Robot Navigation by Eye Pointing

Ikuhisa Mitsugami, Norimichi Ukita, and Masatsugu Kidode

Graduate School of Information Science, Nara Institute of Science and Technology
8916–5, Takayama, Ikoma, Nara, Japan 630–0192
{ikuhi-mi, ukita, kidode}@is.naist.jp

Abstract. We present a novel wearable system for robot navigation. In this system, a user can operate multiple robots in a very intuitive way: the user gazes at a robot and then gazes at its destination on the floor. As this system needs no equipment in the environment, the user can apply it anywhere on a flat floor with only the wearable system. In this paper, we show how to estimate the positions and orientations of the robots and the gazed position. We also describe implementation of the robot navigation system.

1 Introduction

In recent years, a variety of robot systems have been developed. They are becoming more intelligent and are coming into our daily lives. There are a now numerous robot systems, and they provide a variety of functions. Among these we focus here on the function that the robot can move to a position specified by the user because this is a fundamental and very important function; whatever tasks it is to perform, we first have to navigate it to the correct position for the work. To be able to do this, we have to express the positional information and convey it to the robot.

There are many ways to express position in the real world. Nowadays the most popular way is by hand operation devices. However, when more and more robots come into our daily lives and need to be controlled more often, operation by such devices is inconvenient because we are forced to carry them continuously. If we would like to operate robots often and easily, more intuitive ways are required.

Voice recognition is one type of intuitive operation. However, operation by voice is not appropriate for robot navigation because positions in the real world are usually very hard to specify by verbal information, especially on a floor without enough landmarks.

Another way, finger pointing [1–4], which is representative of gesture recognition approaches [5], is good for robot navigation. We can indicate positions or directions in the real world intuitively and simply. However, we cannot navigate the robot while simultaneously doing another manual task. Moreover, it is not the simplest way, because before making these gestures we inevitably gaze at the specified position. Considering this, gazing, which is another type of gesture, should be the most intuitive and simple way to specify positions in the real world.

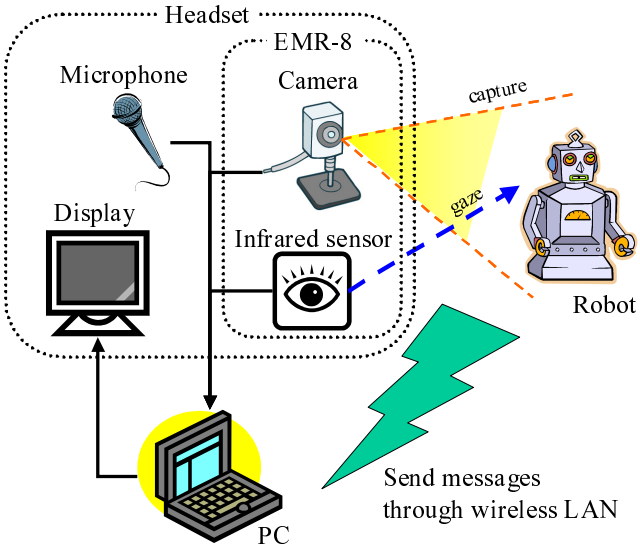

Fig. 1. System configuration

Therefore, we focus on the user's gaze information. In this paper, we describe a novel wearable system to navigate multiple robots on a floor, based on a position specification method incorporating gaze information. In this system, the user can select a robot and specify its destination merely by gazing. Every robot has a 2D square marker on it. The system detects the marker by a camera worn by the user, and then estimates the position and orientation of the robot and the gazed position on the floor. We note that the system has also the advantage that it can be used anywhere on the floor, because it needs only information obtained from a devices worn by the user. This means that the system needs no equipment in the environment around the user, such as cameras placed in the environment to detect the positions of users and robots [6, 7].

2 System Configuration

The configuration of the system is shown in Figure 1. It contains mobile robots, a PC and a headset.

The headset worn by the user is shown in Figure 2. It consists of an eye-tracker, a microphone and a small display. The eye-tracker includes a camera and an infrared sensor that detects the user's eye direction. With the eye-tracker, we can obtain the view image of the camera as well as the direction the user gazes at on the image in real time. Its mechanism and calibration methods are described in Section 3.2. The microphone is used to control the timing of the operation and to accept the user's commands. The display is for checking the

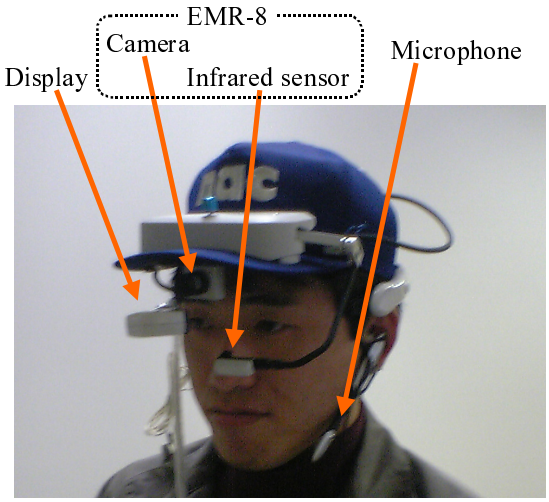

Fig. 2. Headset

estimated results and the current state of the system. It shows the user's view image captured by the camera with virtually annotated objects.[1]

The headset is connected to the PC, which processes information from the eye-tracker and the microphone and estimates the positions and orientations of the robots and the gazed position. Based on the estimated results, the PC sends operation messages to the robots. As both the PC and the mobile robots have wireless LAN interfaces, the messages are sent through a wireless network.

3 Preprocessing for Proposed System

We have to configure the camera, the eye-tracker and the robots before the operation. This configuration needs to be done only once before the first use of the system.

3.1 Camera Intrinsic Calibration

When we use camera images to understand 3D geometry of the scenes precisely, we have to calibrate the camera. From the calibration, we can obtain the distortion factors (k_1, k_2, p_1, p_2) and the intrinsic parameter matrix A. The distortion

[1] For a suitable display device, we suppose a desirable optical see-through head-mounted display (HMD) that covers the user's sight and can display virtual 3D objects onto the scene as if they were situated in the real world. However, in our current implementation, we utilize such a small display instead of the desirable HMD.

factors distort the camera images by the following equation:

$$\tilde{X} = X + (k_1 r^2 + k_2 r^4) + (2p_1 XY + p_2(r^2 + 2X^2)), \quad (1)$$
$$\tilde{Y} = Y + (k_1 r^2 + k_2 r^4) + (2p_1 XY + p_2(r^2 + 2Y^2)), \quad (2)$$

where (X, Y) is an undistorted image coordinate, and $(\tilde{X}, \tilde{Y})$ is a real distorted image captured by the camera, while $r^2 = X^2 + Y^2$. When these distortion factors are obtained, we can inversely obtain the undistorted image (X, Y) from the distorted image $(\tilde{X}, \tilde{Y})$ from the camera. The intrinsic parameter matrix A translates a 3D camera coordinate (x_c, y_c, z_c) into a 2D image coordinate (X, Y) by the following equation:

$$\begin{pmatrix} \lambda X \\ \lambda Y \\ \lambda \end{pmatrix} = A \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix}, \quad (3)$$
$$A = \begin{pmatrix} f_X & 0 & c_X & 0 \\ 0 & f_Y & c_Y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

where λ is a scale factor, (c_X, c_Y) is a coordinate of the principal point of the image, and f_X, f_X are the focal lengths by the X and Y axes.

3.2 Eye-tracker Calibration

To measure the user's eye direction, we used an EMR-8 eyemark recorder (NAC Inc.), in which the corneal reflection-pupil center method is adopted. In this method, infrared ray is emitted to the eye and its reflection is captured by the image sensor. A sample of the captured image is shown in Figure 3, and from the image, positions of the pupil center and the center of corneal reflection is detected as shown in Figure 4. Since the shape of the eye is not spherical as shown in Figure 5, the relative positions of the pupil center and the center of corneal reflection are changed according to the eye direction. By using this characteristic, the eye-tracker obtains the eye direction (u, v) in real time.

Next, to overlay the points representing the view directions onto the image observed by the view camera, correspondence between the view direction and the 2D image coordinates is needed. In the EMR-8, this correspondence is directly computed because it is difficult to obtain the relative geometric configuration of the camera and the eyeballs. To calculate the correspondence, a flat plane in the environment (e.g., a wall) is used. While the user looks toward the wall, the view camera also observes the wall. Note that the wall has to be perpendicular to the view axis of the camera. Nine points are superimposed on the observed image by the EMR-8. Their positions in the 2D image coordinates (X_i, Y_i) $(i = 0, \cdots, 8)$ are known. All the points are then projected onto the wall in the real environment, for example by a laser pointer, and the user gazes at each projected point in turn. Next, the 3D direction of each binocular view line (v_i, v_i) $(i =$

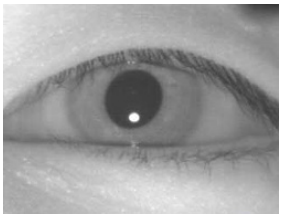

Fig. 3. Captured image of the image sensor

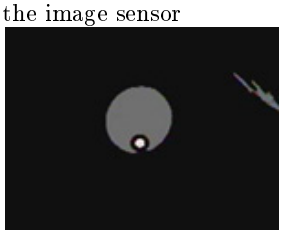

Fig. 4. Detection result of the pupil center and the center corneal reflection

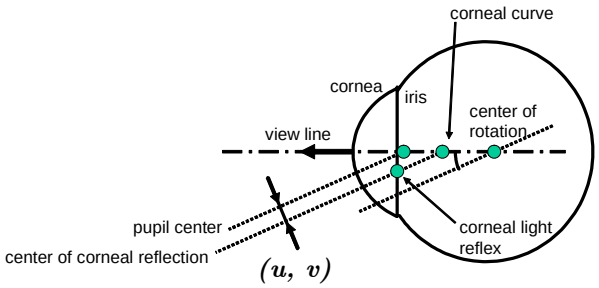

Fig. 5. Corneal reflection-pupil center method

$0, \cdots, 8$) is measured (Figure 6) by the EMR-8. These values are derived from the following equations:

$$X_i = a_0 + a_1 u_i + a_2 v_i + a_3 u_i^2 + a_4 u_i v_i + a_5 v_i^2, \quad (4)$$

$$Y_i = b_0 + b_1 u_i + b_2 v_i + b_3 u_i^2 + b_4 u_i v_i + b_5 v_i^2, \quad (5)$$

where a_i, $b_i (i = 0, \cdots, 5)$ are unknown constants. These simultaneous equations are solved to calculate a_i, b_i. After a_i, b_i are obtained, the EMR-8 is able to correctly overlay the view direction onto the camera image.

3.3 Robot Settings

In this system, a 2D square marker has to be put on every robot.[2] This system detects the marker in the camera images and by then estimates the position and orientation of the robot. As the 2D marker can be placed anywhere on the robot, every marker is at a unique position and orientation in the robot coordinate. We therefore have to obtain the position and orientation of every marker of every robot.

We next prepare a sheet on which 4 square markers and 2 crossing lines are printed, as shown in Figure 7. We place the robot at the intersection of the lines

[2] We can use more markers if the markers are too small to estimate the position and orientation precisely, or if the camera often fails to observe the marker because it is badly positioned or oriented. When we use more than one marker for a robot, we need to estimate the positions and orientations of each one.

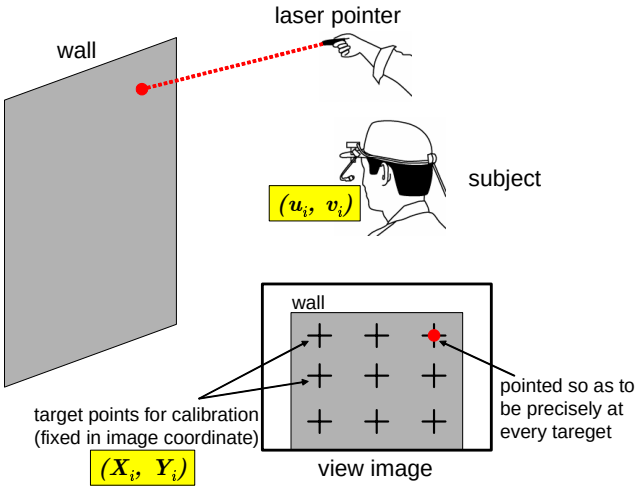

Fig. 6. Calibration operation of EMR-8

on the sheet; this is the origin of the robot coordinate, which corresponds to the position of the robot. The lines represent the x_r and y_r axes of the robot coordinate system. We define x_r as the orientation of the robot, and we direct the robot's face along it. The z_r axis is defined as the line perpendicular to the x_r-y_r plane, and the robot usually stands along the z_r axis.

We direct the camera to the robot on the sheet so that the camera image should capture not only the 4 markers but also the marker on the robot. Figure 8 is a camera image of this situation. As the positions and orientations of the markers are estimated by their appearance, we can obtain a matrix Q_{CR} that consists of a rotation matrix R_{CR} and a translation vector t_{CR}, which transforms the camera coordinate (x_c, y_c, z_c) to the robot coordinate (x_r, y_r, z_r) by the following equation:

$$\begin{pmatrix} x_r \\ y_r \\ z_r \\ 1 \end{pmatrix} = Q_{CR} \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix}, \tag{6}$$

$$Q_{CR} = \begin{pmatrix} R_{CR} & \boldsymbol{t}_{CR} \\ \mathbf{0}^T & 1 \end{pmatrix}.$$

Some of the 4 markers may be undetectable because of occlusion by the robot (for example, marker "B" is occluded in Figure 8). Even in such cases, we can still estimate the transformation matrix Q_{CR} using the other detectable markers. Next, in the same way, by the appearance of the marker on the robot we can also obtain the transformation matrix Q_{CM} that transforms the camera coordinate

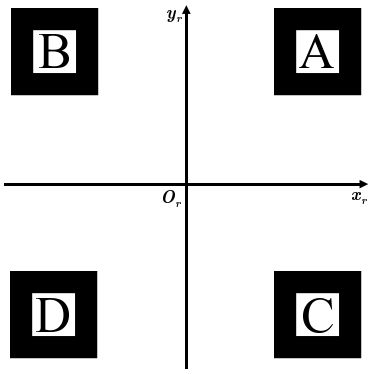

Fig. 7. Sheet with 4 markers

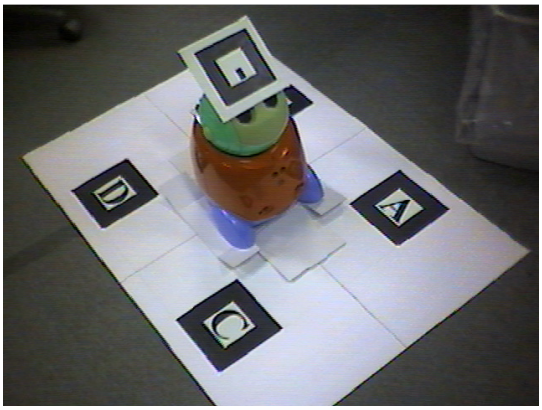

Fig. 8. Captured image of the sheet and the robot

(x_c, y_c, z_c) to the marker coordinate (x_m, y_m, z_m) by the following equation:

$$\begin{pmatrix} x_m \\ y_m \\ z_m \\ 1 \end{pmatrix} = Q_{CM} \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix}. \tag{7}$$

By Equation (6) and (7), the following equation is obtained:

$$\begin{pmatrix} x_r \\ y_r \\ z_r \\ 1 \end{pmatrix} = Q_{MR} \begin{pmatrix} x_m \\ y_m \\ z_m \\ 1 \end{pmatrix}, \tag{8}$$

where

$$Q_{MR} = Q_{CR} Q_{CM}^{-1} \tag{9}$$

is the matrix that transforms the marker coordinate (x_m, y_m, z_m) to the robot coordinate (x_r, y_r, z_r). Because the marker is fixed on the robot, this transformation matrix Q_{MR} is constant. The system stores Q_{MR} for every marker.

We note that the design of each marker should be different, because the marker is used not only for estimation of the position and orientation but also for identification of the robot.[3]

4 Estimation of Robot Position and Gazed Position

When we operate the robot, the system works by estimating the information in the order shown below.

[3] If a robot has multiple markers, each of its markers has to be different.

4.1 Marker's Position and Orientation

The system cannot work without the information of the robot's position and orientation. We orient the camera so that the camera can observe the marker on the robot. By the method described in Section3.3, we can estimate the position and orientation of the marker as a matrix P_{MC} that transforms the marker coordinate (x_m, y_m, z_m) to the camera coordinate (x_c, y_c, z_c):

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = P_{MC} \begin{pmatrix} x_m \\ y_m \\ z_m \\ 1 \end{pmatrix}. \tag{10}$$

4.2 Robot Coordinate System

By Equations (8) and (10), we obtain the following equation:

$$\begin{pmatrix} x_r \\ y_r \\ z_r \\ 1 \end{pmatrix} = P_{CR} \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix}, \tag{11}$$

where

$$P_{CR} = Q_{MR} P_{MC}^{-1}. \tag{12}$$

As the Q_{MR} is constant and P_{MC} has been obtained, we can obtain the transformation matrix P_{CR} between the camera coordinate and the robot coordinate.

As shown in Equation (13), P_{CR} consists of a rotation matrix R_{CR} and a translation vector t_{CR}, which are the orientation and position of the camera in the robot coordinate respectively:

$$P_{CR} = \begin{pmatrix} R_{CR} & \boldsymbol{t}_{CR} \\ \mathbf{0}^T & 1 \end{pmatrix}. \tag{13}$$

4.3 Gazed Position in Robot Coordinate System

Using the EMR-8, we can obtain the gazed position on the camera image. This position indicates a line l corresponding to the user's gaze direction. We assume that the user looks at positions not above the floor but only at positions on the floor that the robot moves on. Considering the definition of the robot coordinate in Section.3.3, the floor corresponds to the x_r-y_r plane. The gazed position can thus be calculated as the intersection of the line l and the x_r-y_r plane. By Equation (4) and (11), we can obtain the following equation:

$$\begin{pmatrix} \lambda X^{(gazed)} \\ \lambda Y^{(gazed)} \\ \lambda \end{pmatrix} = A P_{CR}^{-1} \begin{pmatrix} x_r^{(gazed)} \\ y_r^{(gazed)} \\ z_r^{(gazed)} \\ 1 \end{pmatrix}, \tag{14}$$

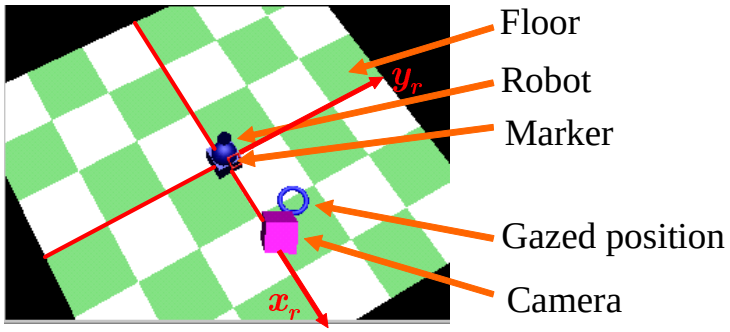

Fig. 9. Position estimation of robot, camera, floor and gazed position

where $(X^{(gazed)}, Y^{(gazed)})$ denotes the gazed position on the camera image that can be obtained by the EMR-8, and $(x_r^{(gazed)}, y_r^{(gazed)}, z_r^{(gazed)})$ denotes the gazed position. As the gazed position is on the x_r-y_r plane, $z_r^{(gazed)}$ must be 0. With this equation, we can calculate $(x_r^{(gazed)}, y_r^{(gazed)})$, which corresponds to the relative gazed position from the robot.

4.4 Estimation Results

Figure 9 shows the virtual 3D space generated from the estimation results. The robot is at the center on the floor, and the camera and the gazed positions move in space corresponding to their positions in the real world.

5 Implementation of Robot Navigation System

5.1 Implementation Environment

We implemented the robot navigation system using a Windows 2000 PC with an Intel Pentium4 2.8GHz CPU and 1024MB memory. ARToolKit [8] helps the system to detect markers from camera images and to estimate 3D position and orientation in real time, and OpenGL [9] is used to annotate virtual objects on the display images. We also use the user's voice, which is desirable for selecting and conveying the action of the robot. For voice recognition, we utilize Julius [10], a well-known recognition engine for Japanese.

5.2 Diagram of Robot Navigation System

We show the state transition diagram of the system in Figure 10. There are 5 states (A)...(E) in the diagram.

At the beginning, the current state is (A). Here, the system searches for markers in the camera image, and if a marker is detected it estimates the robot

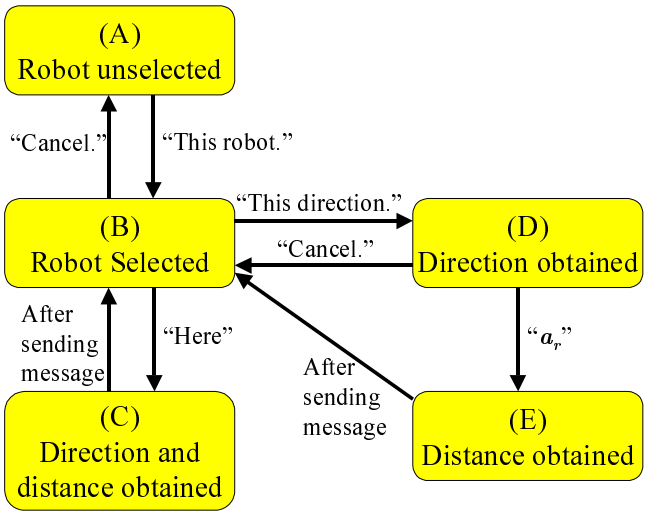

Fig. 10. State transition diagram of the system

coordinate and identifies the robot by the marker. The user can see a torus at the gazed position on the floor and a red cone above the robot, which means that the robot is detected but not selected. The display image is shown in Figure 11.

If the user says "KORE" ("this one" in Japanese) while gazing at the robot, the current state turns to (B). In (B), the user has selected a robot to operate. Here the user can still see the torus at the gazed position and the cone above the robot turns blue, which means that this robot is now selected. The user can also see the distance ($= \sqrt{x_r^2 + y_r^2}$) and the angle ($= \tan^{-1} \frac{y_r}{x_r}$) above the torus, which are calculated in Section4.3, as shown in Figure 12. If the user says "YAME" (means "cancel") the current state returns to (A).

Next, if the user says "KOKO" ("here"), the current state turns to (C). The system regards the gazed position (x_r, y_r) as the destination of the robot and sends an operation message "rotate by $\tan^{-1} \frac{y_r}{x_r}$" and "go straight $\sqrt{x_r^2 + y_r^2}$" to it. After sending it, the current state returns to (B).

The transition between states above is the fundamental behavior of the system. It provides simple and intuitive operation of the robot. However, it has the disadvantage that the robot can move only in a restricted area by one operation because the gazed position can be estimated only when the camera captures both the robot and the gazed position. Here we prepare a different transition path (B)-(D)-(E) in the diagram to overcome this disadvantage.

To make the current state switch from (B) to (D), the user says "KOTCHI" ("this direction"). In (D), the system regards the gazed position (x_r, y_r) not as the destination itself but as the direction of the destination. The system sends an operation message "rotate by $\tan^{-1} \frac{y_r}{x_r}$" to the robot. After sending it, the

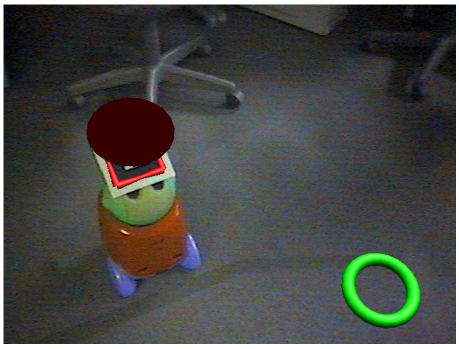

Fig. 11. Display image in State (A)

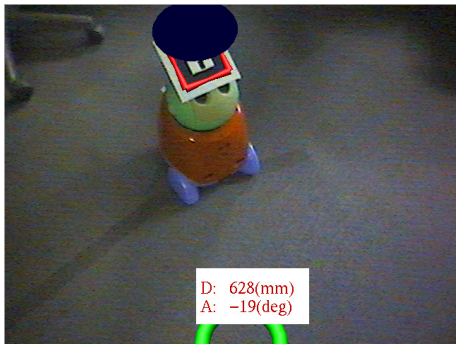

Fig. 12. Display image in State (B)

current state turns to (E), where the system waits for information about the distance to go straight. When the user says the distance, for example "a_r", the system sends an operation message "go straight a_r" to the robot. After sending, the current state returns to (B).

In the diagram there are also other paths for canceling. Using these paths, the system can continue to work without stopping.

6 Conclusions

In this paper, we have presented a novel wearable system for robot navigation on a floor. The user can operate multiple robots very intuitively, because he/she can select a robot to operate and specify its destination on the floor by only his/her gazing. This system has the further advantage that we can use it anywhere on a flat floor, because the positions and orientations of robots and the gazed positions are estimated only by information from a wearable headset.

Future work will include improvements to the system, including introduction of a better HMD and vocabulary addition of voice commands. Evaluation of its usability will also be included. Moreover, quantitative evaluation of the estimation accuracy of position and orientation is another important topic.

Acknowledgments

This research is supported by Core Research for Evolutional Science and Technology (CREST) Program "Advanced Media Technology for Everyday Living" of Japan Science and Technology Agency (JST).

References

1. R.Kahn, M.Swain, P.Prokopowicz and R.Firby: "Gesture recognition using the Perseus architecture," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pp.734–741, 1996.

2. K.Nickel and U.Karlsruhe: "Pointing gesture recognition based on 3D-tracking of face, hands and head orientation," Proc. of the 5th International Conference on Multimodal Interfaces, pp.140–146, 2003.
3. R. Cipolla and H. J. Hollinghurst: "Human-robot interface by pointing with uncalibrated stereo vision", Image and Vision Computing, Vol.14, No.3, pp.178–178, 1996.
4. Nebojsa Jojic, Thomas S. Huang, Barry Brumitt, Brian Meyers, Steve Harris: "Detection and Estimation of Pointing Gestures in Dense Disparity Maps," FG 2000, pp.468-475, 2000.
5. Thomas B. Moeslund and Erik Granum: "A Survey of Computer Vision-Based Human Motion Capture," Computer Vision and Image Understanding, Vol.81, No.3, pp.231–268, 2001.
6. S.Stillman, R.Tanawongsuwan and I.Essa: "A system for tracking and recognizing multiple people with multiple cameras," Technical Report GIT-GVU-98-25, Georgia Institute of Technology, Graphics, Visualization, and Usability Center, 1998.
7. M.Sakata, Y.Yasumuro, M.Imura, Y.Manabe and K.Chihara: "A Location Awareness System using Wide-angle Camera and Active IR-Tag," Proc. of IAPR Workshop on Machine Vision Applications, pp.522–525, 2002.
8. "ARToolKit," http://artoolkit.sourceforge.net/.
9. "OpenGL Library," http://www.opengl.org/.
10. "Julius – open source real-time large vocabulary speech recognition engine," http://julius.sourceforge.jp/.