Auton Robot
DOI 10.1007/s10514-016-9588-7

An incremental nonparametric Bayesian clustering-based traversable region detection method

Honggu Lee[1] · **Kiho Kwak**[2] · **Sungho Jo**[1]

Received: 23 October 2015 / Accepted: 23 June 2016

Abstract Navigation capability in complex and unknown outdoor environments is one of the major requirements for an autonomous vehicle and a robot that perform tasks such as a military mission or planetary exploration. Robust traversability estimation in unknown environments would allow the vehicle or the robot to devise control and planning strategies to maximize their effectiveness. In this study, we present a self-supervised on-line learning architecture to estimate the traversability in complex and unknown outdoor environments. The proposed approach builds a model by clustering appearance data using the newly proposed incremental nonparametric Bayesian clustering algorithm. The clusters are then classified as being either traversable or non-traversable. Because our approach effectively groups unknown regions with similar properties, while the vehicle is in motion without human intervention, the vehicle can be deployed to new environments by automatically adapting to changing environmental conditions. We demonstrate the performance of the proposed clustering algorithm through intensive experiments using synthetic and real data and evaluate the viability of the traversability estimation using real data sets collected in outdoor environment.

Keywords Autonomous driving · Incremental clustering · Self-supervised learning · Dirichlet process mixture model · Traversability prediction

1 Introduction

Detecting traversable regions in outdoor environments is one of the major requirements for autonomous vehicles and robot navigation. In the past decade, traversability estimation and scene understanding to find traversable regions and obstacles have received considerable attention in the machine learning applications (Urmson et al. 2008; Trautmann and Ray 2011) and robot vision fields (Shneier et al. 2008; Geiger et al. 2013). Whereas the previous studies have successfully investigated traversability in outdoor environments, the primary concern of these works was to identify types of paved roads, such as in an urban or structured environment. These roads are typically characterized by their road color, lane-markings, or boundary features. However, unlike in urban environments, traversability estimation in complex and unknown outdoor environments that include dirt roads, foliage, surfaces covered with leaves and dense vegetation remains a challenging problem. More specifically, if a robot is placed in a complex and unknown outdoor environment without any prior information and navigates without human intervention, the robot should be able to identify traversable regions on-line.

Many researchers have focused on autonomous vehicles that are able to learn the properties of traversable region based on visual observations. One common approach is to train the system in a supervised fashion to handle several types of

Electronic supplementary material The online version of this article (doi:10.1007/s10514-016-9588-7) contains supplementary material, which is available to authorized users.

✉ Kiho Kwak
kkwak.add@gmail.com

✉ Sungho Jo
shjo@cs.kaist.ac.kr

Honggu Lee
nickplay@kaist.ac.kr

[1] School of Computing, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea

[2] Agency for Defense Development, Daejeon, South Korea

Published online: 02 July 2016

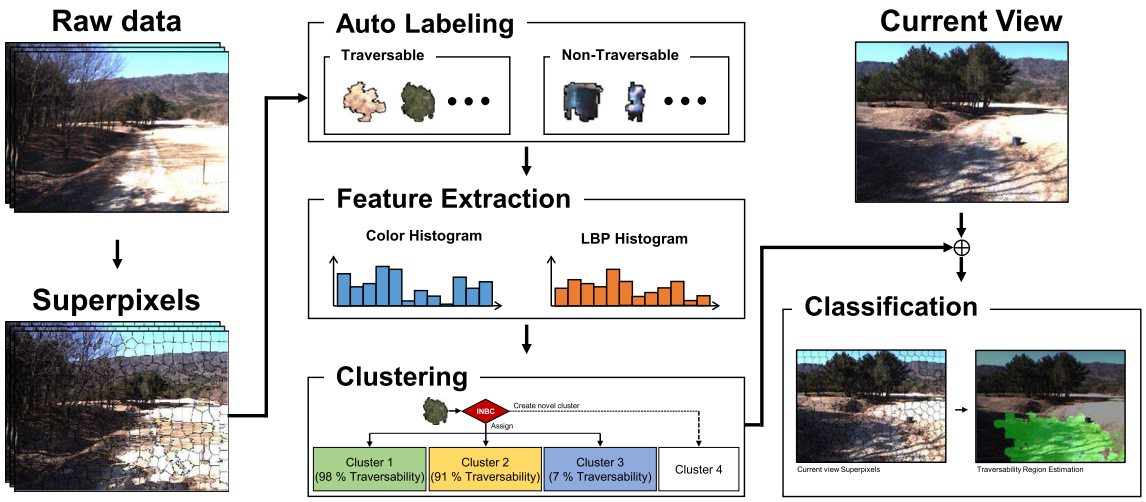

Fig. 1 Overview of the proposed traversable region detection approach based on incremental nonparametric Bayesian clustering

traversable regions (Thrun et al. 2007; Lalonde et al. 2006). In this case, the system requires a supervised classifier to predict the traversability of the terrain. To learn the classifier, data labeled by human experts are required. These approaches based on supervised learning are unlikely to work reliably in unknown or unstructured outdoor environments because the system assesses the traversability using an offline-learned model trained with specific terrain types. For example, if a system is trained with terrain samples in a specific season, the systems might not detect traversable regions in other seasons.

Unsupervised or self-supervised learning approaches may be the solution to the problems of the supervised learning approach for traversable region detection in complex or unknown outdoor environments (Wellington et al. 2006; Zhou et al. 2012; Sofman et al. 2006; Kim et al. 2006). If we assume that a vehicle starts on a traversable road and can therefore use the current road's appearance to assess roads encountered in the future, we could use an unsupervised or self-supervised learning framework to autonomously improve traversability estimation in unknown environments. In this framework, as the vehicle explores its environment, the classifier is trained incrementally with autonomously labeled training samples. These approaches provide promising results for urban or country roads, where the roads and background are clearly distinguished. However, they do not infer the number of clusters needed for the data but define the number of clusters a priori. Thus they provide less promising results in unstructured or unknown environments in the presence of various types of foliage and vegetation. These constraints make the existing unsupervised and self-supervised approaches difficult to adaptively operate in complex and unknown environments.

To overcome the limits of existing traversability estimation approaches based on self-supervised or unsupervised learning techniques, we propose a new traversable region detection method based on incremental nonparametric Bayesian clustering (INBC). The proposed approach determines whether unknown regions in front of a vehicle are drivable while the vehicle is in motion and without human's input. Therefore, the vehicle can be deployed to new environments by automatically adapting to changing environmental conditions. Our approach has three main stages: automatic region labeling, incremental model learning, and classification. In the automatic region labeling stage, we segment the input image into coherent areas (superpixels) based on intensity information and then extract training samples that are automatically labeled based on the outcome of the vehicles attempts to drive over certain terrain. These labeled training data represented by color and texture histograms are used as the input data in the incremental model learning stage. Using the samples, the system obtains the traversability model using INBC in the incremental model learning stage and estimates whether a region is traversable or not based on kNN approach in the classification stage. The overview of the proposed approach is shown in Fig. 1.

Our main contributions in this study are threefold. First, we develop a novel clustering algorithm that allows traversability assessment in on-line and determines the number of clusters without any prior knowledge. Second, we present a self-supervised learning architecture to estimate the traversable region in unknown and unstructured environments. Finally, we demonstrate the performance of our method using real data collected in complex outdoor environments and provide extensive experimental validations of the proposed method.

2 Related work

There are numerous works on the study of traversability perception using various learning strategies. In general, tra-

Table 1 Overview of related works

References	Learning	Model selection
Sun et al. (2006)	Supervised	Parametric
Kim et al. (2006)	Self-supervised	Parametric
Stavens and Thrun (2006)	Self-supervised	Nonparametric
Lalonde et al. (2006)	Supervised	Parametric
Thrun et al. (2007)	Supervised	Parametric
Bradley et al. (2007)	Supervised	Parametric
Angelova et al. (2007)	Self-supervised	Parametric
Munoz et al. (2009)	Supervised	Parametric
Silver et al. (2012)	Learning from demonstration	Nonparametric
Ott and Ramos (2012)	Self-supervised	Nonparametric
Häselich et al. (2013)	Supervised	Parametric
Ours	Self-supervised	Nonparametric

versability learning strategies can be categorized into two groups, supervised (Sun et al. 2006; Thrun et al. 2007; Bradley et al. 2007) and un (or self-) supervised (Kim et al. 2006; Stavens and Thrun 2006; Angelova et al. 2007; Santamaria-Navarro et al. 2015), where pre-defined training samples are provided in the former but not in the latter. Another way of dividing traversability learning strategies distinguishes between parametric (Lalonde et al. 2006; Munoz et al. 2009; Häselich et al. 2013) and nonparametric models (Stavens and Thrun 2006; Silver et al. 2012; Ott and Ramos 2012), where we assume that the model has a fixed structure in the former but is allowed to grow structurally in the latter. In Table 1, an overview of the related works is provided. Because our approach explores a sequence of self-supervised and nonparametric Bayesian approaches that infer the number of clusters from automatically collected training samples, we discuss related work focusing on each learning strategy, including supervised, self-supervised, parametric and nonparametric approaches. Moreover, we discuss some related algorithms for clustering.

Thrun et al. (2007) used a fusion of visual data, ground interaction measurements and auditory signals to estimate urban terrain types with a supervised learning approach. To acquire labels for training samples, the specific terrain region ahead of the vehicle is treated as a traversable region. Nearby terrain that is not traversed is treated as a non-traversable region. Using a supervised learning approach, Kelly et al. (2006) demonstrated the operation of a human-robot team for off-road navigation. They use multi-spectral image-based features, especially those extracted from the near-infrared range, for terrain classification.

Because the application of unsupervised learning methods completely eliminates the need for hand-labeled training samples, it is time and cost-efficient. However, for practical application, the quality of the classifier is often not enough to identify traversability. Sofman et al. (2006) recognized this problem and presented an approach that maps full-cost functions from near-to-far self-supervised learning, which involves the automatic training of a classifier without human intervention. Near-to-far learning involves building an association between near-range, higher-resolution and far-range, lower-resolution sensor information. Ho et al. (2014) also used a near-to-far learning approach to predict vehicle configuration on deformable terrain. Similarly, Zhou et al. (2012) adopted a direct experience based self-supervised learning approach to ground surface detection in forested terrain. As opposed to near-to-far learning, the labeling indicator does not come from a higher-resolution sensor but from a physical sensor such as a bumper. These direct experience-based self-supervised learning methods are broadly used to train the model by estimating various terrain perception properties, such as roughness (Stavens and Thrun 2006), amount of slip (Angelova et al. 2007) and geometric hazards (Kim et al. 2006). In our case, we employ a self-supervised learning approach based on direct experience, whereby various sensor fusion data are used to label terrain that was traversed and the associated visual appearance of the terrain.

Many of the methods used in the literature are closely related to the parametric approach, which is generally fast in practice and allows for easier application in an online setting. Lalonde et al. (2006) directly performed terrain classification over 3D point clouds. They categorized all observed examples into one of three discriminated object categories: sparse vegetation, solid surfaces or linear structures. Each terrain feature is defined as a distribution by fitting a Gaussian mixture model (GMM). Häselich et al. (2013) applied Markov random fields (MFRs) to a fused 3D range and camera image data. They define a set of features that describe different terrain classes based on the assumptions that rough terrain has an inhomogeneous texture and that obstacles have different height variations from those of negotiable terrain. Although these parametric approaches are preferable in the context of computation time and flexibility, it is generally difficult to find classifier rules for a variety of terrain types and assume the number of classes in unknown environments.

Kjærgaard et al. (2011) used a nonparametric approach based on Gaussian process regression (GPR) to estimate and model terrain curvature. This model is more flexible than the traditional parametric model. The prior terrain model is nonparametric, which allows two hypotheses, one for the ground measurements and the other for the measurements around obstacles, that are independent for each measurement point. Tse et al. (2015) applied MRFs to obtain a terrain height mapping model. Unlike previous MRF-based approaches (Angelova et al. 2007; Munoz et al. 2009; Häselich et al. 2013), they proposed terrain measurement MRFs in cases with no such assumption of terrain heights, and that fully represents uncertainty.

Similarly, our method builds clusters using a nonparametric approach. In detail, the objective of our nonparametric clustering method is to directly determine the proper number of classes from an observed dataset without a priori knowledge. In the parametric clustering case, the number of clusters k is the key parameter for accurate scene understanding. For example, if k is too small, several classes in the overall dataset become lumped together into a single cluster. One of the oldest and most popular parametric clustering algorithms is k-means (MacQueen et al. 1967) which partitions data into k non-overlapping regions. In the nonparametric clustering case, spectral clustering (Ng et al. 2002), mean shift (Comaniciu and Meer 2002), affinity propagation (AP) (Frey and Dueck 2007) and latent Dirichlet allocation (LDA) (Blei et al. 2003) are employed in real-world data analysis. The basic idea of these methods is to estimate the partition density of the data to discover clusters with different assumptions. In our work, we develop a density-based clustering model using a Dirichlet process mixture model. The proposed model provides flexible density estimation that enables clustering without a priori knowledge of the number of clusters. Furthermore, it is computationally inexpensive and incrementally adaptable to estimate traversability in real-time.

3 Background approaches

In this section, we briefly review the Dirichlet process mixture (DPM) models and Bayesian hierarchical clustering (BHC). These two approaches are the base elements of our proposed clustering algorithm.

3.1 Dirichlet process mixture

The DPM model is defined as a mixture model with a countably infinite number of components (Blei and Jordan 2006). Basically, the DPM uses a nonparametric prior based on the Dirichlet process (DP). Under the DP prior, it is possible to infer the correct model size such as the proper number of clusters even if the amount of data increases. Assuming a data point x_i to be drawn i.i.d. from the observation distribution, the dataset $\mathbf{X} = \{x_1, \ldots, x_N\}$ can be modeled with K components (or classes) in a finite mixture model, which is written as

$$p(\mathbf{X}|\boldsymbol{\Phi}) = \prod_{i=1}^{N} \sum_{k=1}^{K} p(x_i|\theta_k)\pi_k \tag{1}$$

where $\boldsymbol{\Phi} = \{\phi_1, \ldots, \phi_K\}$, $\phi_k = \{\theta_k, \pi_k\}$, θ_k is a set of parameters for component k, and π_k is a set of mixing weights whose sum is one.

In the Bayesian approach to a mixture model, the mixing weight π_k is specified as the probability of the component k under the multinomial distribution with $\pi_k = p(c_i = k|\boldsymbol{\pi})$ where c_i indicates the component assignments for data point i, $c_i \in \{1, \ldots, K\}$. Given the mixing weight $\boldsymbol{\pi}$, we treat all components as equivalently with a concentration parameter α/K in a symmetric Dirichlet distribution:

$$\begin{aligned} p(\boldsymbol{\pi}|\alpha) &\sim Dirichlet(\alpha/K, \ldots, \alpha/K) \\ &= \frac{\Gamma(\alpha)}{\Gamma(\alpha/K)^K} \prod_{k=1}^{K} \pi_k^{\alpha/K-1} \end{aligned} \tag{2}$$

where $\Gamma(\cdot)$ is the gamma function. Because mixture models assume that each component is independent, the joint distribution of the component indicator assignments is defined as a multinomial distribution

$$p(c_1, \ldots c_N|\boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{n_k} \tag{3}$$

where n_k is the occupation number, which is the number of data belonging to component k. With the prior on $\boldsymbol{\pi}$ and the joint distribution of the indicator variables $p(\mathbf{c}|\boldsymbol{\pi})$, we can directly simplify the prior on $\mathbf{c}$ in terms of the component indicator by integrating out the mixing weight:

$$\begin{aligned} p(\mathbf{c}|\alpha) &= \int p(c_1, \ldots, c_N|\pi_1, \ldots, \pi_K) p(\pi_1, \ldots, \pi_K) \\ &\quad d\pi_1 \cdots \pi_K \\ &= \frac{\Gamma(\alpha)}{\Gamma(N+\alpha)} \prod_{k=1}^{K} \frac{\Gamma(n_k + \alpha/K)}{\Gamma(\alpha/K)} \end{aligned} \tag{4}$$

Now, we consider the conditional prior of a single indicator given the others, using the exchangeability of the data-points prior. This is easily computed from the joint distribution in Eq. (4) by fixing all but the single indicator c_i:

$$p(c_i = k|\mathbf{c}_{-i}, \alpha) = \frac{n_{-i,k} + \alpha/K}{N - 1 + \alpha} \tag{5}$$

where $\mathbf{c}_{-i}$ is a set of indicators of all the other components except c_i and $n_{i,k}$ being the number of data points, excluding x_i, that are associated with component k. Finally, taking the limit as $K \to \infty$, we obtain the conditional prior for the Nth data:

$$p(c_i = k|\mathbf{c}_{-i}, \alpha) = \frac{n_{-i,k}}{N - 1 + \alpha} \tag{6}$$

$$p(c_i = K_{-i} + 1|\mathbf{c}_{-i}, \alpha) = \frac{\alpha}{N - 1 + \alpha} \tag{7}$$

where K_i is the number of components for which $n_{-i,k} > 0$. These probabilities are the same as the distribution over

partitions induced by the Chinese restaurant process (CRP), which is a predictive distribution of DP. In other words, the DPM is equivalent to the infinite limit of a finite mixture model and thus allows us to extend the number of components with the arrival of new data.

3.2 Bayesian hierarchical clustering

BHC (Heller and Ghahramani 2005) is a probabilistic model for hierarchical clustering. It is very similar to traditional bottom-up agglomerative clustering algorithms, except using marginal likelihoods instead of distance measure for similarity. Basically, the statistical hypothesis based on the DPM is used for deciding which data are to be merged into a single cluster. Because the BHC uses the DPM to define the generative model of the binary tree, it is possible to automatically decide which two trees to merge and to infer a proper number of clusters.

Let $\mathbf{X}_u$ be the set of data points at a single binary tree T_u. At each step, the BHC considers two trees T_{ul} and T_{ur} to merge into a new tree T_u, with the two hypotheses compared. The first hypothesis H_1 is that all data in $\mathbf{X}_u$ containing two data sets $\mathbf{X}_{ul}$ and $\mathbf{X}_{ur}$ are generated from a single cluster. In the alternative hypothesis H_2, $\mathbf{X}_u$ is assumed to be generated from each tree T_{ul} and T_{ur} independently. Combining the probability of the data under the two hypotheses H_1 and H_2, we can compute $p(\mathbf{X}_u|T_u)$, where T_u consists of the elements in $\mathbf{X}_u$. The probability of $\mathbf{X}_u$ in T_u is defined as

$$\begin{aligned} p(\mathbf{X}_u|T_u) &= p(H_1)p(\mathbf{X}_u|H_1) + p(H_2)p(\mathbf{X}_u|H_2) \\ &= p(H_1)p(\mathbf{X}_u|H_1) \\ &\quad + (1 - p(H_1))p(\mathbf{X}_{ul}|T_{ul})p(\mathbf{X}_{ur}|T_{ur}) \end{aligned} \tag{8}$$

where the prior $p(H_1)$ is recursively computed bottom-up in the DPM manner.

The posterior probability of the merged hypothesis $p(H_1|\mathbf{X}_u)$ is computed using Bayes' rule:

$$p(H_1|\mathbf{X}_u) = \frac{p(H_1)p(\mathbf{X}_u|H_1)}{p(\mathbf{X}_u|T_u)} \tag{9}$$

and if the posterior probability $p(H_1|\mathbf{X}_u) > 0.5$, which means that hypothesis H_1 is more probable than hypothesis H_2, the data $\mathbf{X}_u$ should be a single cluster. Consequently, the BHC algorithm builds a Bayesian mixture model where each tree node is a mixture component and cuts the tree at point spheres where $p(H_1|\mathbf{X}_u) < 0.5$.

4 Incremental nonparametric Bayesian clustering

One of the characteristics of hierarchical clustering is that it provides a tree structure for all datasets. Although the tree structure is intuitively interpreted to understand the dataset, it is sometimes unnecessarily complex for the simple clustering results. For example, with a dataset that consists of five clusters, the BHC builds tree structures of internal nodes to represent each of the clusters. The tree structures show the relationship between each data point in the clusters, but they do not represent important meanings in the entire dataset properly.

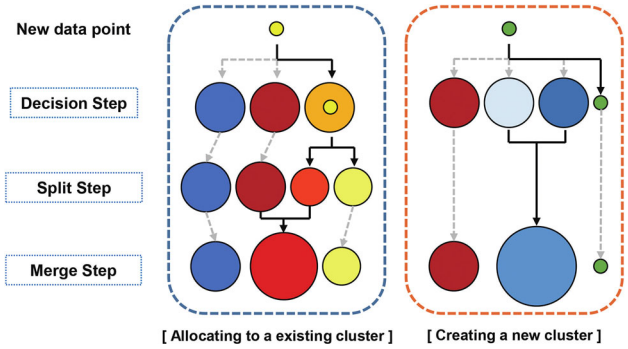

Fig. 2 Overall process of incremental nonparametric Bayesian clustering

To overcome the disadvantage of BHC, we introduce incremental nonparametric Bayesian clustering (INBC) to find the proper number of clusters and to converge quickly and reliably for incrementally evolving data. The proposed INBC collapses the tree structure into a single node representing a single cluster and avoids the processing of the batch-mannered approaches that are generally expensive in time and memory. To estimate the proper number of clusters, an inference method based on the DPM is used. The INBC performs a Bayesian hypothesis test for clustering for each new data input.

Our clustering approach consists of two main steps: (i) decision step, (ii) split and merge step. In the decision step, given the previous or initial clustering results, we assign a new input data point into one of the existing clusters or create a new cluster. In the split and merge step, if the new input data point is assigned to one of the existing clusters, our algorithm checks whether the clusters including the input data could be into two clusters for preventing under-segmentation. However, if a new cluster is created for the new input data, our algorithm checks whether the existing clusters excluding the new cluster could be merged into one cluster. Figure 2 describes the overall processes of incremental nonparametric Bayesian clustering.

4.1 Decision step: assignment and creation

Let x_t denote a single data point at a time t, c_t be the indicator component to cluster and $\mathbf{X} = \{x_1, \ldots, x_t\}$ be the set of stream data points until t. At each time step, a new data

point and the number of groups that were obtained from the previous or initial clustering result arrive. Ideally the probability of the new data point $p(x_t|\mathbf{X})$ can be represented with an associated predictive distribution that sums over all data points weighted by their posterior probabilities

$$p(x_t|\mathbf{X}) = \sum_{k=1}^{K} p(x_t|\mathbf{X}_k) w_k \tag{10}$$

where w_k is the weight mixing proportion of a cluster k and $\mathbf{X}_k$ is the set of data points in k. This equation is derived by considering every single data point to be partitioned into a sum over all clusters in the dataset, noting that a cluster can appear in many possible partitions. This is computationally very expensive and interactable to compute exactly. Instead, we restrict the equation to two possible predictions: (i) the new data are more likely to come from a previous dataset cluster, thereby maintaining the number of clusters, or (ii) the new data are unlikely to come from a previous dataset, so a new cluster is created. Using both alternatives, to maintain and to create, the probability of a new data point x_t given the dataset $\mathbf{X}$ can be computed by the Bayesian clustering model.

In the first case, the predictive distribution of maintaining the number of clusters is defined as

$$p(x_t, k^*|\mathbf{X}) = \sum_{k=1}^{K} p(x_t|\mathbf{X}_k) p(\mathbf{X}_k) \tag{11}$$

where k^* is the nearest cluster maximizing the marginal probability of the data x_t given cluster dataset $\mathbf{X}_k$

$$k^* = \arg\max_k p(x_t, c_t = k|\mathbf{X}_k) p(\mathbf{X}_k) \tag{12}$$

where the marginal probability $p(x_t, c_t = k|\mathbf{X}_k)$ is defined as $\int p(x_t|\theta_k) p(\theta_k|\mathbf{X}_k) d\theta_k$. This integral is tractable using the conjugate priors. However, the predictive distribution when there is a new cluster is defined as

$$\begin{aligned} p(x_t, K+1|\mathbf{X}) = &\sum_{k=1}^{K} p(\mathbf{X}_{-t}|\mathbf{X}_k) p(\mathbf{X}_k) \\ &+ p(x_t|\mathbf{X}_{K+1}) p(\mathbf{X}_{K+1}) \end{aligned} \tag{13}$$

where $\mathbf{X}_{-t}$ is the set of datasets except x_t. To determine whether the new data point belongs to the existing clusters or a new cluster, we compare the two predictive distributions. Let the comparing function $\mathcal{F}$ for two predictive distributions be defined as

$$\mathcal{F} = \frac{p(x_t, k^*|\mathbf{X})}{p(x_t, K+1|\mathbf{X})} \tag{14}$$

Using the marginal likelihood on the DPM, this comparing function is re-written as

$$\mathcal{F} = \frac{p(x_t|\mathbf{X}_{k^*}) \prod_{k_i=1}^{K} \Gamma(N_{k_i})}{\alpha p(x_t|\mathbf{X}_{K+1}) \prod_{k_j=1}^{K+1} \Gamma(N_{k_j})} = \frac{p(x_t|\theta_{k^*}) N_{k^*}}{\alpha p(x_t|\theta_{K+1})} \tag{15}$$

When a new data point x_t is delivered, we estimate that it belongs to existing clusters or does not belong to any of the clusters, so that a new cluster is created using the comparing function. Intuitively, if $\mathcal{F}$ is smaller than one, we create a new cluster and assign the new data-point to it; otherwise, the new data point belongs to the k^* cluster.

4.2 Split and merge step

One limitation of the comparing function is that it is possible to over- or under-fit to an existing number of clusters. For example, when the comparing function indicates that the new data point belongs to an existing cluster, it could be that the updated cluster no longer represents a given mixture model. Thus, we provide an opportunity to re-allocate the existing clusters using split and merge steps. When $\mathcal{F}$ is larger than one, the cluster k^* that contains x_t can be tested by splitting into two partitions. However, we test all pairs of clusters to merge.

Given the dataset associated with cluster k, we consider splitting or merging with three types of partition. In the split step, similar to the original BHC, we compare the current cluster k and product of the splitting clusters k_i and k_j. Cluster k_i and k_j are obtained from cluster k with a simple bisecting using general k-means algorithm. Then, the marginal probability of the data in the clusters with the prior distribution $p(\phi_k)$ corresponding to partition ϕ_k is,

$$p(\mathbf{X}_k) = p(\mathbf{X}_k|\phi_k) p(\phi_k) \tag{16}$$

$$p(\mathbf{X}_{k_i} \cup \mathbf{X}_{k_j}) = p(\mathbf{X}_{k_i}|\phi_{k_i}) p(\phi_{k_i}) p(\mathbf{X}_{k_j}|\phi_{k_j}) p(\phi_{k_j}) \tag{17}$$

Comparing these two possible marginal probabilities provides us with a loose optimal solution whether the current cluster k should be split or maintained in all dataset representations. Thus, we define a marginal likelihood ratio function that is related to the posterior probability in the Bayesian approach:

$$R_s(\mathbf{X}_k) = \frac{p(\mathbf{X}_k|\phi_k) p(\phi_k)}{p(\mathbf{X}_{k_i}|\phi_{k_i}) p(\phi_{k_i}) p(\mathbf{X}_{k_j}|\phi_{k_j}) p(\phi_{k_j})} \tag{18}$$

This equation is similar to the posterior probability of the merged hypothesis in the original BHC. However, we compare the differences in the sizes of datasets in clusters, not a single data point. The tree structure was modified using a likelihood ratio. With the DP prior, the ratio can be written as

Algorithm 1 INBC

Initialize Dataset $\mathbf{X} = \{\}$, Cluster index $\mathbf{c} = \{\}$, Initial Cluster Number $K = 1$ and hyperparameters r and v for the Normal-Inverse Wishart distribution prior
Input: x_t
repeat
  Updata Dataset $\mathbf{X} \leftarrow \mathbf{X} \cup x_t$
  Updata Parameters for NIW Prior $p(\mathbf{X})$
  Compute $\mathcal{F}$ and Find k^*
  if $\mathcal{F} > 1$ **then**
    Assign x_t to cluster k^*
    Compute $R_s(\mathbf{X}_{k^*})$
    if $R_s(\mathbf{X}_{k^*}) < 1$ **then**
      for $k = 1$ to K **do**
        Compute $R_m(\mathbf{X}_k, \mathbf{X}_{k_i^*})$ and $R_m(\mathbf{X}_k, \mathbf{X}_{k_i^*})$
      end for
      if The minimum $R_m(\mathbf{X}_k, \mathbf{X}_{k_i^*}) < 1$ **then**
        Assign $\mathbf{X}_{k_i^*}$ into $\mathbf{X}_k$
      else
        $k_i^* \leftarrow k$
      end if
      if The minimum $R_m(\mathbf{X}_k, \mathbf{X}_{k_j^*}) < 1$ **then**
        Assign $\mathbf{X}_{k_j^*}$ into $\mathbf{X}_k$
      else
        $k_j^* \leftarrow K + 1$
      end if
    end if
  else
    Compute $R_m(\mathbf{X}_k, \mathbf{X}_{k'})$ for all pairs of clusters
    if The minimum $R_m(\mathbf{X}_k, \mathbf{X}_{k'}) < 1$ **then**
      Assign $\mathbf{X}_{k'}$ into $\mathbf{X}_k$ and remove cluster k'
    end if
  end if
  for all dataset **do**
    Allocate single data to the cluster with the nearest mean
  end for
until no more input data x_t is available

$$R_s(\mathbf{X}_k) = \frac{\Gamma(N_k)p(\mathbf{X}_k|\phi_k)}{\alpha\Gamma(N_{k_i})\Gamma(N_{k_j})p(\mathbf{X}_{k_i}|\phi_{k_i})p(\mathbf{X}_{k_j}|\phi_{k_j})} \tag{19}$$

where $p(\mathbf{X}|\phi)$ is calculated with a conjugate prior. If $R_s(\mathbf{X}_k)$ is larger than 1, we maintain the cluster k with a new data point. Otherwise, if $R_s(\mathbf{X}_k)$ is smaller than 1, we split cluster k into clusters k_i and k_j, and the number of clusters is set to be $K = K + 1$.

When $\mathcal{F}$ is smaller than one, we assign the new data point to a novel cluster. In this case, there is a possibility that the existing clusters are over-segmented, so we proceed to the merge step. For the merge test, we define a marginal likelihood ratio function similar to $R_s(\mathbf{X}_k)$ in the split step. Let clusters k_i and k_j testing pair for the merge process, so the ratio $R_m(\mathbf{X}_{k_i}, \mathbf{X}_{k_j})$ is given by

$$R_m(\mathbf{X}_{k_i}, \mathbf{X}_{k_j}) = \frac{\alpha\Gamma(N_{k_i})\Gamma(N_{k_j})p(\mathbf{X}_{k_i}|\phi_{k_i})p(\mathbf{X}_{k_j}|\phi_{k_j})}{\Gamma(N_k)p(\mathbf{X}_k|\phi_k)} \tag{20}$$

Similar to $R_s(\mathbf{X}_k)$, if $R_m(\mathbf{X}_{k_i}, \mathbf{X}_{k_j})$ is smaller than 1, we determine to merge clusters k_i and k_j into k. The ratio R_s and R_m always give us evidence that $p(\mathbf{X}'_k|\phi'_k)p(\phi'_k) > p(\mathbf{X}_k|\phi_k)p(\phi_k)$, where k' is the split or merge possibility in each step. The entire procedure of our approach is summarized in Algorithm 1.

5 Traversability estimation

We present the traversability estimation approach in this section. Our algorithm consists of three steps: automatic region labeling, incremental model learning, and classification. These three key steps allow the robot to build a model that can adapt to an environment in a self-supervised and incremental learning fashion. Algorithm 2 shows the pseudocode including the steps.

Algorithm 2 Traversability Estimation Algorithm

Input: labeled data x_t and a novel superpixel y_s
repeat
  $k, \mathbf{X}_k \leftarrow$ **INBC**(x_t, k)
  cluster label l_k is assigned from $I[\frac{N_k^o}{N_k} \geq 0.5]$
  $p(l_s|y_s) \leftarrow k\mathbf{NN}(y_s, k, l_k, \mathbf{X}_k)$
until Stop driving

5.1 Feature extraction

Given an input image, we first obtain superpixels from the over-segmentation of the image. Superpixels are the result of a perceptual grouping of pixels and align better with the image edges than rectangular image patches. For each of the superpixels, we compute two different types of features: color and texture. For color features, we use a histogram in HSV color space that is known to be more perceptually uniform and robust against outdoor lighting conditions than RGB color space, and for texture features, we use the distribution of local binary patterns (LBP) (Ojala et al. 2002). We assume that each superpixel represents a whole object or a part of the object.

5.2 Region labeling

As the robot explores an environment, the superpixels are continuously generated from the current robot's view. However, at the initial position, the robot has no training examples at all. To generate training examples it needs experiences with the superpixel region corresponding to the real world environment region. The robot performs autonomous region labeling based on the robot's interaction with environments; the robot can adapt to unknown environments. In general,

even within a human perception system, it is hard to predict whether a region is traversable or not, in case where the ground is not directly observed due to tall weeds or grass. However, if we have experience of the environment, it provides environmental knowledge and an evidence about whether the region is traversable or non-traversable. From the interaction with its environment, the robot can associate previous superpixels with the current environmental information reported by its sensors.

To obtain environmental information, we implement the bumper system, inertial measurement unit (IMU), and wheel encoder data to obtain environmental information. The bumper system consists of two sensors: a 2D laser range-finder and switch sensor. Because the measurement from the switch sensor is only binary: hit or miss, it is hard to define a non-traversable region without human intervention. Thus, we integrate the state of vehicle information from the IMU and the wheel encoder into the bumper-measurement system. To integrate this information, we define the wheel slip ratio s as

$$s = 1 - \frac{v_w}{r_w \omega} \tag{21}$$

where v_w is the vehicle velocity from the IMU, r_w is the radius of the wheel and ω is the wheel rotational velocity from the wheel encoder. When detecting an object from the laser range-finder, the robot immediately slows down and interacts with the object using the switch sensor. The non-travesable region is defined as the wheel slip ratio s, increasing over 1 s.

5.3 Incremental model learning

The extracted features with labels are incrementally added into the INBC algorithm, as described in Sect. 4. The clustering task is computationally efficient and adaptive because of the properties of INBC. Moreover, we do not need to determine the number of clusters from the dataset. At each time step, the features are labeled into the highest marginal likelihood ratio group of all the clusters. The clusters represent the different characteristics of the terrain regions to describe traversable or non-traversable. This representation of the clusters is accomplished by a simple classifier,

$$p(\textit{non-traversable}) = I\left[\frac{N_k^o}{N_k} \geq 0.5\right] \tag{22}$$

where N_k^o is the number of non-traversable labels in cluster k and N_k is the total number of data in the cluster k. We determine that the cluster is a non-traversable region, if the number of non-traversable labels in a cluster is larger than the half of the total number of data points in the cluster.

As described in Sect. 4, our clustering model only considers the previous clustering result and the cluster probability with the current labeled features to build and to update. It is important to note that our proposed method can efficiently maintain several groups that represent a traversable region or a non-traversable region and can easily adapt to the novel features during the environmental changes.

5.4 Classification

The decision rule for estimating a new region should be computationally efficient for real-time operation. To classify the region, we use the k-nearest neighborhood (kNN) algorithm because of its simplicity and accuracy. From the INBC results, we have several clusters with traversable or non-traversable clusters. For each cluster, the similarity distance is computed for the newly observed region data. There are many distance functions for comparing histograms such as Euclidean, Manhattan, chi-square and Bhattacharyya. Generally, chi-square and Bhattacharyya give the best results for image histograms (Aherne et al. 1998). Thus, to compute the similarity using the kNN method, we adapt the Bhattacharyya distance measure as follows Comaniciu et al. (2003),

$$D_b(H_1, H_2) = \sqrt{1 - \sum_{i=1}^{n_b} \frac{\sqrt{H_1(i) H_2(i)}}{\sqrt{\sum_{i=1}^{n_b} H_1(i) \sum_{i=1}^{n_b} H_2(i)}}} \tag{23}$$

where n_b is the number of histogram bins, H_1 is the set of histograms that represent clusters, and H_2 represents a new observed region. For the Bhattacharyya distance, low scores indicate good matches between 0 and 1.

6 Experimental results

To verify the effectiveness and accuracy of the proposed clustering method, we show the performance of the INBC algorithm with three synthetic datasets and real-world datasets. We also compare our method with state-of-the-art algorithms for clustering performance. We then show the performance of the traversability estimation with respect to the classification result and adaptation result. In our implementation, we followed reference studies (Heller and Ghahramani 2005; Blei and Jordan 2006) for all parameter settings and used the Gaussian likelihood and Gaussian–Wishart prior for conjugate priors for the parameters:

$$p(\boldsymbol{\mu}, \boldsymbol{\Lambda}) = \mathcal{N}(\boldsymbol{\mu}|\mathbf{m}, (\tau, \boldsymbol{\Lambda})^{-1}) \mathcal{W}(\boldsymbol{\Lambda}|\mathbf{W}, \nu) \tag{24}$$

$$\mathcal{W}(\boldsymbol{\Lambda}|\mathbf{W}, \nu) \propto |\boldsymbol{\Lambda}|^{(\nu-d-1)/2} \exp\left(-\frac{1}{2}\mathrm{Tr}(\mathbf{W}^{-1}\boldsymbol{\Lambda})\right) \tag{25}$$

where $\boldsymbol{\mu}$ is the mean and $\boldsymbol{\Lambda}$ is a precision that is equal to the inverse of the covariance $\boldsymbol{\Sigma}$. The hyperparameters for the Gaussian–Wishart prior include $\mathbf{m}$, which is the prior mean

on the mean, $\mathbf{W}$, which is the prior mean on the precision matrix, τ, which is the scaling factor on the prior precision of the mean which we fixed as 0.01, and ν, which is the degree of freedom that we fixed as the number of feature dimensions plus two.

6.1 Synthetic datasets

6.1.1 Experimental setup

We first perform our algorithm for clustering results with synthetic datasets from Sirinukunwattana et al. (2013). The three synthetic datasets have different properties. All three synthetic datasets have 1000 observations of 10 dimensional random vectors with 7 distributions. The distribution of synthetic datasets 1 (S1) and 2 (S2) are drawn from a multivariate Gaussian distribution, although those of S1 is independent and those of S2 is correlated. Similar to S1 and S2, S3 has same observation, dimension and mixture size, but the multivariate components of the six mixtures are Gaussian, Gamma, uniform, Students t, Weibull and Chi-square distribution, which are independent. The last multivariate component of a mixture in S3 is a correlated Gaussian distribution.

6.1.2 Clustering results

We compared the INBC to the three state-of-the-art nonparametric clustering algorithms, affinity propagation (AP) (Frey and Dueck 2007), BHC (Heller and Ghahramani 2005), GBHC-NODE (Sirinukunwattana et al. 2013), and a traditional clustering algorithm, k-means (MacQueen et al. 1967), in synthetic dataset experiments. Although k-means is not a nonparametric model, it provides us a baseline to determine whether the prior knowledge can aid in the clustering.

To evaluate the clustering performance, we compute the adjusted Rand index (ARI) (Hubert and Arabie 1985) between the clustering result partition and the true partition. Let U and V represent two different partitions from the n number of objects in a set. Then, the contingency table for comparing partitions U and V is shown in Table 2. The equation for ARI to evaluate the clustering results is as follows:

$$ARI = \frac{\binom{n}{2}(a+b) - [(a+b)(a+c)+(c+d)(b+c)]}{\binom{a}{b}^2 - [(a+b)(a+c)+(c+d)(b+c)]} \tag{26}$$

with an expected score between 0 and 1, where a higher score indicates a higher agreement.

To demonstrate the claimed contributions of the incremental clustering algorithm, we randomly and sequentially add the data into the INBC algorithm and compute 20 times. We then evaluate the clustering performance aspect of accuracy and time. The ARI scores and computation time of clustering algorithm are shown in Tables 3 and 4.

Table 2 Contingency table for comparing partitions U and V

Partition	V	
U	Pair in same group	Pair in different groups
Pair in same group	a	b
Pair in different groups	c	d

Table 3 ARI for synthetic dataset clustering

Dataset	k-means	AP	BHC	GBHC-NODE	Ours
Synthetic 1	0.851	0.317	0.455	**1.000**	0.855 (±0.0702)
Synthetic 2	0.412	0.106	0.108	0.270	**0.637** (±0.0599)
Synthetic 3	0.750	0.295	0.162	**0.921**	0.728 (±0.0695)
Mean	0.671	0.239	0.242	0.730	**0.740**

Bold values indicate the best performance

Table 4 Computation time for synthetic dataset clustering

Dataset	k-means	AP	BHC	GBHC-NODE	Ours
Synthetic 1	11.4	**4.72**	142	85.1	18.7 (±2.68)
Synthetic 2	14.4	**5.53**	147	73.2	27.9 (±2.78)
Synthetic 3	14.7	**5.69**	147	74.3	17.5 (±2.44)
Mean	13.5	**5.31**	145	77.5	21.4

Bold values indicate the best performance

The results in Table 3 show that our algorithm is comparable to other batch mode clustering methods. GBHC-NODE, however, outperforms in S1 and S3, which can be explained in the way that GBHC is based on the strong assumption that all clusters are independent and generated from different Gaussian distributions. The important thing for clustering performance here is that our proposed algorithm performs better than the batch mode state-of-the art algorithms even though our method is developed for the incremental manner. The results in Table 4 show that our algorithm is faster than other algorithms except AP. INBC was approximately 3–4 times faster than GHBC and 5–7 times faster than BHC in our MATLAB implementation.

6.2 Real datasets

We then test our algorithm on real-world datasets. We recorded a dataset of video sequences of real outdoor environments at the Changwon test facility in Korea to apply our algorithm and characterize its performance. Figure 3 shows examples of the environment scenes and an overhead map view with driving courses. We recorded two different courses, but there are three different environments: travel course T1

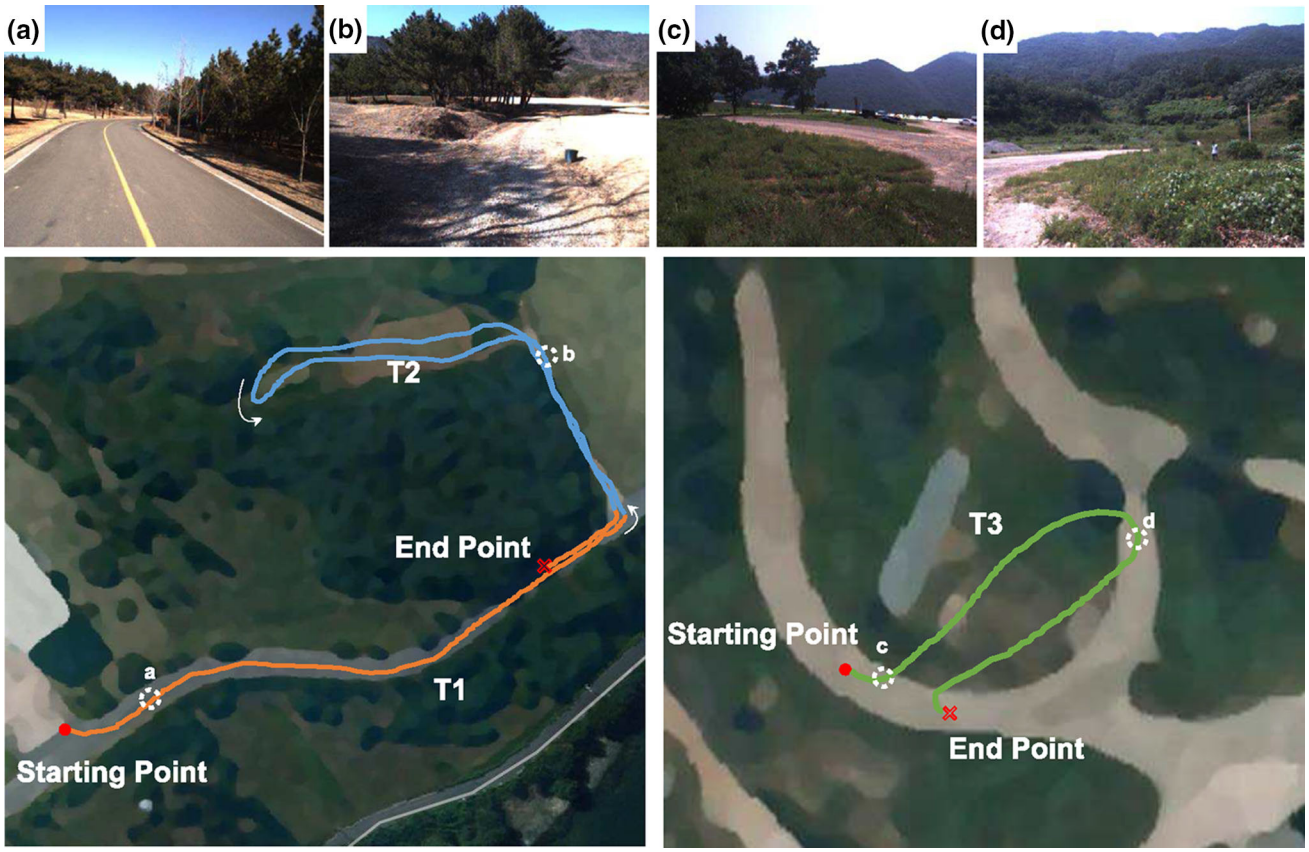

Fig. 3 Example of outdoor environments scene (*top row*) and an overhead map view of the paths taken at an outdoor test facility in Changwon, Korea (*bottom row*); **a** T1 mainly paved road casting shadows by trees, **b** T2 consists of a surface covered with leaves and low grass under 10 cm, **c**, **d** T3 is a challenging scene containing foliage and dense vegetation over 30 cm high

consists of mainly paved road with trees, T2 has a dirt road with short grass and trees and T3 has a dirt road, bushes and complex vegetation.

6.2.1 Experimental setup

Our robot platform is equipped with multiple sensors consisting of a *Velodyne HDL-32* 3D LiDAR and a *PointGrey Flea2* camera. The LiDAR has a 360° horizontal field of view and a 40° vertical field of view, with a 10 Hz frame rate and a 0.16° angular resolution. The cameras have a 60° horizontal field of view, with a frame rate of 30 Hz and a resolution of 640 × 480 pixels. The data from the each sensor is time-stamped to enable synchronization. The extrinsic parameters between the LiDAR and the camera are estimated using the algorithm proposed by Kwak et al. (2011). The current position of the robot is estimated by Kalman filter based on the IMU and GPS data. The bumper system using a switch sensor is utilized to detect hidden obstacles in the vegetation region.

The real dataset used for our experiments are mainly obtained in three outdoor environments: paved roads casting shadows, off-road containing grass and gravel, and complex rough terrain containing foliage and dense vegetation. In order to obtain the dataset in several environment conditions, a person drove the robot platform. Since the driver already knew the road condition and the obstacle positions, the input of the bumper system is artificially generated for collecting the dataset.

Because our superpixels represent minimally meaningful region, it is necessary to empirically find the optimal size of superpixel to maximize the performance. We use a superpixel algorithm from Achanta et al. (2012) called Simple Linear Iterative Clustering (SLIC) which takes two parameters: the size of the superpixel and the strength of the spatial regularization. These two parameters are trade-off between clustering appearance and spatial regularization. In our experiments, we fix these two parameters, regularizer as 0.1 (lower regularization would lose more spatial information), and used the five different superpixel sizes. We originally tested our algorithm with the superpixel step size 5 from 20 to 150. However, since the objective of the experiment is to compare the effect of different superpixel size on clustering performance, we ruled out ambiguous regions that did not stand out as being more important than the rest. Thus we sampled the sizes of superpixels

Table 5 Purity and the number of clusters on a superpixel dataset

Superpixel size	$s = 20$	$s = 30$	$s = 50$	$s = 100$	$s = 150$
Paved road purity	**0.941** (±0.020)	**0.986** (±0.012)	0.931 (±0.018)	0.927 (±0.022)	0.901(±0.025)
Dirt road purity	0.891 (±0.027)	**0.914** (±0.033)	**0.909** (±0.021)	0.887 (±0.031)	0.872 (±0.039)
Yellow short grass purity	0.889 (±0.041)	**0.920** (±0.038)	**0.911** (±0.024)	0.871 (±0.040)	0.866 (±0.043)
Green short grass purity	0.822 (±0.048)	**0.852** (±0.057)	**0.849** (±0.031)	0.810 (±0.041)	0.801 (±0.047)
Green tall grass purity	0.818 (±0.074)	**0.838** (±0.079)	**0.823** (±0.088)	0.807 (±0.077)	0.784 (±0.091)
Number of clusters	6.41 (±0.387)	**5.32** (±0.118)	**5.74** (±0.223)	6.77 (±0.499)	7.22 (±0.581)

Bold values indicate the best performance and the second performance

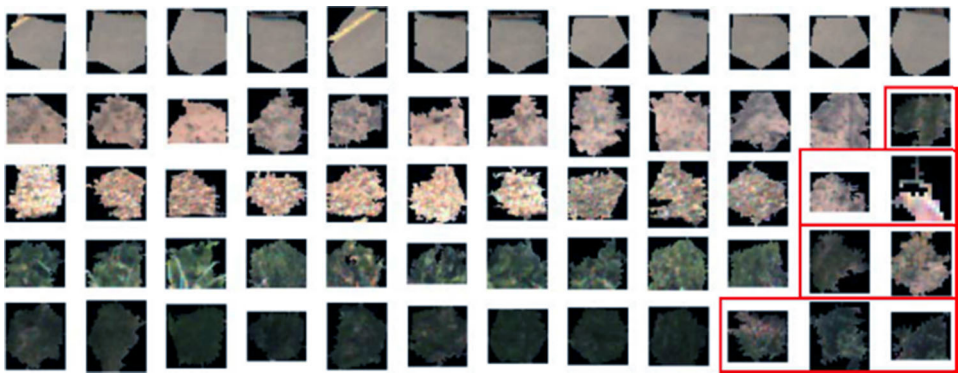

Fig. 4 Clustering result of our traversable region detection algorithm on an unstructured outdoor scene, including five clusters showing in five rows: paved road, dirt road, short yellow grass, short green grass and tall green grass. 12 regions are randomly chosen from each cluster. The number of incorrect regions is highlighted by the *red rectangle* and proportional to the cluster purity (Color figure online)

$s = \{20, 30, 50, 100, 150\}$, which were determined experimentally.

6.2.2 Clustering results

We first investigate the ability of our algorithm to find the number of terrain patterns and the cluster purity on the real datasets. The cluster purity is defined as the ratio of the dominant class label in each cluster. For this purpose, we generate large real datasets which contain 10,000 superpixels with 5 different terrain patterns including, paved road, dirt road, short yellow grass, short green grass and tall green grass, from recorded video sequences. Then, we randomly select 3000 superpixels including all 5 classes and run INBC to measure the clustering accuracy. This test process was carried out 30 times for every randomly selected datasets.

Table 5 shows the average numbers of the clusters and cluster purities of the 5 terrain patterns using our algorithm with different sizes of superpixels. As proved in Table 5 the strongest superpixel size in the clustering is $s = 30$. This indicates that although the smaller superpixel size mostly represents independent objects, it does not have enough pixels to extract the features. In contrast, the larger size of the superpixels detracts from the purity of each cluster. Note that the size of the superpixel generally has a greater effect when it is larger because one superpixel region has the possibility of containing two or more objects.

Figure 4 shows examples of cluster members. The superpixels have slightly different resolutions, illumination conditions and poses. However, obtaining the number of clusters and cluster purity, when the superpixel size is 30, is an encouraging result for the dataset. Because our approach is based on imaging, it has a limitation in acquiring depth information. Due to the limitation of image-based methods, the superpixel size is more important in complex environments for dividing object boundaries.

We also compared the performance of our algorithm with the same state-of-the-art algorithm in Sect. 6.1. Same as the finding the number of terrain patterns and the cluster purity experiments, 3000 superpixels are randomly selected from the large real datasets to test the clustering accuracies by the ARI and computation time. Table 6 describes the clustering results by INBC achieves a higher clustering accuracy than that by state-of-the-art algorithms. It also shows that INBC are faster than other clustering methods except for parametric algorithm k-means. The general computation time is dependent on the size of the datasets using batch-style clustering algorithm. Thus the computation time increases in general as the number of superpixels increase. However, because we build our clustering model as incremental method, it is beneficial in terms of computational cost during the data updates. The computational costs of our method and the state-of-the-art nonparametric clustering methods are compared during data updates in Fig. 5. Whereas the computational cost of the batch style state-of-the-art approaches increases as data are added, the cost of INBC remains low, and the incremental solution of our approach yields similar or better accuracy compared to the batch-style clustering.

Table 6 ARI and computation time

Dataset	k-means	AP	BHC	GBHC-NODE	Ours
$s = 20$	0.739 (±0.069)	0.426 (±0.167)	**0.822** (±0.132)	0.487 (±0.154)	**0.878** (±0.082)
$s = 30$	0.783 (±0.104)	0.483 (±0.122)	**0.880** (±0.107)	0.503 (±0.174)	**0.908** (±0.040)
$s = 50$	**0.748** (±0.127)	0.456 (±0.203)	0.702 (±0.144)	0.418 (±0.159)	**0.897** (±0.069)
$s = 100$	**0.708** (±0.159)	0.427 (±0.213)	0.613 (±0.147)	0.378 (±0.186)	**0.861** (±0.108)
$s = 150$	**0.617** (±0.145)	0.352 (±0.219)	0.607 (±0.195)	0.356 (±0.170)	**0.833** (±0.120)
Mean	0.719	0.429	**0.725**	0.428	**0.875**
Time (s)	**20.5** (±0.289)	86.1 (±18.3)	1230 (±19.8)	264 (±9.20)	**48.3** (±0.498)

Bold values indicate the best performance and the second performance

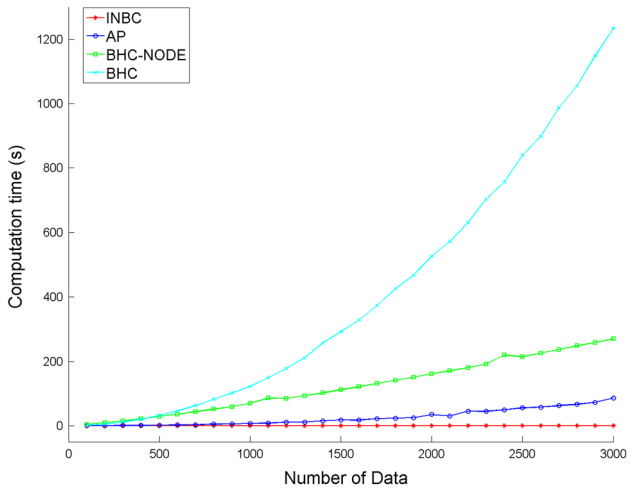

Fig. 5 Computational times (s) of the state-of-the-art nonparametric clustering algorithm and our proposed algorithm. As the number of training data increases, the computation times of AP (Frey and Dueck 2007), BHC (Heller and Ghahramani 2005) and GBHC-NODE (Sirinukunwattana et al. 2013) significantly increase. However, the computation time of INBC remains low

6.3 Traversable region detection

The results in the previous Sects. 6.1 and 6.2 show the clustering capability of the INBC algorithm. In this section, we demonstrate the performance of our traversability estimation method in real applications with respect to the classification and adaptation capability. We qualitatively evaluate the prediction accuracy of our traversability estimation method with a receiver operating characteristic (ROC) curve. To measure the classification rates, we generate ground truth data by hand. Also, we evaluate the adaptation of our method while the environment is changing.

6.3.1 Classification evaluation

To evaluate the capability of the traversability estimation, we utilize 300 frames to test in each environment T1, T2 and T3. All evaluations start when the labeled data are the input to the model and ends after 300 frames pass in real-time. Once the data are fed into the learning system described in the Algorithm 2, it takes an average of 0.02 sec to estimate the result. To measure the classification performance, we manually construct ground truth data with respect to an image from each frame for all 300 frames and use ROC curves and area under curves (AUCs). The ROC curves shows how positive examples are correctly classified as traversable (True Positive Rate) and how the negative examples are misclassified as traversable (False Positive Rate). It represents the quantitative accuracy of the traversability estimation approach, and the AUC represents the accuracy measurement between 0 and 1.

Figure 6 illustrates our inference results for the traversable region estimation in three different environments. The top row of our figure shows examples of input images with different sizes of superpixels, and the bottom row of the figure shows the ROC curves and AUC. In all cases, the size of the superpixels $s = 30$ gives the most accurate performance in classification. As expected, we can observe in Fig. 6 that our algorithm accurately classifies the traversable region at the current view under the various environments, and a larger size of superpixel generally provides more inaccurate results, which is similar to the clustering results.

Comparing the results in three environments, we have the lowest AUC in T1 and the highest AUC in T2. Intuitively, these results are not reasonable, as T1 is a more structured and simpler environment than T2 and T3. However, when we construct the ground truth, we consider all possible traversable regions such as the grass region that is located on the side of the paved road in T1. Because the robot travels only on a paved road until it comes into T2, it cannot identify the grass region as a traversable region. This is why the AUC in T1 is the lowest and does not follow the trend in which the smaller size of superpixel provides more accurate results.

6.3.2 Adaptation evaluation

As mentioned in Sect. 5, our method starts its exploration without any prior knowledge of surroundings. However, the

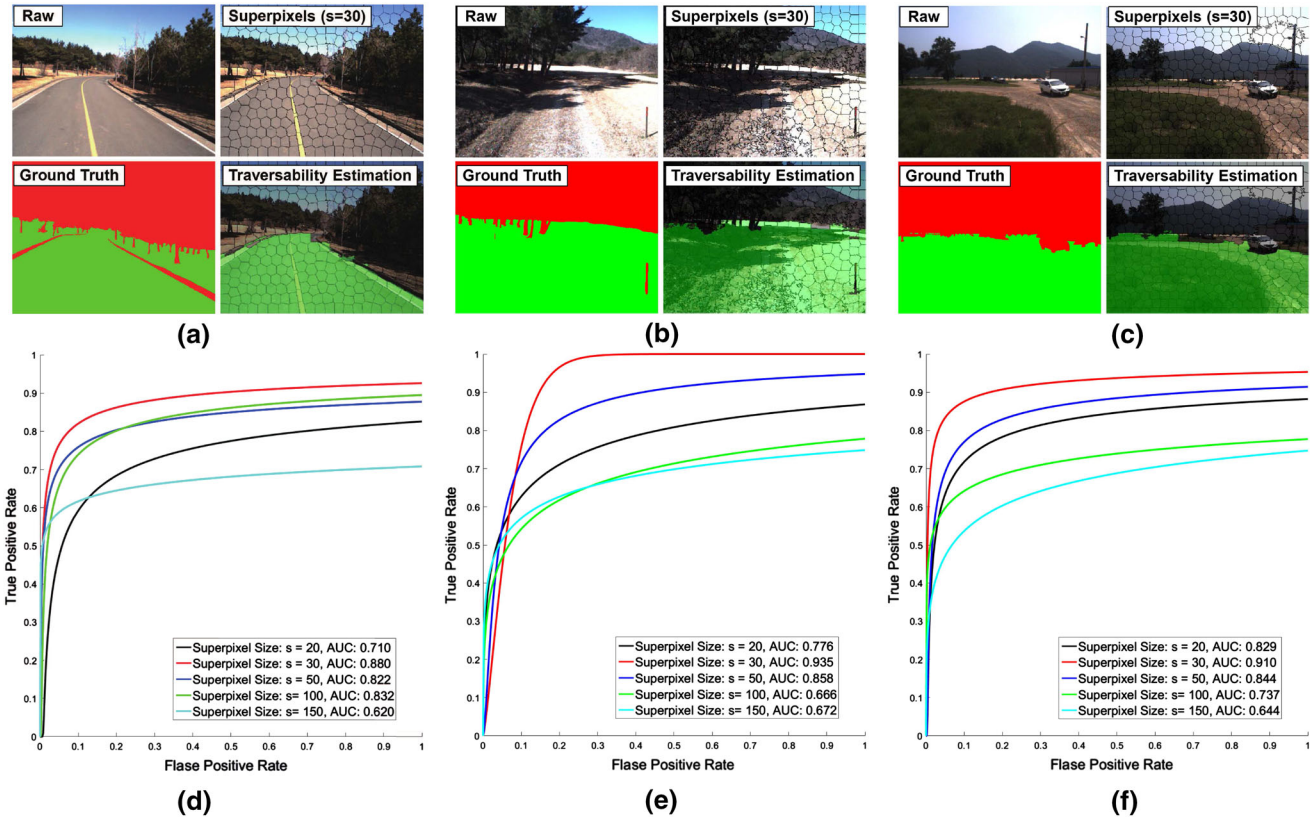

Fig. 6 Qualitative examples of the success of the traversable region classifier in T1, T2 and T3 environments and qualitative results. Original RGB inputs with superpixels, ground truth and estimation of traversability are in **a** T1, **b** T2, **c** T3. ROC curve for each environment evaluated on different superpixel sizes of image in **d**, **e**, **f**

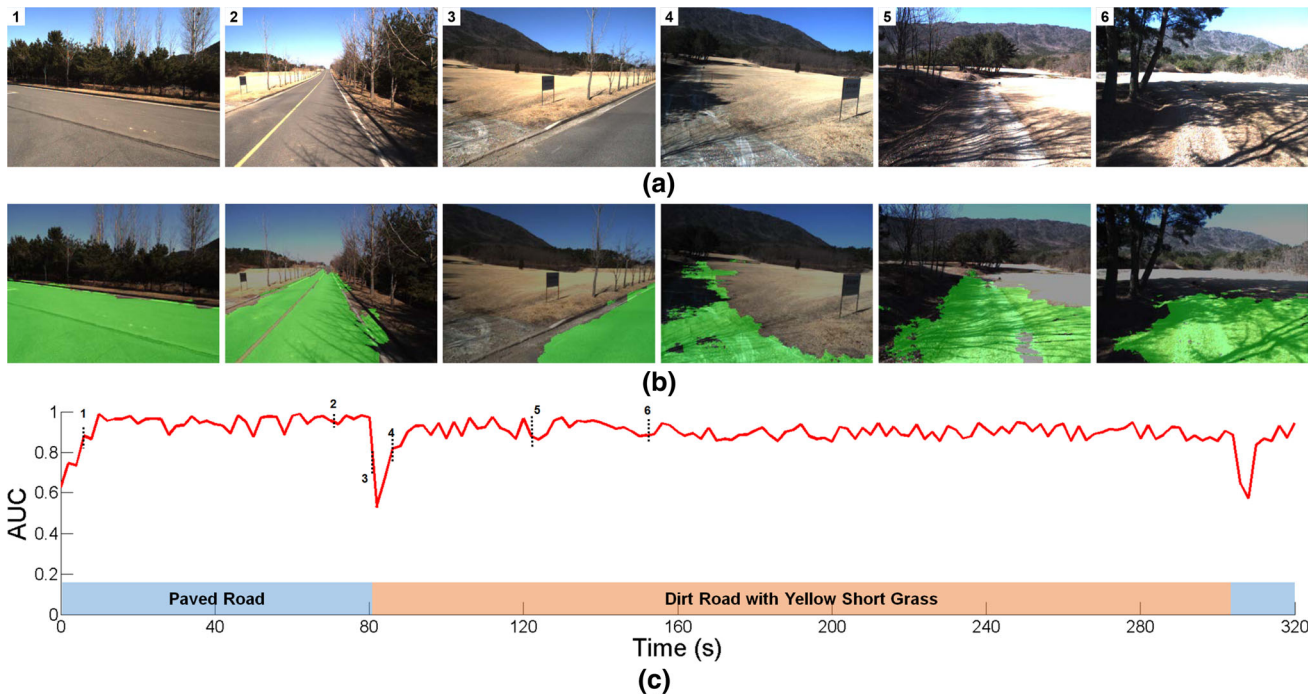

Fig. 7 Incremental Learning Experiment in T1 and T2. **a** Some challenging exemplary original scenes, **b** classifier results (*green* is traversable), **c** AUC changing when the robot is exploring environments T1 and T2. The *small box with color* indicates the specific environment, and the *color changing* indicates the change from one environment to the other (Color figure online)

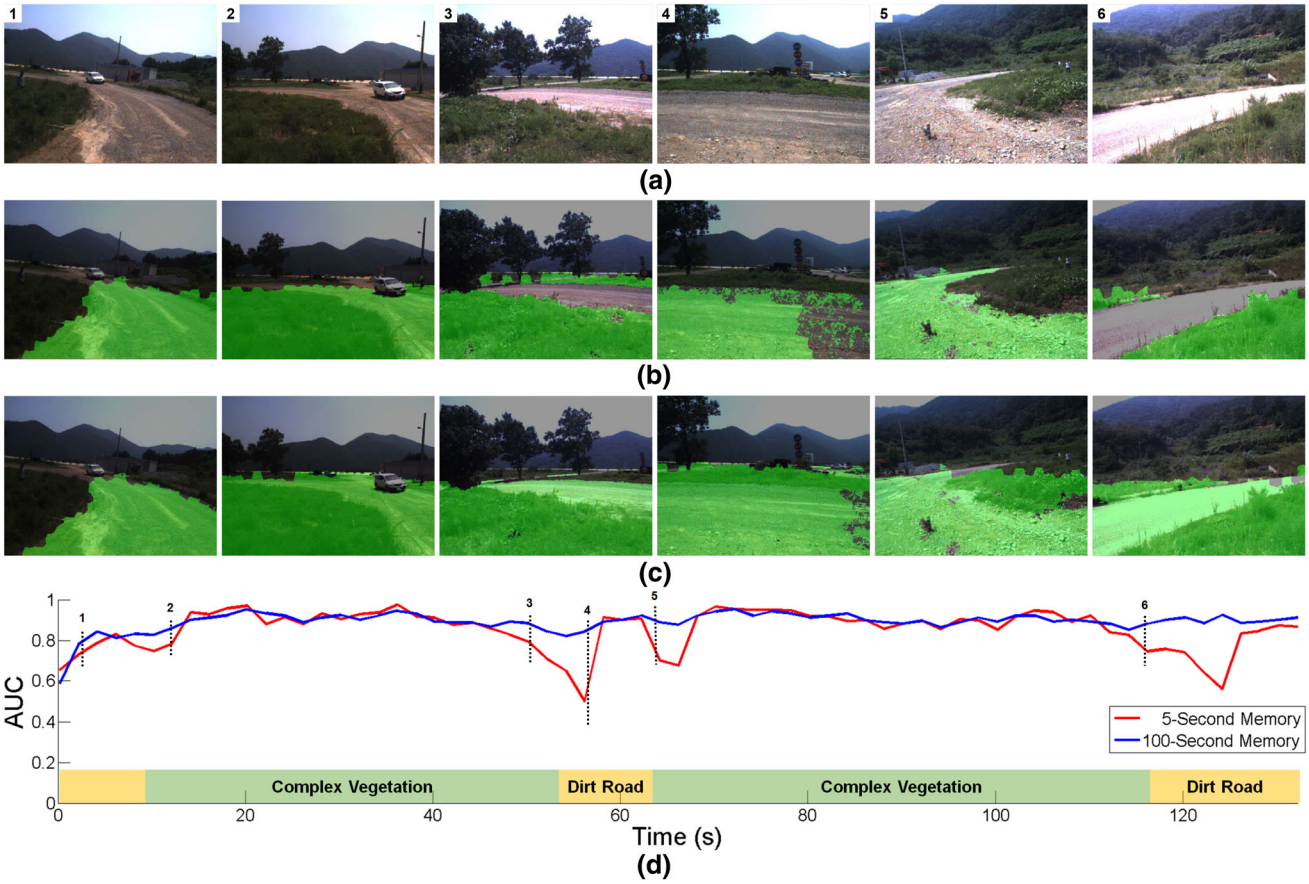

Fig. 8 Incremental Learning Experiment in T3. **a** Some challenging exemplary original scenes, **b** classifier results with 5-s memory (*green* is traversable), **c** classifier results with 100-s memory (*green* is traversable), **d** AUC changing when the robot is exploring environment T3 with different size of memory, which clearly shows the different performance of two memory types. In the beginning, upon entering a novel environment, the 5- and 100-s memories have similar performances. Upon switching back into the already acquired environment, the 100-s memory case experiences no decreasing in AUC performance (Color figure online)

system is capable of learning in a self-supervised fashion through interaction with the environment. Thus, as time progresses, the traversable region information in unknown environments is added to the model. To demonstrate the incremental and on-line learning performance in real outdoor environments without prior knowledge, we first test the robot with short memory (5 s). Using this limitation of memory size, we can observe the performance of incremental learning adaptation speed and accuracy. Second, we enforce the robot memory to 100 s which means that the robot forgets learning knowledge later than 5 s memory. Through this experiment, we observe that the performance of the INBC maintains the learning knowledge while the environments change.

Figure 7 shows the change in value of the area under the ROC curve (AUC) when the robot is driving in two types of environments, starting from T1 then switching to T2. In this experiment, we set the robot memory to 5 s prior to the current time. This means that the robot stores the superpixels with labels for the previous 5 s. There are two environments changing around time 80 and 310 s. The evaluation of the AUC number demonstrates that our method adapts to and trains its model to distinguish traversable and non-traversable areas in fewer than 10 s. we can see that our method can adapt and recover to new environments when its memory is limited.

Although there are some recent similar works regarding traversability estimation (Lu et al. 2015) or road detection (Wang et al. 2015), it is hard to directly compare the results with them, because these approaches are not similar in condition of environment (complex vegetation) and sensor configuration. Lu et al. (2015) achieved approximately 90 % true positive rate using unsupervised learning algorithm with the LAGR datasets (field environment, stereo camera) and Wang et al. (2015) also achieved 90 % true positive rate using weakly supervised learning algorithm with the CamVid and their own datasets (urban environment, mono camera). Nevertheless, during our experiment, we achieve approximately 91 % of the true positive rate, with 8 % of the false positive rate in complex environments. These outcomes are described as an acceptable level of accuracy indirectly compared to Lu et al. (2015) and Wang et al. (2015).

Figure 8 shows the result of comparing the stored 5-s memory and 100-s memory in environment T3. In this experiment, there are four environments, and the environment illumination changes around time 10, 55, 63 and 117. We can see that the performance behavior of the 5-s memory case is very similar to that in Fig. 7c, whereas the performance behavior of the 100-s memory case maintains an AUC level of approximately 0.9. These results demonstrate that our proposed algorithm INBC successfully works in maintaining clustering results even in complex environmental changes with long-term datasets. From these experiments, we show how the proposed method can rapidly train a model from observation (5-s memory case) and affect the size of memory to maintain the classification accuracy (100-s memory case).

7 Conclusion

We have presented an approach that estimates the traversability in complex and unknown outdoor environments. To accomplish this goal, we have also proposed an incremental nonparametric Bayesian clustering (INBC) algorithm to find the proper number of clusters and to converge reliably quickly for incrementally evolving data. As explained, the principal approach in this paper has three main stages. First stage is automatic region labeling, in which the input image is segmented into coherent area based on intensity information. Then, the training samples, previously labeled based on the outcome of the information collected from the traversing vehicle, are obtained. Second stage is incremental model learning. This stage facilitates the labelled data, which are incrementally clustered. These models end up building a traversability. The last stage estimates the traversability of upcoming regions with the learned model.

We demonstrated the performance of the proposed INBC algorithm through intensive experiments using synthetic and real data and evaluated the viability of the traversability estimation using collected real data sets. The experiments showed that the INBC has a better computation cost and clustering accuracy than the existing batch style approaches. The experiments of the traversability estimation have shown that our approach works robustly and effectively in unknown and complex environments.

Acknowledgments This research was supported by the MOTIE (The Ministry of Trade, Industry and Energy), Korea, under the Technology Innovation Program supervised by KEIT (Korea Evaluation Institute of Industrial Technology),10045252, Development of robot task intelligence technology.

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Susstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(11), 2274–2282.

Aherne, F. J., Thacker, N. A., & Rockett, P. I. (1998). The Bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetika*, *34*(4), 363–368.

Angelova, A., Matthies, L., Helmick, D., & Perona, P. (2007). Learning slip behavior using automatic mechanical supervision. In *IEEE international conference on robotics and automation, 2007* (pp. 1741–1748). IEEE.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *The Journal of Machine Learning Research*, *3*, 993–1022.

Blei, D. M., Jordan, M. I., et al. (2006). Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, *1*(1), 121–143.

Bradley, D. M., Unnikrishnan, R., & Bagnell, J. (2007). Vegetation detection for driving in complex environments. In *2007 IEEE international conference on robotics and automation* (pp. 503–508). IEEE.

Comaniciu, D., & Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(5), 603–619.

Comaniciu, D., Ramesh, V., & Meer, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(5), 564–575.

Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. *Science*, *315*(5814), 972–976.

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, *32*(11), 1231–1237.

Häselich, M., Arends, M., Wojke, N., Neuhaus, F., & Paulus, D. (2013). Probabilistic terrain classification in unstructured environments. *Robotics and Autonomous Systems*, *61*(10), 1051–1059.

Heller, K. A., & Ghahramani, Z. (2005). Bayesian hierarchical clustering. In *Proceedings of the 22nd international conference on machine learning* (pp. 297–304). New York: ACM.

Ho, K., Peynot, T., & Sukkarieh, S. (2014). Analyzing the impact of learning inputs on near-to-far terrain traversability estimation. In *Proceedings of 2014 IEEE international conference on robotics and automation (ICRA 2014)*.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*(1), 193–218.

Kelly, A., Stentz, A., Amidi, O., Bode, M., Bradley, D., Diaz-Calderon, A., et al. (2006). Toward reliable off road autonomous vehicles operating in challenging environments. *The International Journal of Robotics Research*, *25*(5–6), 449–483.

Kim, D., Sun, J., Oh, S. M., Rehg, J. M., & Bobick, A. F. (2006). Traversability classification using unsupervised on-line visual learning for outdoor robot navigation. In *Proceedings of 2006 IEEE international conference on robotics and automation (ICRA 2006)* (pp. 518–525). IEEE.

Kjærgaard, M., Bayramoglu, E., Massaro, A. S., & Jensen, K. (2011). Terrain mapping and obstacle detection using Gaussian processes. In *2011 10th international conference on machine learning and applications and workshops (ICMLA)* (Vol. 1, pp. 118–123). IEEE.

Kwak, K., Huber, D. F., Badino, H., & Kanade, T. (2011). Extrinsic calibration of a single line scanning lidar and a camera. In *2011 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3283–3289). IEEE.

Lalonde, J. F., Vandapel, N., Huber, D. F., & Hebert, M. (2006). Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of Field Robotics*, *23*(10), 839–862.

Lu, H., Jiang, L., & Zell, A. (2015). Long range traversable region detection based on superpixels clustering for mobile robots. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 546–552). IEEE.

MacQueen, J., et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Oakland, CA, USA* (Vol. 1, pp. 281–297).

Munoz, D., Vandapel, N., & Hebert, M. (2009). Onboard contextual classification of 3-d point clouds with learned high-order Markov random fields. In *Proceedings of 2009 IEEE international conference on robotics and automation (ICRA 2009)*. IEEE.

Ng, A. Y., Jordan, M. I., Weiss, Y., et al. (2002). On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, *2*, 849–856.

Ojala, T., Pietikäinen, M., & Mäenpää, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(7), 971–987.

Ott, L., & Ramos, F. (2012). Unsupervised incremental learning for long-term autonomy. In *Proceedings of 2012 IEEE international conference on robotics and automation (ICRA 2012)* (pp. 4022–4029). IEEE.

Santamaria-Navarro, À., Teniente, E. H., Morta, M., & Andrade-Cetto, J. (2015). Terrain classification in complex three-dimensional outdoor environments. *Journal of Field Robotics*, *32*(1), 42–60.

Shneier, M., Chang, T., Hong, T., Shackleford, W., Bostelman, R., & Albus, J. S. (2008). Learning traversability models for autonomous mobile vehicles. *Autonomous Robots*, *24*(1), 69–86.

Silver, D., Bagnell, J.A., & Stentz, A. (2012). Active learning from demonstration for robust autonomous navigation. In *Proceedings of 2012 IEEE international conference on robotics and automation (ICRA 2012)* (pp. 200–207). IEEE.

Sirinukunwattana, K., Savage, R. S., Bari, M. F., Snead, D. R., & Rajpoot, N. M. (2013). Bayesian hierarchical clustering for studying cancer gene expression data with unknown statistics. *PLoS One*, *8*(75), 748.

Sofman, B., Lin, E., Bagnell, J. A., Cole, J., Vandapel, N., & Stentz, A. (2006). Improving robot navigation through self-supervised online learning. *Journal of Field Robotics*, *23*(11–12), 1059–1075.

Stavens, D., & Thrun, S. (2006). A self-supervised terrain roughness estimator for off-road autonomous driving. In *In proceedings of the conference on uncertainty in AI (UAI)*, Citeseer.

Sun, J., Mehta, T., Wooden, D., Powers, M., Rehg, J., Balch, T., et al. (2006). Learning from examples in unstructured, outdoor environments. *Journal of Field Robotics*, *23*(11–12), 1019–1036.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., et al. (2007). Stanley: The robot that won the darpa grand challenge. In *The 2005 DARPA grand challenge* (pp. 1–43). Berlin: Springer.

Trautmann, E., & Ray, L. (2011). Mobility characterization for autonomous mobile robots using machine learning. *Autonomous Robots*, *30*(4), 369–383.

Tse, R., Ahmed, N. R., & Campbell, M. (2015). Unified terrain mapping model with Markov random fields. *IEEE Transactions on Robotics*, *31*(2), 290–306.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., et al. (2008). Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, *25*(8), 425–466.

Wang, Q., Fang, J., & Yuan, Y. (2015). Adaptive road detection via context-aware label transfer. *Neurocomputing*, *158*(C), 174–183.

Wellington, C., Courville, A., & Stentz, A. T. (2006). A generative model of terrain for autonomous navigation in vegetation. *The International Journal of Robotics Research*, *25*(12), 1287–1304.

Zhou, S., Xi, J., McDaniel, M. W., Nishihata, T., Salesses, P., & Iagnemma, K. (2012). Self-supervised learning to visually detect terrain surfaces for autonomous robots operating in forested terrain. *Journal of Field Robotics*, *29*(2), 277–297.

Honggu Lee received B.S. degree in Mechanical Engineering from Sungkyunkwan University, Suwon, Korea, in 2010. He received his M.S. degree in 2012 and is currently a Ph.D. candidate in the School of Computing from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea. His current research interests include nonparametric method, machine learning and computer vision for autonomous vehicles.

Kiho Kwak is a senior researcher in Agency for Defense Development, South Korea. He received his B.S. and M.S. degrees from the Korea University in 1999 and 2001 respectively, and Ph.D. in ECE from the Carnegie Mellon University (CMU) in 2012. His research interests include sensor fusion, online object modeling and perception and navigation for autonomous vehicles in outdoor environment.

Sungho Jo received the B.S. degree in School of Mechanical and Aerospace Engineering from the Seoul National University, Seoul, Korea, in 1999, the S.M. in mechanical engineering, and Ph.D. in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 1999 and 2006 respectively. While pursuing the Ph.D., he was associated with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Laboratory for Information Decision and Systems (LIDS), and Harvard-MIT HST NeuroEngineering Collaborative. Before joining the faculty at KAIST, he worked as a postdoctoral researcher at MIT media laboratory. Since December in 2007, he has been with the department of computer science at KAIST, where he is currently (tenured) Associate Professor. His research interests include intelligent robots, neural interfacing computing, neuromorphic computing, and wearable computing etc.