A simple modeling of complex environments for mobile robots

Jun Miura*

Department of Information and Computer Sciences,
Toyohashi University of Technology
E-mail: jun@ics.tut.ac.jp
*Corresponding author

Suguru Ikeda

Department of Mechanical Engineering, Osaka University
Currently with NTT Data corporation.

Abstract: This paper deals with 3D environment modeling for mobile robots. The environment model we would like to generate is to be used as a medium of communication of location information between users and mobile robots. For such communication, a very precise environment model is not necessary. We thus develop a method of generating 3D model which is simple enough to generate fast but has sufficient geometric and appearance information. We also develop a robot localization method and a graphical user interface, and verify that the model can actually support such communication.

Keywords: 3D environment modeling; mobile robots; active contour model

Reference to this paper should be made as follows: J. Miura and S. Ikeda, 'A simple modeling of complex environments for mobile robots', *Int. J. Intelligent Systems Technology and Applications,* Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Jun Miura received Dr.Eng. degree in information engineering in 1989 at University of Tokyo. From April 1989 to March 2007, he had been a faculty member at Department of Mechanical Engineering, Osaka University. Since April 2007, he has been a Professor at Department of Information and Computer Sciences, Toyohashi University of Technology. From March 1994 to February 1995, he was a Visiting Scientist at Computer Science Department, Carnegie Mellon University, Pittsburgh, PA. He received the Best Paper Award from the Robotics Society of Japan in 1997. Prof. Miura published over 100 papers in international journals and conferences in the areas of intelligent robotics, computer vision, and artificial intelligence.

Suguru Ikeda received B.Eng. and M.Eng. degrees in mechanical engineering in 2005 and 2007, respectively, at Osaka University, Suita, Japan. He joined NTT Data corp. in 2007.

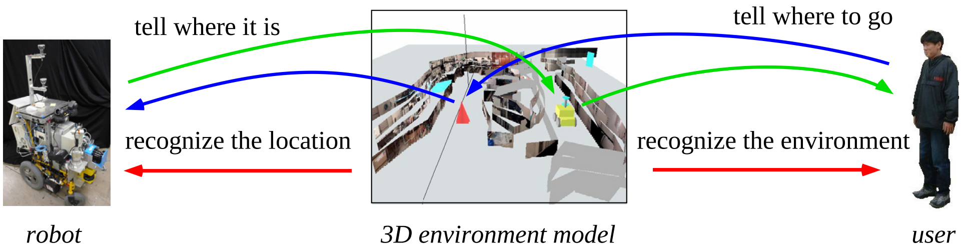


Figure 1 Four activities in communicating location information via an environment model.

1 Introduction

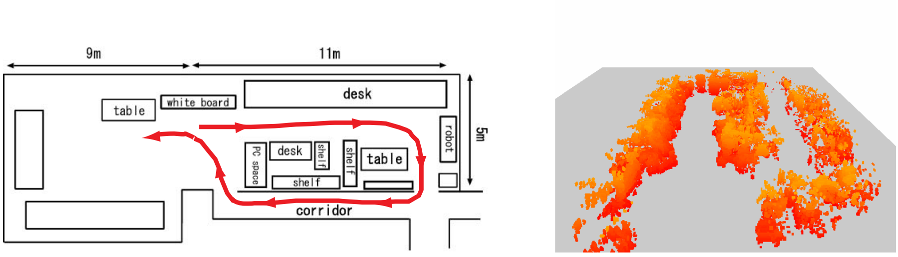
3D environment modeling is one of the active research areas in robotics and computer vision. Most previous works deal with generation of precise 3D models using a large amount of data and elaborate statistical and geometrical estimation techniques (Stamos, 2002; Fleck, 2005; Nevado, 2004; Sato, 2002). These works mainly focus on generating as precise models as possible to be used for applications such as virtual reality and telepresence; preciseness of the generated model is sometimes more important than efficiency of the modeling process. Thrun et al. (Thrun, 2003) proposed a method of generating multi-planar models from dense range data and image data using an improved EM algorithm. The method seems fast but only applicable to a relatively simple corridor environment surrounded mostly by large vertical planes.

We are interested in modeling usual office environments which have various shaped objects. For such environments, precise modeling approaches will require much computation, while fast, planar surface-based approaches may fail to capture the geometric characteristics of the environment. A modeling method is, therefore, desirable which can efficiently model the environment with *sufficient* details; obviously, whether a degree of details is sufficient depends on how the model is used.

This paper describes an approach to environment modeling for robots. The environment model we would like to generate is to be used as a medium for communication of location information between users and mobile service robots. Mobile service robots moves in an environment autonomously to perform service tasks such as bringing drinks and operating appliances. For a user and a mobile robot to communicate with each other about tasks of moving to some specific positions, they should be able to perform the following four activities (see Fig. 1):

1. The user recognizes the environment.
2. The user tells the robot where to go.
3. The robot recognizes the current location.
4. The robot tells the user where it is.

Both 3D geometric and appearance information are certainly useful for supporting such activities. A very precise environment model is, however, not necessary; the model has to keep only a necessary level of information. We thus develop a method of generating *approximate* models efficiently, and verify if the model can actually support the above four activities.



(a) An indoor environment. (b) 3D obstacle map.

Figure 2 A result of 3D obstacle map generation.

2 Geometric modeling

Usual indoor environments in which many objects exist are not formed only by vertical planes. To simplify the modeling process while keeping a certain degree of geometric information in the model, we represent an environment with a set of layered 2D contours with textures; contours in each layer approximate the shape of 2D free space in the corresponding height interval. We use the mobile robot in Fig. 1, which has vision and range sensor, for the environment modeling.

2.1 Generating 3D obstacle map

The first step of our 3D environment modeling is to generate a 3D obstacle map. We use a 3D probabilistic voxel map (Ikeda, 2006) as the representation of the obstacle map. Each voxel is a cube whose edge length is $5\,[cm]$, and holds the probability that an obstacle exist there. We use the data within $3\,[m]$ from the robot position for map generation. Voxels whose probabilities are higher than a threshold (currently, 0.8) are considered to be occupied by some obstacle. Fig. 2 shows an indoor environment and its 3D obstacle map; higher voxels in the map are drawn with brighter colors for increasing observability of voxels in the figure.

2.2 Generating layered contours from 3D obstacle map

We use four layers, three of which (layers 1 to 3) from omnidirectional stereo data and one (layer 0) from laser range finder (LRF) data, to cope with objects and walls at various heights in usual indoor environments. By considering the typical objects and the sensors' characteristics, the height ranges of the layers are determined as follows:

- *layer 3*: from $120 \sim 160\,[cm]$ for shelves and other high objects.
- *layer 2*: from $85 \sim 120\,[cm]$ for objects on tables.
- *layer 1*: from $60 \sim 85\,[cm]$ for tables and chairs.
- *layer 0*: lower than $60\,[cm]$ for other low objects.

We adopt active contour models (Blake, 1988) for determining the contours. All contours in the four layers are simultaneously refined so as to minimize an energy function, which considers the contours' smoothness, fitness to object data, the degree of passing the 2D free space, and consistency between layers. Initial contours are generated from the 2D

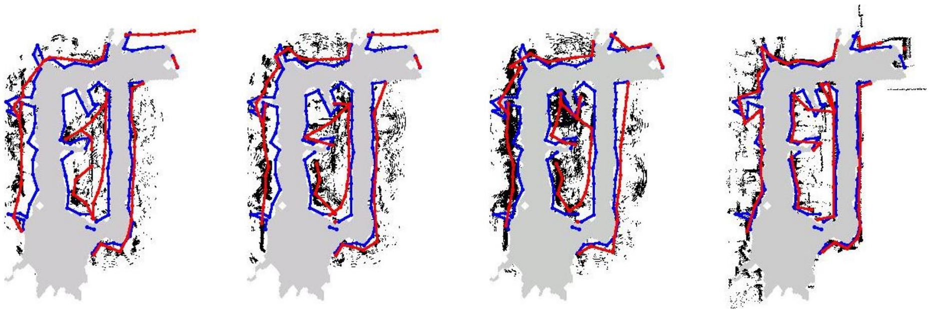

(a) Layer 3 (b) Layer 2 (c) Layer 1 (d) Layer 0

Figure 3 Initial (blue) and final (red) contours for the map in Fig. 2.

free space map, which is obtained by our 2D mapping method (Miura, 2002). The details are described in the next subsection.

2.3 Active contour model for contour generation

The first step of contour generation is to detect initial vertical plane segments. As the robot moves, obstacle data on the left and the right side are obtained. We separately detect initial contours on each side. We detect line segments by selecting points on the free space boundaries (Ikeda, 2006). Since we use a piecewise-linear model for 2D contours, which represent vertical planes in 3D, endpoints of the detected line segments are used as control points of the active contour model. This line segment detection is iterated as the robot moves by a certain distance.

We then refine the initial segments by using the following energy function for each control point:

(1) $$E = \alpha E_{int} + \beta E_{ext} + \gamma E_{btw} + \omega E_{free},$$

where each E_* is a sub-function defined below and α, β, γ, and ω are weights.

E_{int} is for evaluating the smoothness of a contour and the uniformness of segment lengths. It is defined as the sum of the following three values:

(2) $$E_{int}^1 = ||\boldsymbol{v}_c - \boldsymbol{v}_p|| + ||\boldsymbol{v}_c - \boldsymbol{v}_n||,$$

(3) $$E_{int}^2 = ||(\boldsymbol{v}_n - \boldsymbol{v}_c) - (\boldsymbol{v}_c - \boldsymbol{v}_p)||,$$

(4) $$E_{int}^3 = |||(\boldsymbol{v}_n - \boldsymbol{v}_c)|| - k| + |||(\boldsymbol{v}_c - \boldsymbol{v}_p)|| - k|,$$

where $\boldsymbol{v}_c$ is the position of the control point currently under consideration, $\boldsymbol{v}_p$ and $\boldsymbol{v}_n$ indicate those of the previous and the next points, respectively, $||\cdot||$ indicates the Euclidean norm and k is a constant.

E_{ext} is for evaluating the fitness to obstacle data and is defined as:

(5) $$E_{ext} = -G * I_{obs},$$

where I_{obs} is the binary image representation of 2D obstacle map in one layer (one pixel corresponds to $5\,[cm] \times 5\,[cm]$ cell), G is the Gaussian operator and $*$ denotes convolution. σ of G is $4.0\,[pixel]$ in layer 0, and $6.0\,[pixel]$ in the other layers.

E_{btw} is for evaluating consistency between layers, and is defined using distance D_1 to the contours in neighboring layers:

$$(6) \quad E_{btw} = \begin{cases} -\frac{100}{D_1+1} & (0 \leq D_1 < d) \\ 0 & (d \leq D_1) \end{cases}$$

where d is a threshold (currently, 10).

E_{free} is the penalty for entering the 2D free space, and is defined using distance D_2 to the free space boundary:

$$(7) \quad E_{free} = \begin{cases} D_2 & \text{(inside free space)} \\ 0 & \text{(outside free space)} \end{cases}$$

The summation of E for all control points in all layers is the objective function to be minimized. The minimization steps are as follows (in C-like code):

```
// Sequential minimization
refine_contour(){
  // Minimize energy without $E_{btw}$ in each layer
  apply_acm($\alpha_1$, $\beta_1$, $\gamma_1$, $\varepsilon_1$)
  // Minimize energy in all layer
  apply_acm($\alpha_2$, $\beta_2$, $\gamma_2$, $\varepsilon_2$)
}
//active contour refinement
apply_acm($\alpha$, $\beta$, $\gamma$, $\varepsilon$){
  while (true) do {
     for i = 3 to 0 do {// Minimize energy in each layer
        for each control point do {
           Move to one of 8-neighbors
           which minimizes the energy of the point; }}
     if no point has moved in this round then break;
}}
```

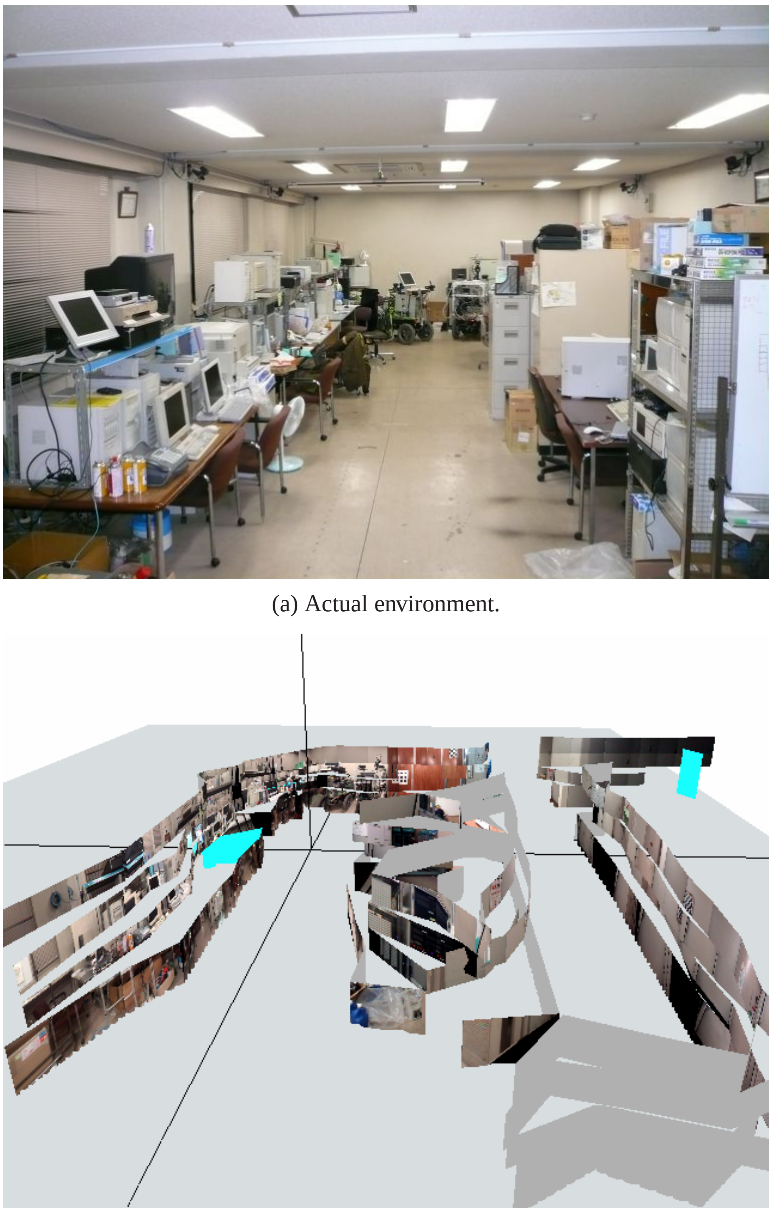
We currently use the following weights in eq. (1): $(\alpha_1, \beta_1, \gamma_1, \varepsilon_1) = (1, 100, 0, 100)$ and $(\alpha_1, \beta_1, \gamma_1, \varepsilon_1) = (1, 100, 50, 100)$. At the first stage of the minimization, the consistency between layers is ignored (i.e., $\gamma_1 = 0$), while at the second stage, all energy sub-functions are considered. This two-stage minimization is for certainly modeling the gap between layers such as the one between PCs on the table and walls.

Fig. 3 shows the initial and the final plane segments. It is clearly shown that the final plane segments fit well to the object data while having sufficient smoothness.

3 Texture extraction and mapping

Appearance of objects is useful for robot localization and human-robot interface. We therefore extract textures from the images taken by two pan-tilt-zoom (PTZ) cameras and map them to the generated plane segments.

We can extract textures for one plane segment from several images taken at various robot positions. In order to get the best textures, we select the image which provides the highest resolution; that is, we select the robot position which maximizes the area of the mapped region of the segment in the image. The extracted textures are stored as images of the size of 256×256 pixels.



(a) Actual environment.

(b) Modeling result.

Figure 4 The modeling result of an environment.

4 Modeling results

Fig. 4 shows a generated model of our laboratory, with an actual view of the environment. Fig. 5 compares the modeled and the actual scenes for four different viewpoints. The results show that the proposed method models the geometry and the appearance of the environment relatively well.

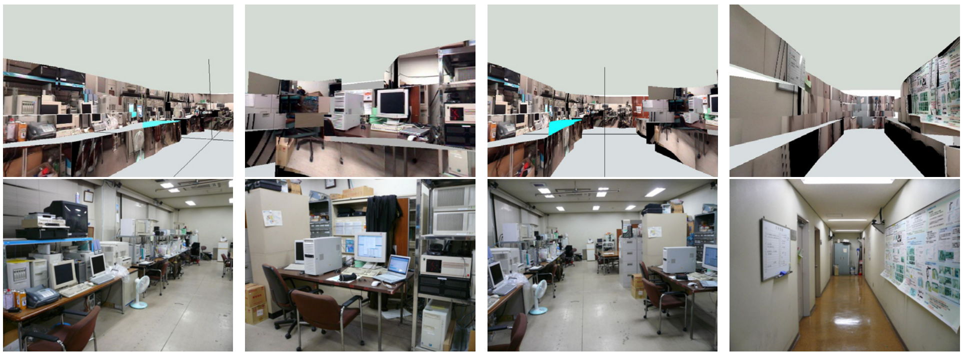


Figure 5 Comparison of modeled (upper) and actual (lower) scenes.

The total calculation time for modeling with 235 observations was about $700\ [s]$, the breakdown of which are: $183\ [s]$ for obstacle map generation, $17\ [s]$ for contour generation and refinement, $500\ [s]$ for texture generation and mapping. Since we had to move the PTZ cameras to acquire images of a wide scene, the texture mapping took a large part of the total time. When we used the omnidirectional images for texture extraction, it took only $43\ [s]$ in the expense of low texture quality. We also implemented and tested an on-line modeling system using two wireless-connected PCs; it actually worked but the latency was not small due to a slow wireless communication.

5 Using the model for human-robot communication

The objective of the research is to develop a method of generating 3D environment models which are to be used for human-robot communication of location information. To realize this, we develop a robot localization method and a graphical user interface (GUI) for users to communicate with robots.

5.1 Robot localization using the model

We develop a localization method which uses both the geometric and the appearance information. We try to select the most-matched position from a given set of candidate positions. For each candidate position, we compare the test data with the environment model by first using geometric information and then using appearance (i.e., texture) information, and select the best matched position.

The robot generates a local 3D map from several (currently, 10) consecutive data of omnidirectional stereo and LRF, and uses it as the test data for localization.

The *range profile*, which shows the distance to the nearest object in each direction centered at the robot position, is used as the representation of geometric property of a location, as shown in Fig. 6. We also make the range profile from the local map. We compare two profiles using an SAD (sum of absolute difference) measure with changing orientation (i.e., with shifting the profile), and select a set of candidate positions which have small differences. The number of selected candidates is limited to a predetermined percentage to the total number of candidates.

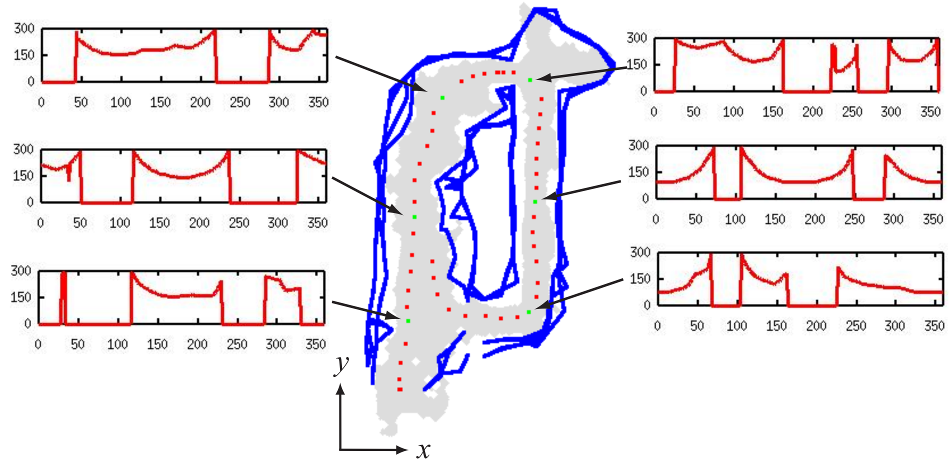
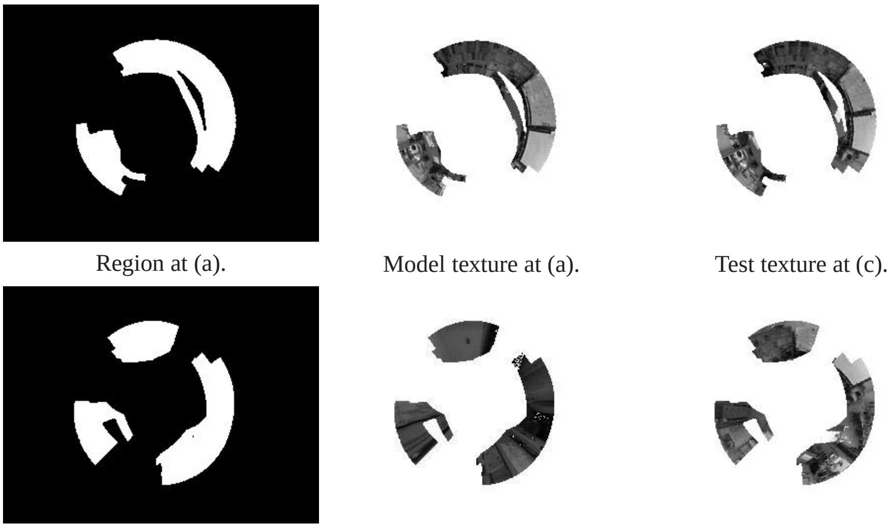


Figure 6 50 Candidate positions (red) and examples of the range profiles (at layer 2) at several candidate positions.

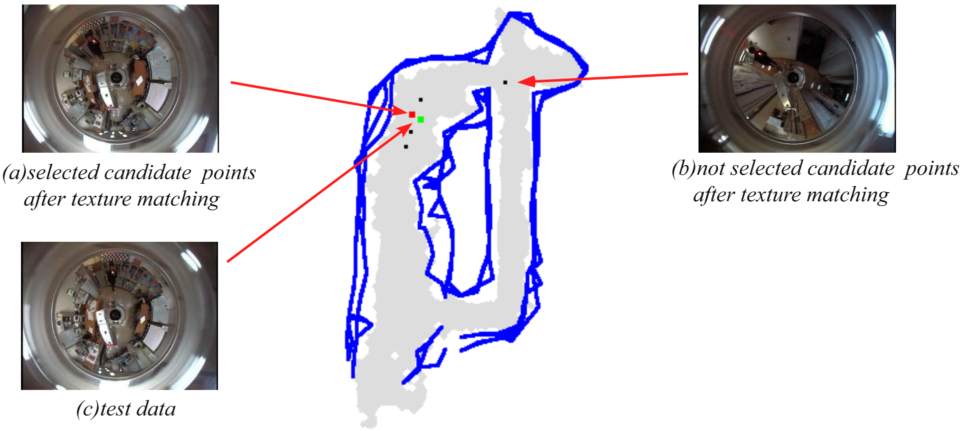


Region at (a). Model texture at (a). Test texture at (c).

Region at (b). Model texture at (b). Test texture at (c).

Figure 7 Mapped regions and textures of model and test data at positions (a), (b), and (c) in Fig. 8.

We then use texture information for determining the best position. For each remaining candidate position after the geometry matching, we extract the region in the omnidirectional image corresponding to the plane segments in the model using the robot pose (position and orientation) of the candidate position. Fig. 7 shows examples of texture comparison for two candidate positions (position (a) in Fig. 8 in the upper row and position (b) in the lower row). For each candidate position, the mapped region, the texture extracted from the model, and the test texture extracted from an input image (taken at position (c)) are shown. We compare the model and the test texture using the SAD measure



(a)selected candidate points after texture matching

(b)not selected candidate points after texture matching

(c)test data

Figure 8 Result of texture matching. The red point, the green point, and the black points are the selected candidate position, the position of the test data, and the remaining candidate positions after geometry matching, respectively.

and determine the position which minimizes the texture difference. Fig. 8 shows a result of localization.

We use as candidate positions the observation positions where the robot took data for modeling. We prepared a different set of 140 positions as test data. In the experiment, the maximum percentage of the kept candidates after geometric matching was set to $20\,[\%]$; a position is considered to be correctly recognized if the distance between that position and the selected candidate position is within $1.5\,[m]$. For 140 test positions, 128 positions passed the geometric matching (i.e., the correct candidate is among the kept candidates) and 122 positions were finally correctly recognized. The success rate was $87\,[\%]$. The result shows that our 3D model contains enough information for robot localization. Use of filtering techniques (e.g., Kalman filter) would further improve the localization performance.

5.2 *A GUI for human-robot communication*

The previous subsection verifies that the model is effective in supporting one of the four activities (i.e., robot recognizes location) mentioned in the introduction section. This subsection describes how the other three activities are supported by our GUI.

Fig. 9 shows the GUI we developed. The screen is divided into the following six areas:

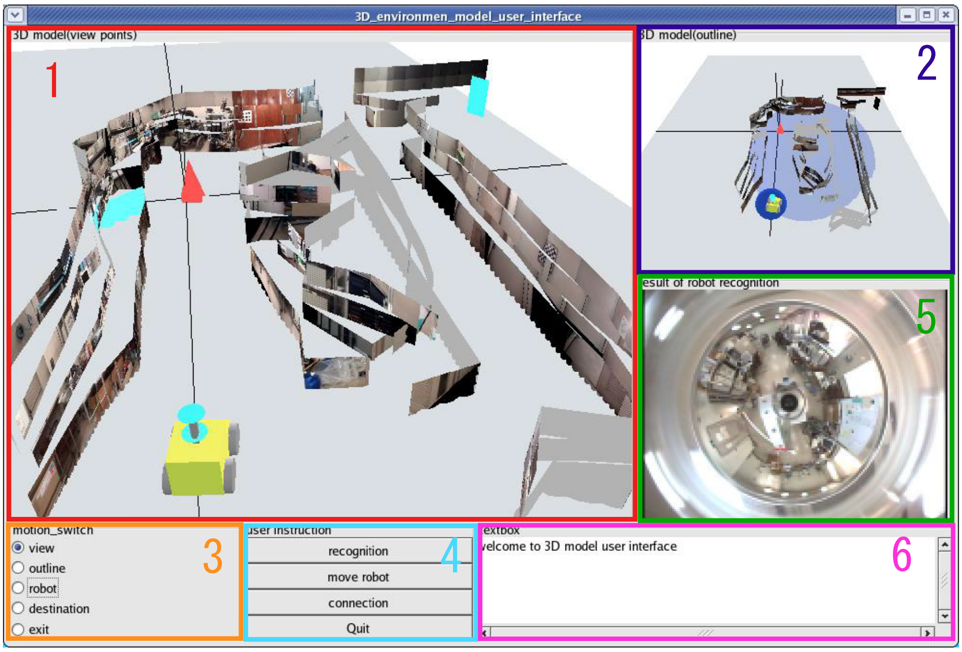
Area 1 displays the generated model with the robot (yellow figure) and the destination (red cone). The user can move the viewpoint and the destination using a 6-axis pointing device (by *3D connexion*). The viewpoint can also be automatically set to follow the robot's viewpoint.

Area 2 is for the birds-eye-view of the whole environment. The viewpoint can be adjusted using the same device.

Area 3 is for selecting items to operate.

Area 4 is for issuing commands to the robot.

Area 5 is for displaying the current image from the robot.



● view
○ outline
○ robot
○ destination
○ exit

velcome to 3D model user interface

Figure 9 A graphical user interface.

Area 6 is for the messages from the system.

Since the current robot position and the user-specified destination are displayed on the screen, and since the user can change the viewpoint easily, the robot can tell the user where it is and the user can tell the robot where to go.

We tested the usability of the system for five subjects among the members of our laboratory. Each subject was asked to do the following three steps:

1. Recognize the environment with changing the viewpoint by using the pointing device.
2. Set a destination at an arbitrary position.
3. Move the robot to come to the destination.

The evaluations of the subjects were all positive (4.5/5 points in average).

Fig. 10 shows a sequence of remotely moving the robot to the destination. The user can monitor the state of the robot on-line. The *Area 1* of the GUI shows the views of the model from the robot. At the left-bottom corner of the figures, the actual movements of robot are superimposed.

6 Conclusions and Discussion

This paper has described a method of generating 3D indoor environment model. The model is intended to be used for communication of location information between users and mobile services robots. The model is generated by the following three steps: generation of 3D obstacle map by temporal integration of stereo and LRF data, detection of plane

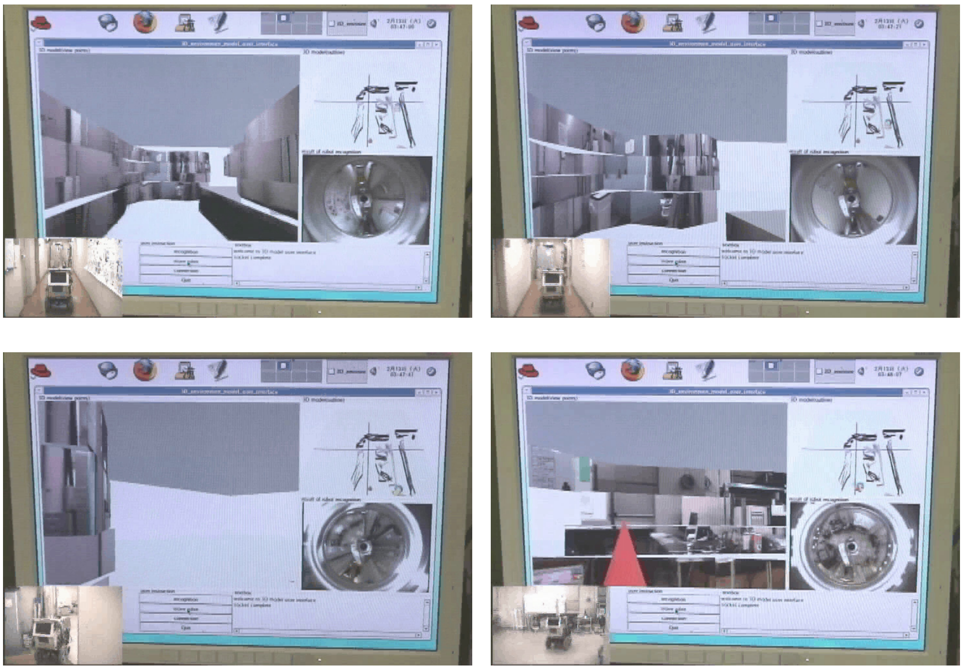


Figure 10 Snapshots of a remote control of the robot.

segments in four layers using an active contour model, and extraction and mapping of textures from images.

The model is efficiently generated for a complex environment but carries enough information for human-robot communication. We have shown this by verifying if the four activities in the communication can be supported by the model, using both a robot localization method and a GUI.

A future work is to speed up the modeling process. We currently use all data acquired during the robot movement. A part of data, however, could be enough for modeling, and reduction of data size will decrease the computation time. We also plan to test the method in various indoor environment to examine its robustness.

Another future work is to add an object recognition capability and to put semantic information such as object names and functions (Siegwart, 2007) or place names (Saffiotti, 2005) in the map so that a user can communicate with a robot using symbols.

References

Stamos, I. and Allen, P.K. (2002) 'Geometry and Texture Recovery of Scenes of Large Scale', *Computer Vision and Image Understanding*, Vol. 88, pp. 94–118.

Fleck, S. et al. (2005) 'Omnidirectional 3D Modeling on a Mobile Robot using Graph Cuts', In *Proceedings of 2005 IEEE Int. Conf. on Robotics and Automation*, pp. 1760–1766.

Nevado, M.M., Garcia-Bermejo, J.G. and Casanova, E.Z. (2004) 'Obtaining 3D models of indoor environments with a mobile robot by estimating local surface directions', *Robotics and Autonomous Systems*, Vol. 48, pp. 131-143.

Sato, T. et al. (2002) 'Dense 3-D Reconstruction of an Outdoor Scene by Hundreds-baseline Stereo Using a Hand-held Video Camera', *Int. J. of Computer Vision*, Vol. 47, Nos. 1–3, pp. 119–129.

Thrun, S. et al. (2003) 'A Real-Time Expectation Maximization Algorithm for Acquiring Multi-Planar Maps of Indoor Environments with Mobile Robots', *IEEE Trans. on Robotics and Automation*, Vol. 20, No. 3, pp. 433–442.

Ikeda, S. and Miura, J. (2006) '3D Indoor Environment Modeling by a Mobile Robot with Omnidirectional Stereo and Laser Range Finder', In *Proceedings of 2006 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 3435-3440.

Blake, A. and Isard, M. (1988) 'Active Contours', Springer-Verlag.

Miura, J., Negishi, Y. and Shirai, Y. (2002) 'Mobile Robot Map Generation by Integrating Omnidirectional Stereo and Laser Range Finder', In *Proceedings of 2002 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 250-255.

Vasudevan, S., Gächter, S., Nguyen, V. and Siegwart, R. (2007) 'Cognitive Maps for Mobile Robots – An Object Based Approach', *Robotics and Autonomous Systems*, Vol. 55, No. 5, pp. 359–371.

Galindo, C., Saffiotti, A., Coradeschi, S., Buschka, P., Fernández-Madrigal, J.A. and González, J. (2005) 'Multi-Hierarchical Semantic Maps for Mobile Robotics' In *Proceedings of 2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 3492–3497.