.4	83.4	80.8	89.3	81.9
(c) DR	93.2	94.4	91.1	81.4	95.1	82.6	89.1	95.7	84.1	82.4	90.7	83.1
(d) MR	90.2	97.6	90.1	80.9	93.2	80.1	83.0	95.6	72.0	79.1	66.9	97.3
(e) SR	83.5	98.4	80.5	81.1	99.3	81.0	72.9	91.4	61.2	93.3	87.4	88.5
(f) IR	95.7	98.1	99.7	92.4	94.2	84.4	80.2	95.0	77.6	73.2	51.4	98.3
Total	91.2	96.3	91.3	85.3	95.1	83.5	83.8	95.3	79.4	83.1	80.1	88.6

$$\text{Precision rate (\%)} = \frac{N_{TP}}{N_{TP} + N_{FP}} \times 100, \quad (24)$$

$$\text{Recall rate (\%)} = \frac{N_{TP}}{N_{TP} + N_{FN}} \times 100, \quad (25)$$

where N_{TP} is the number of road cells labeled as road in the reference scene, N_{FP} is the number of non-road cells erroneously labeled as road (false positives) in the reference scene, N_{FN} is the number of road pixels erroneously labeled as non-road (false negatives) in the reference scene, and N_{TN} is the number of non-road cells labeled as non-road in the reference scene. Table 1 shows some performance results of the proposed method for the six different road types and those obtained with the methods in [14], [9], and [24], respectively. First, the results of our method in the case of the flat road (FR) type are shown in Table 1. Similar results are obtained by all methods for this road type (92.3%, 89.8%, 89.2%, and 90.0%, respectively). Guo's method, which is most similar to our own, shows that misclassification has happened at the boundary between vehicle and road because of a lack of point cloud information (see Fig. 7(m)). This is because Guo's method uses only a gradient value for labeling, but the gradient value of the vehicle roof is similar to that of the ground. The other two methods, [9] and [24], are based on ground plane modeling and show good detection results (approximately 90%) in the case of the FR type, unlike other road types (UR, DR, MR, SR, IR). In FR type, the Himmelsbach method [9] using the local plane model shows better results than Guo's method [14], which is based on an MRF model, as shown Table 1(b). We think that the line fitting method proposed by Himmelsbach gives a good result for the slope road along the radial direction, but it causes a problem particularly at the boundary region between the road and the non-road along the circular direction because of the discontinuity of the model (see Fig. 7(r)). The accuracy of the Silk method [24] is 80.8 %, and we think that this method is

not robust in the case of a sloped road because it uses only a plane model.

In Table 1, (d) and (e) show the separated and combined road types of various slope. From the results, we can see that the accuracy of two methods, [9] and [24], which are based on plane estimation, is not very high (see Figs. 7(s) and 7(w)). These road types cannot be described well by one model or even a few models because of the various height profiles.

However, our proposed method, which is based on a 3DGI feature and region-based labeling, did detect these road types accurately. The accuracy of our approach reached 90.2% and 83.5%, respectively, which is reasonable for complicated road conditions.

In Table 1, (f) shows another sloped-road scene with a T-junction. Our method and the algorithm in [14] gave good results among the four types of methods, though some ground points (false negative points) are misclassified as obstacles by the algorithms in [9] and [24] (as shown in Figs. 7(t) and 7(x)). This is because the root-mean-square error of the line fitting for a steep slope exceeds a predefined threshold.

B. Computation Time

As the algorithm is applied to the autonomous vehicle, the real-time performance of the road detection algorithm is our main focus. We have calculated the computation time of the proposed approach for numerous scenes acquired by our experimental vehicle. As can be seen from Fig. 8, the average computation time of our method is 0.150 s/scan.

Our method is slower than those that are based on plane estimation, yet it is much faster than the previous works of [12]–[13], which were based on an MRF model. The processing times of these methods, [12]–[13], with the exception of the training times, are similar at about 2.31 s/scan. Consequently, our approach can reach an almost real-time performance, because the refresh rate of the scan is 0.1 s/scan.

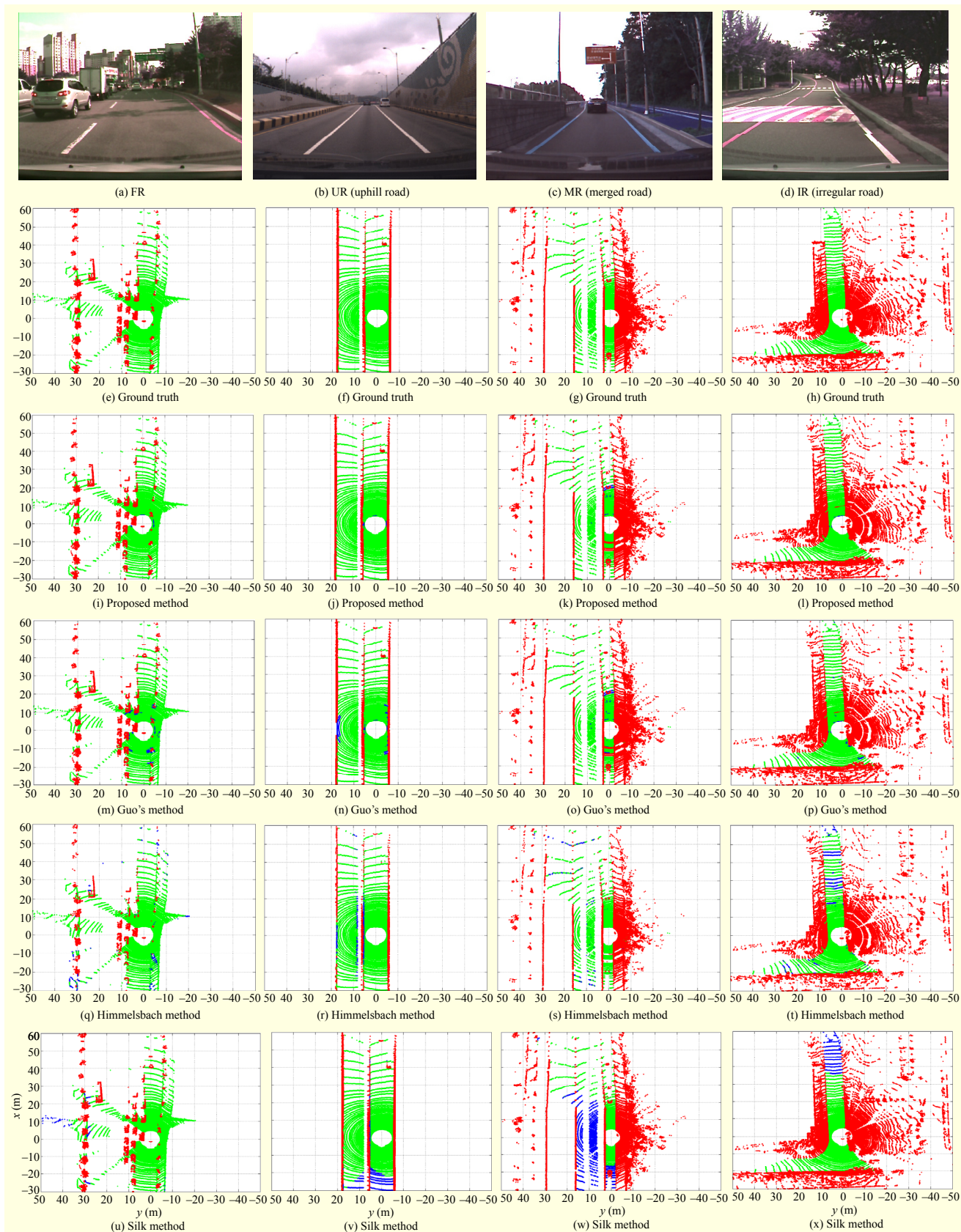


Fig. 7. Four examples in the 3D LiDAR datasets are presented. Their ground truth is labeled as ground (green) and other objects (red). Each algorithm's result is described as true positive (green), false positive (red), and false negative (blue), respectively.

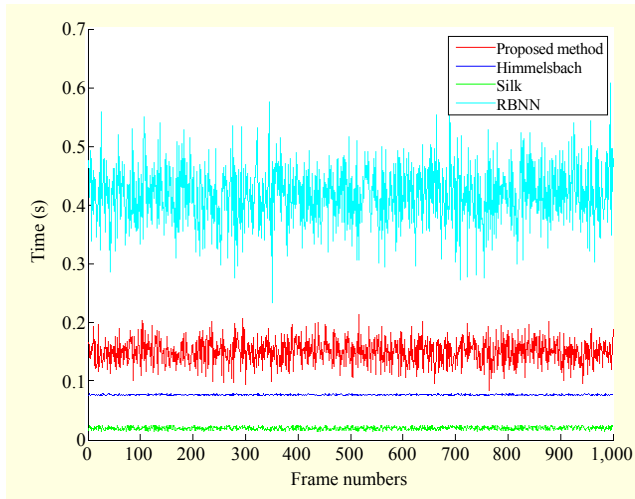


Fig. 8. Computation time of our method (red) compared to that of the Silk [24] (green), Himmelsbach [9] (blue), and RBNN [25] (cyan) methods.

VI. Conclusion

In this paper, we presented a robust method for road detection using 3D point clouds on challenging road environments such as downhill/uphill sloped roads. The correction of a 3D point cloud distorted by the motion of the vehicle was proposed. We introduced a guideline for the best selection of the proper features, such as the gradient value and average height of a neighboring node.

We have described, in detail, the transformation of the road detection problem into a classification problem of different features and an inference algorithm based on MRF with loopy belief propagation. In our experiments, the results proved that the proposed method is more robust and reliable than conventional approaches in a variety of challenging environments.

References

- [1] A. Broggi et al., "Terrain Mapping for Off-Road Autonomous Ground Vehicles Using Rational B-Spline Surfaces and Stereo Vision," *IEEE Intell. Vehicles Symp.*, Gold Coast, Australia, June 22–26, 2013, pp. 648–653.
- [2] C. Urmson et al., "Autonomous Driving in Urban Environments: Boss and the Urban Challenge," *J. Field Robot.*, vol. 25, no. 8, June 2008, pp. 425–466.
- [3] M. Montemerlo et al., "Junior: The Stanford Entry in the Urban Challenge," *J. Field Robot.*, vol. 25, no. 9, Sept. 2008, pp. 569–597.
- [4] J. Bohren et al., "Little Ben: The Ben Franklin Racing Team's Entry in the 2007 Darpa Urban Challenge," *J. Field Robot.*, vol. 25, no. 9, Sept. 2008, pp. 598–614.
- [5] M. Buehler, K. Iagnemma, and S. Singh, "The DARPA Urban Challenge: Autonomous Vehicles in City Traffic," Berlin, Germany: Springer, vol. 56, 2009.
- [6] S. Thrun, "Learning Occupancy Grid Maps with Forward Sensor Models," *Auton. Robots*, vol. 15, no. 2, Feb. 2003, pp. 111–127.
- [7] M. Tay et al., "An Efficient Formulation of the Bayesian Occupation Filter for Target Tracking in Dynamic Environments," *Int. J. Vehicle Auton. Syst.*, vol. 6, no. 1, Jan. 2008, pp. 155–171.
- [8] J. Leonard et al., "A Perception-Driven Autonomous Urban Vehicle," *J. Field Robot.*, vol. 25, no. 10, Oct. 2008, pp. 727–774.
- [9] M. Himmelsbach, F. Hundelshausen, and H. Wuensche, "Fast Segmentation of 3D Point Clouds for Ground Vehicles," *IEEE Intell. Vehicles Symp.*, San Diego, CA, USA, June 21–24, 2010, pp. 560–565.
- [10] B. Douillard et al., "On the Segmentation of 3D LiDAR Point Clouds," *IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 9–13, 2011, pp. 2798–2805.
- [11] F. Moosmann, O. Pink, and C. Stiller, "Segmentation of 3D Lidar Data in Non-flat Urban Environments Using a Local Convexity Criterion," *IEEE Intell. Vehicles Symp.*, Xi'an, China, June 3–5, 2009, pp. 215–220.
- [12] D. Anguelov et al., "Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data," *IEEE Comput. Vis. Pattern Recogn.*, vol. 2, June 20–25, 2005, pp. 169–176.
- [13] Y. Lu and C. Rasmussen, "Simplified Markov Random Fields for Efficient Semantic Labeling of 3D Point Clouds," *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vilamoura, Portugal, Oct. 7–12, 2012, pp. 2690–2697.
- [14] C. Guo et al., "Graph-Based 2D Road Representation of 3D Point Clouds for Intelligent Vehicles," *IEEE Intell. Vehicles Symp.*, Baden-Baden, Germany, June 5–9, 2011, pp. 715–721.
- [15] M. Li and Q. Li, "Real-Time Road Detection in 3D Point Clouds Using Four Directions Scan Line Gradient Criterion," *Future*, Xi'an, China, 2009, p. 5.
- [16] N. Muhammad and S. Lacroix, "Calibration of a Rotating Multi-beam Lidar," *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, Oct. 18–22, 2010, pp. 5648–5653.
- [17] R.B. Rusu, "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments," *KI-Künstliche Intelligenz*, vol. 24, no. 4, Nov. 2010, pp. 345–348.
- [18] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, Sept. 2004, pp. 1124–1137.
- [19] Y. Li et al., "Lazy Snapping," *ACM Trans. Graph.*, vol. 23, no. 3, Aug. 2004, pp. 303–308.
- [20] C.-C. Chang and C.-J. Lin, "LibSVM: A Library for Support Vector Machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, Apr. 2011, pp. 1–27.

- [21] P.F. Felzenszwalb and D.P. Huttenlocher, "Efficient Belief Propagation for Early Vision," *Int. J. Comput. Vis.*, vol. 70, no. 1, Oct. 2006, pp. 41–54.
- [22] C. Jang et al., "OPRoS: A New Component-Based Robot Software Platform," *ETRI J.*, vol. 32, no. 5, Oct. 2010, pp. 646–656.
- [23] R.B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," *IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 9–13, 2011, pp. 1–14.
- [24] S. Choi et al., "Robust Ground Plane Detection from 3D Point Clouds," *Int. Conf. Contr., Autom. Syst.*, Goyang, Rep. of Korea, Oct. 22–25, 2014, pp. 1076–1081.
- [25] K. Klasing, D. Wollherr, and M. Buss, "A Clustering Method for Efficient Segmentation of 3D Laser Data," *IEEE Int. Conf. Robot. Autom.*, Pasadena, CA, USA, May 19–23, 2008, pp. 4043–4048.



Jaemin Byun received his BS degree in mechatronics engineering and his MS degree in robotics from Chungnam National University, Daejeon, Rep. of Korea, in 2005 and 2007, respectively. From 2007 to 2008, he worked as a researcher engineer for Doddam Systems, Daejeon, Rep. of Korea. He is currently a senior researcher engineer of the Intelligent Robot Control Research Section, Electronics and Telecommunications Research Institute, Daejeon, Rep. of Korea. His main research interests are autonomous vehicles, environmental perception for navigation, and motion planning.



Beom-Su Seo received his BS and MS degrees in computer science and statistics from the University of Seoul, Rep. of Korea, in 1996 and 1998, respectively. He worked as a researcher engineer for the System Engineering Research Institute, Daejeon, Rep. of Korea, in 1998. He is currently a director of the Intelligent Robot Control Research Section, Electronics and Telecommunications Research Institute, Daejeon, Rep. of Korea.



Jihong Lee received his BS degree in electronics engineering from Seoul National University, Rep. of Korea, in 1983 and his MS and PhD degrees in robotics from the Korea Advanced Institute of Science and Technology, Daejeon, Rep. of Korea, in 1985 and 1991, respectively. From 1985 to 1988, he worked for Hyundai Heavy Industry, Ulsan, Rep. of Korea. Since 1994, he has been with the Department of Mechatronics Engineering, Chungnam National University, Daejeon, Rep. of Korea, where he is now a professor. His main research interests are high-speed mobile robots in rough terrain, underwater robots, and localization.