The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Obstacle Classification and Location by Using a Mobile Omnidirectional Camera Based on Tracked Floor Boundary Points

Tsuyoshi Tasaki and Fumio Ozaki

Abstract— **Locating all obstacles around a moving robot and classifying them as stable obstacles or not by a sensor such as an omnidirectional camera are essential for the robot's smooth movement and avoiding problems in calibrating many cameras. However, there are few works on locating and classifying all obstacles around a robot while it is moving by only one omnidirectional camera. In order to locate obstacles, we regard floor boundary points where robots can measure the distance from the robot by one omnidirectional camera as obstacles. Tracking them, we can classify obstacles by comparing the movement of each tracked point with odometry data. Moreover, our method changes a threshold to detect the points based on the result of comparing in order to enhance classification. The classification ratio of our method is 85.0%, which is four times higher than that of a method without changing a parameter to detect the points.**

I. Introduction

It is important for moving robots to locate obstacles and classify them as stable obstacles or not. Many works use distance measurement devices such as the Laser Range Finder (LRF) and stereo cameras [1]. However, robots have to be equipped with more than one sensor when they classify all obstacles around them at once by these sensors. Using many LRFs is expensive, and calibration is troublesome.

An ominidirectional camera can take images of all obstacles around a robot simultaneously while moving. Therefore, it is often used for localization of robots [2]. However, it is difficult to apply previous obstacle classification techniques such as [3] to omnidirectional images without changing them to general images. Even if we change images, previous techniques do not work well because changed images lose a lot of information. Moreover, classifying obstacles as stable or not by a moving camera is more difficult than classifying by a static camera.

We have developed an original method of classifying all obstacles around the robot by only one omnidirectional camera while moving. In order to locate and classify obstacles, we focus on floor boundary points where the robot can measure the distance from itself by only one omnidirectional camera. Our robot classifies a floor boundary point as a dynamic obstacle when its movement is different from the robot's movement. In particular, we aim to detect dynamic obstacles to ensure the robot's smooth movement.

Section II describes how to find the floor boundary points. Section III describes the obstacle classification method based on the result of tracking floor boundary points. In Section IV, we confirm an ability of our floor detection method and accuracy of our classification method. Section V concludes this paper.

Tsuyoshi Tasaki is with the Corporate Research & Development Center, TOSHIBA CORPORATION, Japan , mail: tsuyoshi.tasaki@toshiba.co.jp
Fumio Ozaki is with the Corporate Research & Development Center, TOSHIBA CORPORATION, Japan , mail: fumio.ozaki@toshiba.co.jp

II. Floor Boundary Points Detection

A. Floor Detection by Ward's clustering

We use floor colors for floor detection because floor colors are simple. Previous works use the Gaussian Mixture Model (GMM) for specific color detection [4]. The GMM can detect many specific colors, increasing a number of a mixed Gaussian. However, we have to evaluate the GMM many times in order to decide parameters such as the number of mixed Gaussian. Therefore, it is difficult for robots to apply the GMM to various environments quickly and accurately just after they start up.

Our robot learns representative colors of the floor by itself based on the distribution of floor color data without prior setting. Considering the distribution, our floor detection method can adjust more easily than the GMM can and detects the floor as accurately as the GMM does. Here, in order to detect the representative colors of the floor, we use Ward's clustering [5], which is one of the hierarchical clustering methods. Our robot selects the representative colors by Ward's clustering as follows.

1) Our robot takes an image and gets N color data from pixels to which the close area around it is projected. In an initial state, each datum shows a representative color. A cluster of color data that are similar to the representative color i is denoted by C_i .
2) We choose two clusters C_1 and C_2 that minimize D as shown in Eq. (1), and create a new claster C_k that consists of the data in both C_1 and C_2. Let c_i denote an average color vector in the cluster C_i.

$$D(C_1, C_2) = d(C_1 \cup C_2) - d(C_1) - d(C_2)$$
$$d(C_i) = \sum_{x \in C_i} \| x - c_i \| \quad (1)$$

3) In step 2), when C_k satisfies both Eq. (2) and Eq. (3), it is decided that c_k is the representative color and data in C_k are not used for following loops. When C_k satisfies only Eq. (2), data in C_k are just not used for following loops. T_D and T_N is a constant threshold, $|C_k|$ is a number of the data in C_k.

$$\min_{k \neq i} D(C_k, C_i) > T_D \quad (2)$$

$$|C_k| > T_N \quad (3)$$

4) Step 2) and 3) continue until all data are not used.

Because Ward's clustering considers the distribution of data, each cluster is identified easily by Mahalanobis distance. A color datum I is classified as floor color when we find a C_m that satisfies Eq. (4). μ_m, Σ_m and σ denote an average vector, a covariance matrix of data in C_m and a threshold, respectively.

$$\sqrt{(I - \mu_m)^T \Sigma_m^{-1} (I - \mu_m)} < \sigma \quad (4)$$

When a robot uses an omnidirectoinal camera mounted on its head, the floor is projected to around the image center. Therefore, our robot classifies the pixels from center to outer by applying Eq. (4). If our robot finds continuous p pixels that do not satisfy Eq. (4), a floor boundary point is detected at the position where the first pixel in p pixels is located.

B. Transforming Coordinates of Floor Boundary Points from Image Coordinates to Robot Coordinates

In the case of using an omnidirectional camera incorporating a hyperbolic mirror, a position (X, Y, Z) on the robot coordinates is projected to a position (x, y) on the image coordinates as follows. Constant b and c denote proper parameters of the mirror, and f denotes a focal distance.

$$x = \frac{Xf(b^2 - c^2)}{(b^2 + c^2)(Z - c) - 2bc\sqrt{X^2 + Y^2 + (Z - c)^2}}$$
$$y = \frac{Yf(b^2 - c^2)}{(b^2 + c^2)(Z - c) - 2bc\sqrt{X^2 + Y^2 + (Z - c)^2}} \quad (5)$$

Many robots are equipped with an omnidirectional camera, and they can measure or know the distance from the floor to the camera while they are moving [6]. Therefore, with regard to floor boundary points, the variable Z in Eq. (5) becomes constant, and we can measure the distance from the robot to floor boundary points by applying Eq. (5).

In order to decide the parameters Z, b, c and f, we have drawn cross-stripes on the floor as shown in Fig. 1. n pairs of

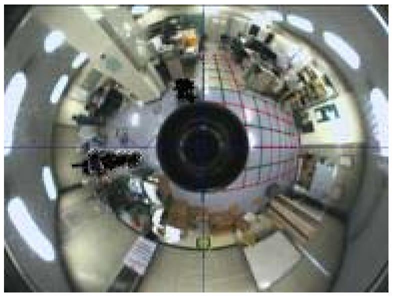

Fig. 1. The omnidirectional image of the cross-stripes on the floor. This image is used for deciding the parameters.

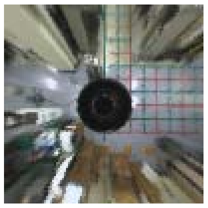

Fig. 2. The bird's-eye image. In this image, the distance between any two points on the floor is linear to the distance in the real space.

(X_m, Y_m) and (x_m, y_m) are acquired from the image to which n cross-points are projected. Here, (X_m, Y_m) and (x_m, y_m) denote the position of the cross-point m on the robot coordinates and the image coordinates, respectively. Using n pairs, parameters that minimize the evaluation function F as shown in Eq. (6) are decided by the downhill simplex method.

$$F = \sum_{m=0}^{n-1} \left| x_m - X_m f \frac{b^2 - c^2}{(b^2 + c^2)(Z - c) - 2bc\sqrt{X_m^2 + Y_m^2 + (Z - c)^2}} \right| + \sum_{m=0}^{n-1} \left| y_m - Y_m f \frac{b^2 - c^2}{(b^2 + c^2)(Z - c) - 2bc\sqrt{X_m^2 + Y_m^2 + (Z - c)^2}} \right| \quad (6)$$

For confirmation of parameters, a bird's-eye image is created by using the decided parameters. Fig. 2 shows the bird's-eye image. The lines that make cross-stripes on the floor are not distorted, because the decided parameters are corrected. Here, 1 pixel in this bird's-eye image denotes about 5 cm in the real world.

III. Obstacle Classification By Floor Boundary Points

A. Classification Equation

A floor boundary point m on the image at time t-dt is detected by the method as shown in Section II. dt depends on a processing speed. If the point m can be tracked from t-dt to t correctly, the position of m at t is located correctly on the image at t. It is easy to transform the coordinates of m at t-dt and t from the image coordinates to the robot coordinates $(X_m, Y_m)^{(t\text{-d}t)}$ and $(X_m, Y_m)^{(t)}$ by referring to the bird's-eye image. The relative position (dX, dY, dΘ) from t-dt to t is estimated by odometry data. dΘ is based on the direction from the center of the robot to the front of the robot at t-dt. When m is located at the boundary between a stable obstacle and the floor, $(X_m, Y_m)^{(t)}$ is calculated by (dX, dY, dΘ) and $(X_m, Y_m)^{(t\text{-d}t)}$, as shown in Eq. (7).

$$\begin{pmatrix} X_m \\ Y_m \end{pmatrix}^{(t)} = \begin{pmatrix} \cos d\Theta & -\sin d\Theta \\ \sin d\Theta & \cos d\Theta \end{pmatrix} \begin{pmatrix} X_m \\ Y_m \end{pmatrix}^{(t-dt)} + \begin{pmatrix} dX \\ dY \end{pmatrix} \quad (7)$$

When m is located at the boundary between a dynamic obstacle and the floor, Eq. (7) is not satisfied. Therefore, we can regard Eq. (7) as a Classification Equation (CE), that is, the floor boundary point m can be classified as a stable obstacle or a dynamic one by confirming whether Eq. (7) is satisfied or not. Actually, Eq. (7) includes a small error $\boldsymbol{\varepsilon}$ depending on an image resolution, which is ignored.

The following conditions should be satisfied in order to regard Eq. (7) as the classification equation.

1. Floor boundary points have to be located at the boundary between obstacles and the floor correctly in the image.
2. Floor boundary points have to be tracked correctly.
3. Camera parameters have to be decided correctly.
4. Odometry has to be calculated correctly.

Condition 4 is satisfied in the general environment, because the odometry is comparatively correct during short movement. Fig. 1 verifies that parameters are not so bad that condition 3 is satisfied, too. Floor boundary points can be tracked easily and tracking is not a major problem when they are detected accurately, because they are located at the boundary where the colors change significantly. However, floor boundary points cannot always be detected correctly by using only the floor colors in various environments. We apply the result of confirming whether the CE is satisfied or not to the floor detection method.

B. Obstacle Classification

The CE is satisfied as long as floor boundary point m is located on the floor. One of the reasons why m is not located on the floor is that the threshold σ in Eq. (4) is inappropriate. When the position of m does not satisfy the CE, σ is too large or m shows a dynamic obstacle. For confirmation, new floor boundary point m' is detected by decreasing σ in the direction where m is located to σ-dσ. The parameter dσ should be small so that the robot does not narrow the floor area. The new floor boundary point m' is tracked from t to t-dt and classified by confirming the CE again. When the position of m' satisfies the CE, our robot regards m as a stable obstacle. Moreover, the position of m is changed to the position of m'. Conversely, if it is not satisfied, m is regarded as a dynamic obstacle. Our method changes the parameter dynamically by the result of the CE. For example, in Fig. 3, the position of floor boundary point a located at the boundary between the floor and a static obstacle satisfies the CE. The point b that is not located at the boundary does not satisfy the CE. Therefore, b creates a new floor boundary point b' and b' is tracked from t to t-dt. Using the result of tracking, our robot confirms whether b' satisfies the CE or not. Because b' is located at the boundary, b' satisfies the CE in this case. Therefore, the position of b is changed to the position of b' and b is classified as a static obstacle. The point c located at the boundary between a dynamic obstacle and the floor also does not satisfy the CE. The point c creates a new point c' and its position is confirmed. Because dσ is small, the point at the boundary does not create a new point far from the original point. The position of c' does not satisfy the CE in this case, and c is regarded as a dynamic obstacle.

If the threshold is low at the beginning of the robot's activation, all points are located on the floor. However, they are located between the boundary and the robot, and free space looks very small. Our classification method first uses high thresholds and detects the boundary that is a little larger than the true boundary. Moving and confirming the CE refine the threshold of each direction where the floor boundary point

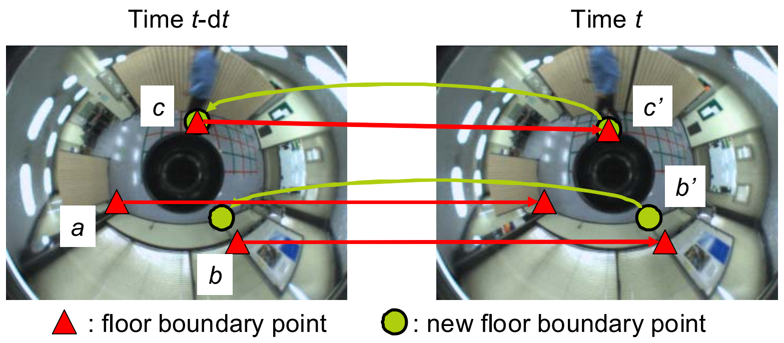

Fig. 3. The example of classification process by using floor boundary points. A point a shows a stable obstacle. A point b is relocated and classified as a static obstacle. A point c shows a dynamic obstacle.

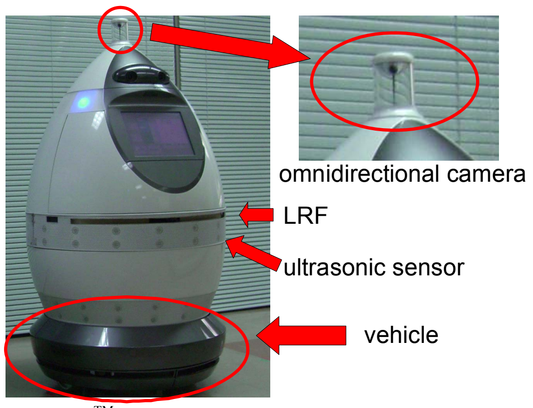

Fig. 4. ApriTau^{TM}. It is 120 cm tall.

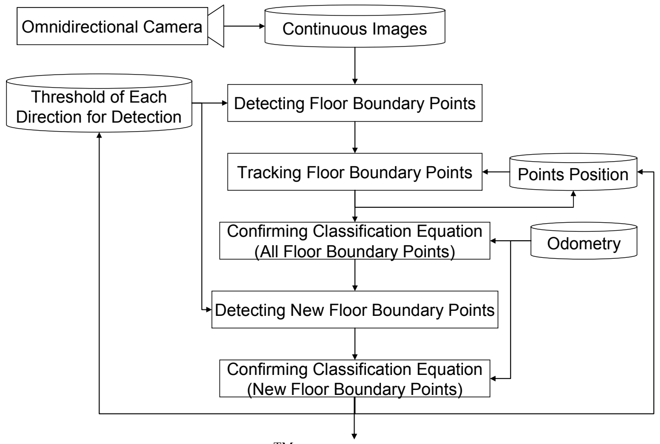
Omnidirectional Camera
Continuous Images
Threshold of Each Direction for Detection
Detecting Floor Boundary Points
Tracking Floor Boundary Points
Points Position
Confirming Classification Equation (All Floor Boundary Points)
Odometry
Detecting New Floor Boundary Points
Confirming Classification Equation (New Floor Boundary Points)

Fig. 5. The system in ApriTau^{TM}. The inputs are continuous omnidirectional images. The outputs are the results of the classification.

classified as a dynamic obstacle is located. Finally, the robot adapts the threshold of each direction and makes it possible to locate and classify obstacles accurately.

IV. Evaluation

A. Our Robot and Obstacle Classification System

Our classification method is implemented on our robot called ApriTau^{TM} as shown in Fig. 4. It has a vehicle that can acquire the odometry data. An omnidirectional camera is mounted on the top of its head and does not move with the head motion. Taking images while moving by means of its vehicle, it synchronizes the odometry data.

Fig. 5 shows our classification system. ApriTau^{TM} takes images whose size is 320x240 (pixels) continuously at 30fps and inputs them to the system. In image at t-dt, the system detects 360 floor boundary points using the result of tracking previous points or the floor detection method. Red or blue

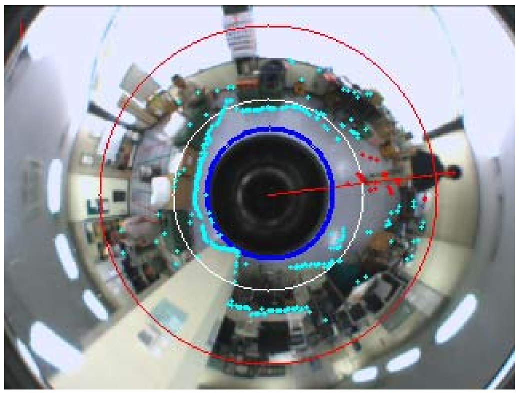

Fig. 6. The output of classification system. Blue points and red points show stable obstacles and dynamic ones, respectively.

points are floor boundary points in Fig. 6. 360 points are detected every one degree. If we use more points, they form complete floor boundary. However, we think 360 points are sufficient for the robot's movement. These points are tracked and classified. In Fig. 6, blue and red points are classified as stable obstacles and dynamic obstacles, respectively. Most of them are located at the boundary between the floor and obstacles. A red line is drawn from the image center to the average of red points' positions. This system integrates floor boundary points which are classified as dynamic obstacle like the red line, when points which are classified as dynamic obstacles are located near the other points which are classified as dynamic obstacles.

B. Evaluation of Floor Detection

1) *Setting:*

In order to evaluate the ability of the floor detection, we compared our method with the previous floor detection method based on the GMM. Two datasets are used in this experiment. Each dataset consists of learning data and test data.

First, color distribution is learned and parameters such as the number of mixed Gaussian and the threshold are optimized for the GMM by using learning data of the first dataset. Our method learned color distribution automatically, and parameters such as T_N, T_D, and σ in Section II are optimized manually. Using optimized parameters and learned colors, both methods are tested by test data of the first dataset.

Next, both methods learn only color distribution by using learning data of the second dataset. Using learned colors and parameters optimized by the first dataset, both methods are tested by test data of the second dataset. The second dataset is obtained in various places such as an experimental room, a center corridor in a mock store, and so on.

In both test cases, the following two values (Hit and Correct Rejection: CR) are calculated for evaluation. P_f denotes the number of pixels to which the floor is projected. P_{cf} denotes the number of pixels detected correctly as the floor. P_o denotes the number of pixels to which objects except the floor are projected. P_{co} denotes the number of pixels detected correctly as objects except the floor. Although pixels which are close to the robot are more important than pixels which are far from it for the robots' smooth movement, in this experiment we regard all pixels as equal because we want to evaluate only the ability of the floor detection.

$$\begin{aligned} Hit &= P_{cf} \big/ P_f \\ CR &= P_{co} \big/ P_o \end{aligned} \tag{8}$$

2) Result and Discussion

The outputs of the first test and the second test are shown in Fig. 7 and Fig. 8. In the figures, (a) and (b) are outputs of the GMM method and our method, respectively. White and black regions correspond to the floor and the objects except the floor, respectively. The calculated evaluation values are shown in TABLE I.

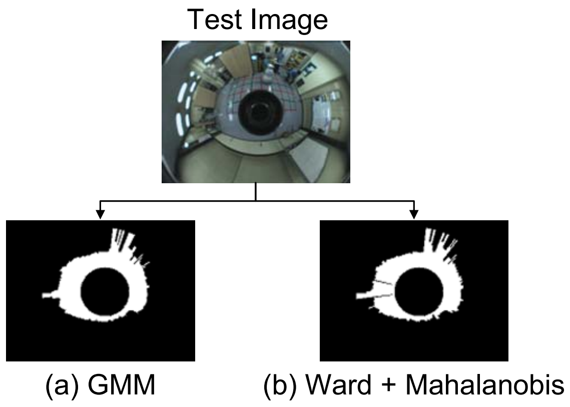

Fig. 7. The output of first test. Outputs of both methods are similar.

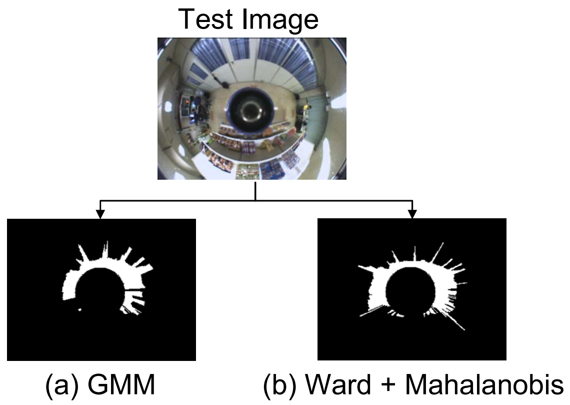

Fig. 8. The output of second test. The GMM cannot detect the floor which is far from our robot.

These results show that, when parameters are optimized manually, the ability of the GMM floor detection method is similar to that of our method. However, when parameters are not optimized, the Hit value of the GMM method is lower than that of our method. The result shows both maximum abilities are similar, but our method learns the floor colors more easily than the GMM does.

One of the reasons why the second Hit value calculated by the result of the GMM is low is that many experiments are needed in order to decide the parameter of the GMM. Comparing the GMM parameters, parameters of Ward's clustering and the threshold of Mahalanobis distance do not change dramatically because they are related to the distribution of learning data and decided based on them. In this experiment, the second test data includes more colors than the first test data does because of illumination changing and so on. The number of mixture Gaussian optimized at the first test is too small for the GMM to learn the floor colors completely.

TABLE I
COMPARING THE GMM METHOD WITH OUR METHOD

Method	Hit	CR
First Test The GMM (optimized parameters)	0.95 (7277 / 7655)	0.95 (65826 / 69145)
First Test Ours (optimized parameters)	0.94 (7210 / 7655)	0.96 (66029 / 69145)
Second Test The GMM (first test parameters)	0.73 (13204 /18022)	0.96 (203953 / 212378)
Second Test Ours (first test parameters)	0.90 (16245 /18022)	0.95 (202061 / 212378)

C. Evaluation of Obstacle Classification

1) Setting

In order to confirm the effectiveness of changing the threshold σ dynamically based on the result of the CE, we compared the classification ratio of our method with that of a simple method using a constant threshold. The experimental steps are as follows:

1. ApriTau^{TM} and another robot move on the given route.
2. ApriTau^{TM} takes images synchronized with odometry data continuously while moving.
3. The images and the odometry data are input to the systems of both our method and the simple method. Note that, although same data are input to both systems, each system processes some of them because of the difference of the processing speed.

4. The classification ratios of our method and the simple method are calculated by both outputs.

In this experiment, the classification ratio is the *F* value calculated by the recall ratio *R* and the precision ratio *P* as shown in Eq. (9). Here, *A*, *O* and *C* show the number of images to which another moving robot is projected, the number of obstacles the system classified as dynamic obstacles, and the number of dynamic obstacles the system outputs and locates correctly, respectively.

$$\begin{aligned} R &= C/A \\ P &= C/O \\ F &= 2RP/(R+P) \end{aligned} \tag{9}$$

2) Result and Discussion

The classification ratios of both methods are shown in TABLE II. The classification ratio of our method is 4 times higher than that of the simple method. In particular, the improvement of the precision ratio affects the *F* value. One of the reasons why the precision ratio of our method is much higher than that of the simple method is that ApriTau^{TM} can select floor boundary points showing candidates of dynamic obstacles by the CE and relocate points correctly by strengthening the threshold detecting each point. The result shows obstacles can be classified even if we regard obstacles as small points. Moreover, the accuracy of locating points greatly affects classification ratio.

TABLE II
THE CLASSIFICATION RATIOS

Method	Recall Ratio	Precision Ratio	F value
Simple	0.63 (10 / 16)	0.13 (10 / 79)	0.21
Dynamic	0.94 (17 / 18)	0.77 (17 / 22)	0.85

However, the precision ratio is too low for robots' smooth movement. In this paper, we assume errors of tracking points are very small, which is certainly correct to some extent for the image coordinates. In the case of omnidirectional camera image, the distance resolution changes depending on the distance from the image center. It is very low for a distant place. Tracking errors of a few pixels become errors of a few meters for the world coordinates. Because of errors of a few meters, Eq. (7) does not work as the CE. When the floor boundary point is located at a position distant from the center of the image, we have to track it for a longer time and use its average movement.

V. CONCLUSION

This work deals with the problems of how one moving omnidirectional camera locates all obstacles around the robot and of how it classifies them as stable or not. In particular, we aim to detect dynamic obstacles while the robot moves. In order to locate obstacles, floor boundary points where one omnidirectional camera can measure the distance from the robot are used. They are detected by the floor detection method using Ward's clustering to find representative colors and Mahalanobis distance to identify floor colors.

For classification, our robot tracks the floor boundary points. Comparing the robot's movement with floor boundary points' movement, our robot classifies obstacles and dynamically changes the threshold that the floor detection uses.

The first experimental result shows our floor detection method detects floor color as accurately as the GMM does and learns the floor colors more easily than the GMM does. In the second experiment, we confirm the classification ratio increases to 85% by dynamically changing the threshold of our classification method. In future work, we plan to change tracking duration based on the distance between the center of the image and the position of floor boundary points.

ACKNOWLEDGMENT

This research was supported by New Energy and Industrial Technology Development Organization (NEDO, Japan) Project for Strategic Development of Advanced Robotics Elemental Technologies, Conveyance Robot System in the Area of Service Robots, Robotic Transportation System for Commercial Facilities.

REFERENCES

[1] Z. Jia, A. Balasuriya, S. Challa, "Sensor Fusion based 3D Target Visual Tracing for Autono-mous Vehicles with IMM", IEEE International Conference on Robotics and Automation, pp.1841-1846, 2005.

[2] A.C.Murillo, J. J. Guerrero and C. Sagues: "SURF features for efficient robot localization with omnidirectional images", IEEE International Conference on Robotics and Automation, pp. 3901 - 3907, 2007.

[3] G. Silveira, E. Malis, P. Rives: "Real-time Robust Detection of Planar Regions in a Pair of Images", International Conference on Intelligent Robots and Systems, pp.49-54, 2006.

[4] H. Takeshima, T. Ida, T. Kaneko: "Extracting Object Regions Using Locally Es-timated Probability Density Functions", Proceedings of the IAPR Conference on Machine Vision Applications, 2007.

[5] J. H Ward: "Hierarchical Grouping to Optimize an Objective Function", Journal of the American Statistical Association, Vol 58, No. 301, pp. 236-244, 1963.

[6] T. Kanda, and H. Ishiguro, "Friendship estimation model for social robots to understand human relationships", IEEE International Workshop on Robot and Human Communication, pp. 539-544, 2004.