Remote vehicle operator must quickly decide on the motion and path. Thus, rapid and intuitive feedback of the real environment is vital for effective control. This paper presents a real-time traversable ground surface segmentation and intuitive representation system for remote operation of mobile robot. Firstly, a terrain model using voxel-based flag map is proposed for incrementally registering large-scale point clouds in real time. Subsequently, a ground segmentation method with Gibbs-Markov random field (Gibbs-MRF) model is applied to detect ground data in the reconstructed terrain. Finally, we generate a texture mesh for ground surface representation by mapping the triangles in the terrain mesh onto the captured video images. To speed up the computation, we program a graphics processing unit (GPU) to implement the proposed system for large-scaledatasets in parallel. Our proposed methods were tested in an outdoor environment. The results show that ground data is segmented effectively and the ground surface is represented intuitively.

1. Introduction

Mobile robots must be able to navigate and interact with unknown environments, without collisions or encountering other dangers, by determining traversable terrain regions, reconstructing terrain models, and employing other technologies. The technologies of dynamic terrain reconstruction and modeling with multiple sensors have been researched to provide mobile robots with the ability to conduct free space detection and support navigation without collision [1].

Datasets received by multiple sensors are integrated to produce accurate and reliable terrain information, such as 3D point cloud, video image, GPS data, and rotation state [2, 3]. It is necessary to integrate instantaneously received datasets into terrain model, such as voxel map and textured mesh. The conventional terrain modeling methods are difficult to process large-scale datasets, which are so large that they exceed the memory capacity of mobile robots. Meanwhile, the huge computational cost of the large-scale datasets processing causes a low speed of terrain modeling and visualization.

In remote operation applications, ground segmentation is necessary in the assessment of traversable regions. To accomplish this objective, we apply a ground segmentation method using the Gibbs-Markov random field (MRF) method. To create a photorealistic visualization for traversable ground surface, textured terrain models are represented using sensed datasets from each frame. By mapping captured video images on the 3D ground surface mesh, the visualization system provides intuitive imagery of a 3D geometric model for easy
terrain perception. Conventionally, the captured images are registered into the terrain model frame by frame. However, when mobile robots navigate a large-scale environment, the sensed images are registered incrementally. The amount of data becomes so large that it exceeds the robots’ memory capacity.

For the purpose of improving traversable ground surface reconstruction to get rapid and intuitive representation, real-time terrain modeling and photorealistic visualization systems have been developed [4, 5]. Photorealistic visualization attempts to realistically represent the realistic 3D terrain and objects model in virtual world, as these appear in the real world [6, 7]. This type of visualization provides perceptive geospatial information for path planning and decision-making. In order to improve the speed performance, the real-time terrain visualization methods are researched [8, 9]. A level-of-detail (LOD) method is commonly used to render the near-field regions of the terrain model in real time. In far-field regions, billboard rendering methods, which represent a texture in front of the terrain model, are commonly applied to realize photorealistic visualization. Considering a large-scale terrain model, the sequential computation of data registration requires substantial computational power.

In this paper, we propose a real-time traversable ground surface segmentation and intuitive representation system. To speed up the computation of the proposed methods, we used graphics processing unit (GPU) programming to implement the segmentation and modeling processes for large-scale datasets in parallel [10].

To reconstruct an intuitive terrain model using limited memory in real time, we firstly create a voxel-based flag map in GPU memory to register point clouds without redundancy. Then, we create a node-based texture mesh for ground surface representation. Each node in the mesh contains a certain quantity of vertices and a node texture. The node texture is generated by mapping the triangles in the captured image, which are projected from the vertices of the node mesh. Subsequently, we implement Gibbs-MRF method to segment traversable ground surface in the flag map and create a ground mesh from the segmentation results. Next, we map the triangles in a mesh node onto the captured images in order to generate the node texture. Finally, we represent the reconstructed terrain model by overlaying the ground mesh with the node textures.

This paper is organized as follows. In Section 2, we discuss related works on large-scale terrain reconstruction and traversable ground segmentation. In Section 3, we describe a real-time traversable ground surface segmentation and intuitive representation system. In Section 4, we analyze the results of the proposed segmentation and modeling methods. In Section 5, we present our conclusions.

2. Related Works

During navigation and interactive tasks, rapid feedback on intuitive representations of a robot’s surrounding terrain is required for real-time remote operation. To provide a remote operator with representations of a robot’s surrounding terrain, robots need to reconstruct a terrain model using multiple sensors [11]. Sukumar et al. [12] propose an unmanned terrain modeling method using a multisensor system. This method integrates sensed datasets from each frame into a textured terrain mesh and provides a convenient visualization. However, when robots explore outdoor environments for long periods of time, this system cannot process large-scale datasets.

To prevent the terrain model from being too large for the physical memory capacity, the terrain storage system requires highly compressed terrain model and fast memory loading performance [13]. Chang et al. [14] proposed a panorama image generation method, which combines video sequence into a panoramic scene. By this method, overlapping regions of video sequence were removed so that video storage was effectively compressed. Gingras et al. [15] reconstructed an unstructured surface from a 360° point cloud scan and represented the traversable areas using a compressed irregular triangular mesh. They applied a mesh simplification algorithm to reduce the triangles quantity on the large-scale terrain surface. Zhuang et al. [16] proposed an edge-feature-based ICP algorithm to extract edge points, which were registered into a 3D roadmap integrated with planar features and elevation information. By using this method, a large number of redundant points of pseudoedge were removed so that large-scale point clouds registration was implemented fast. However, the visualization of these works is not intuitive enough for remote operation. It is necessary to overlay 3D terrain model with captured video image.

Other researchers have enhanced the performance of photorealistic visualization to provide real-time terrain modeling with low CPU consumption. Huber et al. [8] and Kelly et al. [17] describe methods for real-world representation using video-ranging modules. Scenes are rendered in the near field using a 3D color mesh, whereas far-field scenes are rendered using a billboard texture of the scene in front of robots. However, they allocated one color per grid, which caused distortion.

Meanwhile, ground segmentation is an important task that determines the traversability of a terrain. To segment ground data in 2D image or 3D terrain model, we need to calculate each pixel or voxel’s probability of being in the ground and nonground configurations [18, 19]. Vernaza et al. [20] presented a prediction-based structured terrain classification method for the DARPA Grand Challenge. They used an MRF model to classify the pixels in 2D images into obstacles or ground regions. Because the computation of MRFs is too complicated for large-scale datasets, we need to remove redundant elements from the MRF in order to reduce the computational cost of ground segmentation.

3. Traversable Ground Surface Reconstruction

In this section, a real-time traversable ground surface segmentation and intuitive representation system is presented. Firstly, a GPU-based ground surface reconstruction system is described. Then, a voxel-based flag map is proposed for
generating a nonreduplication terrain model. Next, a Gibbs-MRF model is applied to detect ground surface in the reconstructed terrain. Finally, we create a node-based texture mesh for ground surface representation.

3.1. GPU-Based Ground Surface Reconstruction System. In a large-scale environment, there are a large number of triangles of several nodes to be mapped from the captured video images. Hence, to realize real-time traversable ground surface segmentation and reconstruction, we implement the proposed methods in parallel by applying GPU programming. The framework of the GPU-based terrain modeling system is shown in Figure 1.

3.2. Voxel-Based Flag Map. After the vehicle collects several consecutive frames of 3D point clouds, some points are inserted into the same voxel. This causes wasteful duplication of memory if we register these points into the terrain model. To remove redundant points, we developed a voxel-based flag map to register 3D points into the terrain model without reduplication.

The sensed point clouds are quantized into a space of regular voxels. We specify a bitstream to define a voxel-based flag map. We allocate a 1-bit variable $b(v)$ for each voxel $v$. We initialize the flag map by specifying $b(v) = 0$. When at least one point exists in the covered area of the voxel $v$, $b(v) = 1$; otherwise, $b(v) = 0$. We register voxels when they are sensed for the first time. Hence, when a new point is converted to a voxel $v$ and $b(v) = 1$, the vehicle does not register this point in the terrain model.

For long-term navigation in an environment, the range of the sensed points exceeds the defined range of the flag map. To solve this problem, we shift the center of the flag map to the position of the vehicle when the vehicle moves to a certain distance. It is necessary to drop the passing information from the memory of the flag map and store new sensed points. In this manner, we utilize a flag map with limited range to represent the information about the dynamic environment surrounding the vehicle.

3.3. Ground Segmentation Based on Gibbs-MRF Model. To segment ground data, we need to calculate each voxel’s probability for ground or nonground configuration. The voxels in the voxel-based flag maps are highly affected by their neighboring voxels. This phenomenon follows the property of Markov chains; therefore, an MRF model is widely used in segmentation. In this section, we introduce a ground segmentation method using Gibbs-MRF.

3.3.1. MRF and Gibbs Distribution. We define $S$ as a set of voxels. The random vector $X = \{X_v\}$ on $S$ has a vector consisting of a color observation variable and a configuration variable. The configuration variable has a value for the ground, objects, and the background.
A neighborhood system for \( v \) contains all voxels within a distance of \( r \) \((r \geq 0)\) from \( v \), defined as \( N = \{N_v \mid v \in S\} \), where \( N_v \) is the neighbor set for voxel \( v \). We define a clique as a voxel set neighboring the given voxel. In our application, a clique contains the given voxel and its neighboring voxels. A clique set \( C \) is defined as a collection of single-voxel and pair-voxel cliques, as follows:

\[
C = C_1 \cup C_2,
\]

where \( C_1 = \{v\} \) for \( v \in S \) and \( C_2 = \{\{v, v'\} \mid v', v \in N_v, v \in S\} \).

We define the observation variable \( d \) as a joint value computed based on the height value and color information of a voxel. The color information of a voxel is mapped from the voxel center position to the captured image. Variable \( f \) only represents the configuration. Given observation \( d \) and configuration \( f \), the segmentation process for an image aims to find the best possible configuration \( f^* \) for voxel \( v \), which gives the following optimal solution:

\[
f^* = \arg \max_f p(X_v = f \mid X_1 = d, \forall t \neq v).
\]

(2)

Based on its MRF property, it can be said that the configuration at location \( v \) only depends on the configuration of its neighboring voxels. The random field \( X \) on \( S \) has the joint probability density function (PDF), as follows:

\[
p(f \mid d) = p(X_v = f \mid X_1 = d, \forall t \neq v) = p(X_v = f \mid X_1 = d, \forall t \in N_v).
\]

(3)

However, it is difficult to practically specify the joint probability density \( p(f \mid d) \). To evaluate \( p(f \mid d) \), we apply the Gibbs distribution, following the Hammersley-Clifford theorem.

We use a potential function \( V_c(f) \) to evaluate the impact of the neighboring voxels in clique \( c \), \( c \in C \). The energy function \( U(f) \) in (4) is defined as the sum of the impacts of clique set \( C \):

\[
U(f) = \sum_{c \in C} V_c(f).
\]

(4)

The PDF is calculated using the Gibbs distribution form, as follows:

\[
p(f) = Z^{-1} e^{-U(f)/T},
\]

\[
Z = \sum_f e^{-U(f)/T}.
\]

(5)

Constant \( T \) is referred to as the temperature factor in Gibbs’ theory, and it controls the deviation of the distribution of \( p(f) \) in MRF. The posterior probability of \( f \) under the Bayesian rule is as follows:

\[
p(f \mid d) = \frac{p(d \mid f)p(f)}{p(d)}.
\]

(6)

The solution of (2) is as follows:

\[
f^* = \arg \max_f p(f \mid d) = \arg \max_f p(d \mid f)p(f).
\]

(7)

We assume that the probability functions \( p(d \mid f) \) and \( p(d) \) are in the form of a Gibbs distribution, as expressed in (5). Thus, the process for determining the maximum value of \( p(f) \) is the same as that for determining the minimum value of the energy function \( U(f) \). Therefore, the solution to (7) is equivalently obtained by minimizing the energy functions, as follows:

\[
f^* = \arg \min_f U(f \mid d) = \arg \min_f [U(d \mid f) + U(f)].
\]

(8)

In this application, we consider the impact of the neighboring voxels in single-voxel and pair-voxel potential cliques. The energy function of \( U(d \mid f) + U(f) \) is expressed as follows:

\[
U(d \mid f) + U(f) = \sum_{v \in C_1} V_1(f_v) + \sum_{\{v, v'\} \in C_2} V_2(f_v, f_{v'}),
\]

\[
+ \sum_{v \in C_1} V_1(d_v \mid f_v),
\]

\[
+ \sum_{\{v, v'\} \in C_2} V_2(d_v, d_{v'} \mid f_v, f_{v'}).
\]

(9)

The values of the clique potential functions \( V_1(f_v) \) and \( V_1(d_v \mid f_v) \) depend on the local configuration and observations of clique \( C_1 \). The clique potential functions \( V_2(f_v, f_{v'}) \) and \( V_2(d_v, d_{v'} \mid f_v, f_{v'}) \) reflect the pair-voxel consistency of clique \( C_2 \).

3.3.2. Ground Segmentation in the Global Voxel Map. In the 3D segmentation process, we initially segment the 3D ground data using the robot vehicle’s height \( h \) as the standard. We assume that if the sensed 3D point is below the vehicle height value, ranging from \(-h - \Delta \) to \(-h + \Delta \), then this point is the ground data. In our experiment, \( \Delta = 0.05 \) m. The rough segmentation results produce dataset \( G_1 \). This dataset does not contain all the ground data, because we only determine the configuration using a local height value. To refine the segmentation result computed by the height histogram method, we apply the Gibbs-MRF model to ground segmentation in the global voxel map, which contains only few errors.

The sum of the clique potential functions \( V_1(d_v \mid f_v) \) and \( V_1(f_v) \) is formulated as follows. Constant \( \alpha \) in (10) is a positive numerical value:

\[
V_1(f_v) + V_1(d_v \mid f_v)
\]

\[
= \begin{cases} 
-\alpha & \text{if } (v \notin G_1, f_v = \text{ground}) \\
& \text{or } (v \notin G_1, f_v = \text{nonground}) \\
+\alpha & \text{if } (v \notin G_1, f_v = \text{ground}) \\
& \text{or } (v \in G_1, f_v = \text{nonground}). 
\end{cases}
\]

(10)
3.4. Texture Mesh Generation for Ground Surface. In the voxel map visualization results, we find that there are some holes that exist between adjacent voxels. To solve this problem, we apply a texture terrain mesh so as to produce a photorealistic visualization with intuitive and sufficient information. A terrain mesh is a kind of 2.5D elevation map which represents an x-z cell with the top point in this cell. The texture terrain mesh is generated by mapping the texture onto the terrain mesh.

In this section, we describe a node-based texture mesh which provides an intuitive representation of the reconstructed traversable ground surface. The mesh is generated using several nodes. Each node has a certain quantity of vertices and a texture. The height value of each vertex in the mesh is updated with the top voxel of an x-z cell in the global voxel map. If a new voxel is to be inserted into the mesh but is outside the existing nodes, we create a new node to register this voxel.

Figure 3 shows the process of node texture generation. For each 3D triangle \((T_0, T_1, T_2)\) in the mesh of that node, we create a triangle \((A_0, A_1, A_2)\) in a node texture, which has a set of triangle pixels. Subsequently, a triangle \((B_0, B_1, B_2)\) in a captured image is projected from \((T_0, T_1, T_2)\). We then duplicate triangle \((A_0, A_1, A_2)\) from triangle \((B_0, B_1, B_2)\). After all of the triangles in a node mesh are mapped from the current image, the node texture is updated.

In large-scale terrain environment, there are more than a thousand of such triangles sensed in the terrain mesh. Therefore, it is difficult to apply CPU programming to implement such node texture generation method in real time. Therefore we realize the triangle duplication method in parallel by applying GPU programming, which is implemented by CUDA kernel.

We firstly copy the current captured images, an updated node mesh, and its texture into GPU memory. Next, each triangle of node mesh in GPU is projected onto its destination triangle in the captured image. Then, we duplicate the destination triangle to the triangle in node texture. Finally,
the generated node texture in GPU memory is copied into CPU memory (see Pseudocode 1).

4. Experiments

Experiments were performed to test the proposed traversable ground surface segmentation and modeling method. The mobile robot shown in Figure 4 was used to gather data using its integrated sensors, including a LiDAR sensor, a GPS receiver, a gyroscope detector, and three video cameras. The valid data range of the LiDAR sensor was approximately 70 m from the robot. The multiple sensors collect terrain information in the form of 3D point clouds, 2D images, GPS data, and rotation states. The proposed algorithms were implemented on a laptop with a 2.82 GHz Intel Core2 Quad CPU, a GeForce GTX 275 graphics card, and 4 GB RAM. The terrain model is reconstructed and represented using Microsoft’s DirectX application programming interface.

Figures 5(a) and 5(b) show the result of terrain reconstruction by registering the sensed 3D point clouds and captured video images into the texture terrain mesh. The bottom images were captured by the three cameras. The robot captured one image of 659 × 493 pixels every 0.1 s. The node size was 12.8 × 12.8 m². The resolution of the node texture was 512 × 512 pixels.

We compared the texture buffer sizes of the node textures and the video images, as shown in Figure 6. After 20 s, 200 images were captured and 82 nodes were registered in the terrain model. The node textures buffer size of these nodes was 82.0 Mbit, generated from 247.8 Mbit of video images. Therefore, the results demonstrate that large-scale video images were registered to the node textures with a low memory overhead using the proposed GPU-based ground surface mesh generation method.

We compare the GPU-based rendering speed with CPU-based terrain modeling results, as shown in Figure 7. After the voxels were registered in the terrain model, we segmented the points into the ground dataset and nonground dataset using the Gibbs-MRF method. In our project, we implemented the ground segmentation procedure once for 180 packets of registered voxels. The ground segmentation duration was
0.5271 ms on average, which is much faster than 0.1 s and satisfies the real-time requirement.

We compute the rendering frame counts every second. After 5 seconds, 21,134 triangles were generated in the ground mesh. The rendering speed is approximately 38 fps (frames per second) using GPU programming. After 35 seconds, 105,723 triangles were generated. The rendering speed slows to 26 fps using GPU programming. Using the CPU, the terrain rendering speed is 6 fps after 30 seconds of data collection. In our study, we aimed to render more than 15 fps to achieve real-time visualization. Therefore, the results demonstrate that we meet the real-time requirements for dataset registration and visualization using GPU programming.

5. Conclusions
In this study, we developed an intuitive terrain modeling technique with a traversable ground surface segmentation method for a mobile robot. The mobile robot collects 3D point clouds, 2D images, GPS, and rotation states through multiple sensors. We constructed a voxel-based flag map in order to register the sensed 3D point clouds into the terrain model without redundancy. We segment ground data from the reconstructed voxel map with few errors using a Gibbs-MRF model. The ground voxels are registered into a node-based texture mesh to provide rapid and intuitive information of the visualized terrain with low memory. In order to realize real-time approach, we employed a GPU to implement the proposed methods in parallel. We tested our proposed system using a mobile robot mounted with integrated sensors. The results demonstrated an intuitive visualization performance and low memory requirement in a large-scale environment.

However, in large-scale environment, the nonground objects have complex shapes, such as persons, trees, and vehicles. In this paper, we represent these objects using point-based rendering method and the visualization results are not intuitive enough. In the future, we will research feature-based object classification algorithms to group the sensed objects into different types. Then, we will represent different objects using modeling method.

Conflict of Interests
The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment
This work was supported by the Agency for Defense Development, Republic of Korea.

References
[1] Y. Matsushita and J. Miura, “On-line road boundary modeling with multiple sensory features, flexible road model, and particle filter,” Robotics and Autonomous Systems, vol. 59, no. 5, pp. 274–284, 2011.
[2] V. Sequeira, K. Ng, E. Wolfart, J. G. M. Gonçalves, and D. Hogg, “Automated reconstruction of 3D models from real environments,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 54, no. 1, pp. 1–22, 1999.
[3] A. Nüchter and J. Hertzberg, “Towards semantic maps for mobile robots,” Robotics and Autonomous Systems, vol. 56, no. 11, pp. 915–926, 2008.
[4] A. Agrawal, R. C. Joshi, and M. Radhakrishna, “Real-time photorealistic visualisation of large-scale multiresolution terrain models,” Defence Science Journal, vol. 57, no. 1, pp. 149–162, 2007.
[5] S. Kim, S. Lee, S. Kim et al., “Object tracking of mobile robot using moving color and shape information for the aged walking,” International Journal of Advanced Science and Technology, vol. 3, pp. 59–68, 2009.
[6] A. Dey, A. Cunningham, and C. Sandor, “Evaluating depth perception of photorealistic mixed reality visualizations for occluded objects in outdoor environments,” in Proceedings of the IEEE Symposium on 3D User Interfaces (3DUIT ’10), pp. 127–128, March 2010.
[7] Y. Ho, “Challenging technical issues of 3D video processing,” Journal of Convergence, vol. 4, no. 1, pp. 1–6, 2013.
[8] D. Huber, H. Herman, A. Kelly et al., “Real-time photorealistic visualization of 3D environments for enhanced remote operation of vehicles,” in Proceedings of the International Conference on 3-D Digital Imaging and Modeling (3DIM ’09), pp. 1518–1525, October 2009.
[9] Y. Zhao, X. Cui, and Y. Cheng, “High-performance and real-time volume rendering in CUDA,” in Proceedings of the 2nd International Conference on Biomedical Engineering and Informatics (BMEI ’09), pp. 1–4, October 2009.
[10] C. Gong, J. Liu, H. Chen et al., “Accelerating the sweep 3D for a graphic processor unit,” Journal of Information Processing Systems, vol. 7, no. 1, pp. 63–74, 2011.
[11] C. Frueh, S. Jain, and A. Zakhor, “Data processing algorithms for generating textured 3D building facade meshes from laser scans and camera images,” International Journal of Computer Vision, vol. 61, no. 2, pp. 159–184, 2005.
[12] S. R. Sukumar, S. Yu, D. L. Page, A. F. Koschan, and M. A. Abidi, “Multi-sensor integration for unmanned terrain modeling,” in
[13] S. Quan, J. Xu, and Y. Kim, “Utility-based rate allocation scheme for mobile video streaming over femtocell networks,” *Journal of Information Processing Systems*, vol. 5, no. 3, pp. 151–158, 2009.

[14] S. Chang, H. Chang, S. Yen et al., “Panoramic human structure maintenance based on invariant features of video frames,” *Human-Centric Computing and Information Sciences*, vol. 3, no. 14, pp. 1–18, 2013.

[15] D. Gingras, T. Lamarche, É. Dupuis, and J.-L. Bedwani, “Rough terrain reconstruction for rover motion planning,” in *Proceedings of the 7th Canadian Conference on Computer and Robot Vision (CRV’10)*, pp. 191–198, May 2010.

[16] Y. Zhuang, W. Wang, H. Chen, and K. Zheng, “3D scene reconstruction and motion planning for an autonomous mobile robot in complex outdoor scenes,” in *Proceedings of the International Conference on Modelling, Identification and Control (ICMIC’10)*, pp. 692–697, July 2010.

[17] A. Kelly, N. Chan, H. Herman et al., “Real-time photorealistic virtualized reality interface for remote mobile robot control,” *International Journal of Robotics Research*, vol. 30, no. 3, pp. 384–404, 2011.

[18] C. Kertész, “Texture-based foreground detection,” *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 4, no. 4, pp. 51–61, 2011.

[19] M. Fattah, “The use of MSVM and HMM for sentence alignment,” *Journal of Information Processing Systems*, vol. 8, no. 2, pp. 301–314, 2012.

[20] P. Vernaza, B. Taskar, and D. D. Lee, “Self-supervised terrain classification via discriminatively trained submodular Markov random fields,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA ’08)*, pp. 2750–2757, May 2008.
