# A Simple Interface for Mobile Robot Equipped with Single Camera using Motion Stereo Vision

Stephen Karungaru, Atsushi Ishitani, Takuya Shiraishi, and Minoru Fukumi

***Abstract*—Recently, robot technology has gained popularity because of labor shortage, ability to work for long hours, etc. Remote-controlled robots are especially being put to practical use for several purposes such as conveying and security systems, etc. This is because their safety is higher than autonomous robots in addition to their easier development. However, remote-controlled robots require a user interface that can control the robot actions and show information captured by robot to the users. In this work, we develop a simple interface for mobile robots. In robot actions, movement of the robot itself is most important. To use this interface, the users need only point to the position on a screen where they want the robot to move to. After that, the interface controls the robot movement between the current and the target positions autonomously. Therefore, to move the robot to the selected point, the robot needs to calculate the target's 3D location. To calculate the 3D location, we use motion stereo vision that uses a single camera. Using this method, the interface can be applied to many robot systems. In this paper, we proved the effectiveness of the proposed method by performing experiments in real environments.**

***Index Terms*—Robot interface, motion stereo, single camera.**

## I. Introduction

Recently, labor shortage due to the low birthrate and aging population has become a major social problem in Japan. In addition, Japanese industrial technologies have been declining because of competition from countries with low-cost labor. Robot technology is one of the methods that could solve the problems because the robots can replace the workforce at low cost. This idea has been discussed for about 10 years. However, the robots have not yet become practical due to several obstacles. Of these problems, robot productivity is the main one. Currently, research institutes and companies are making robots from scratch. However, many of the processes are unnecessary. The common functions such as communication and motor control should be reused as function modules. Although RT (Robot Technology) middle-ware [1], [2] which makes it easy to develop robots is proposed and developed, it is seldom used because of low-feasibility, few supported hardware, etc. Moreover, the problems of safety, including whether the robots are safe around humans, are a major concern. However, although there are many obstacles in the spread of the robots, they are certainly needed. A remote controlled robot is one of the robots with comparatively high safety. They are commonly used for conveying, security, rescue, etc. They demand a user interface which receives images from the robot and sends orders from the user to the robot. Currently, various devices and methods are used. However, they have several problems. For example, in a control method which sends commands such as FOWARD, ROTATION and STOP, it cannot control the minute movement of the robot. Additionally, another method using game controllers is difficult for new game users. A method using display of the images sent from the robot, directly displaying only the images is common. However, in this method, it is difficult for the user to get the depth of the objects. Moreover, if the distance to the object is too short, the user cannot see it. To solve the problem, an interface that shows the information of range sensors together with the image and one that shows overhead view from another camera mounted above are proposed. However, these methods are expensive and lack versatility.

Based on these reasons, in this work, we propose an easy-to-use interface that can be applied to many robots. Although various kinds of actuators are used on robots, almost all robots have mobile actuators. To control the movement of the robot easily, the interface should be intuitive. For intuitive control, the robot should be controlled by a single instruction from the user indicating the point where they want the robot to move to. For example, the target could be an object which the user wants to carry in the conveying robot. The only task required from the users is a single identification of the target on the image sent from the robot. Therefore, the movement of the robot should be autonomous. For autonomous movement, the robot needs to know the target's 3D location. To reconstruct 3D position, stereo cameras and range sensors are generally used [3], [4], and [5]. However, some robots which do not have these sensors also exist. In this work, we are aiming to develop the interface with versatility. Therefore, the robots equipped with single camera and odometry are targeted. 3D reconstruction method using a single camera has been proposed. Especially, motion stereo vision is often used [6], [7], and [8]. This is a method that reconstructs 3D location by same fundamental methods such as stereo vision, after capturing images by camera motion. In stereo vision, the longer the base-line length, the more the reconstruction accuracy is improved. Motion stereo vision can make the base-line longer, because base-line is the movement of the camera. Therefore, it can calculate 3D position with higher accuracy. However, odometry (movement calculation method using rotation of wheels) data often includes some error due to tire slips. In that case, motion stereo vision cannot reconstruct 3D position accurately. We consider that motion stereo vision is suitable for wheeled mobile robots, and use the method. Some previous studies using motion stereo vision already exist.

Manuscript received July 10, 2012; revised August 29, 2012.

The authors are with the graduate School of Advanced Technology and Science, The University of Tokushima, Japan. (e-mail: karunga@is.tokushima-u.ac.jp).

However, they have set conditions such as the following.

- Experiments are only simulations.
- Off-line processing due to high processing cost using high-resolution image.
- Unnecessary information such as background and floor is removed in advance.

Additionally, in camera movement, gaze movement is avoided, because reconstruction accuracy is reduced. In our proposed interface, the robot moves forward after facing the target. Thus, the possibility that gaze movement will occur depending on the angle of the camera is high. From this reason, we propose a method that can perform at high accuracy even in gaze movement, and confirm the effectiveness using experiments. Moreover, 3D positions are calculated on-line in unknown environments.

## II. Structure of Interface

### A. Supported Robot Systems and Environments

Fig. 1 shows an example of the structure of the remote-control robot system. The robot and the control systems need not be at the same location. The user side terminal and the robot are connected by a network. The robot receives surrounding information from equipped sensors, and sends it to the terminal. The user can see this information on a screen, and if the user wants to move the robot, the user sends a command. Then, the robot should proceed depending on it. In this work, we assume that the terminal consists of a display and mouse for PCs. Instead of these devices, we can also allow the user to use touch panels. We also suppose that the robot has a single camera and an odometry system. Odometry is a method using rotation angles of tires (motors) to calculate self-location. As the odometry devices, rotary encoders, servomotors and stepping motors are usually used.

In this work, the targets are static and unknown objects having texture. Currently, the robot is designed to move on flat floors. There should also be enough illumination which enables capture of the images by the camera.

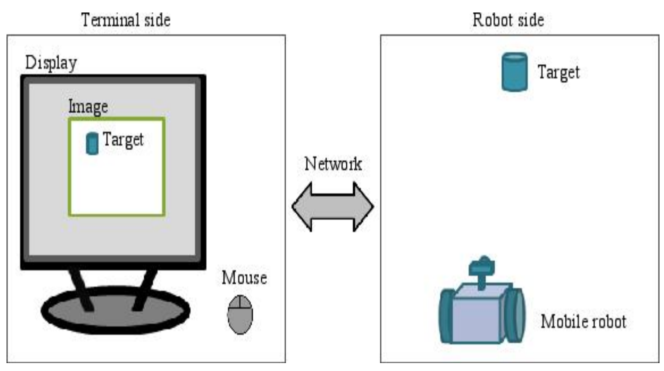

Fig. 1. Structure of remote-control robot system

### B. Movement Algorithm

The interface can control the robot movement by only the instruction about the target. Fig. 2 shows the algorithm used by the robot to reach the target point from the target instruction.

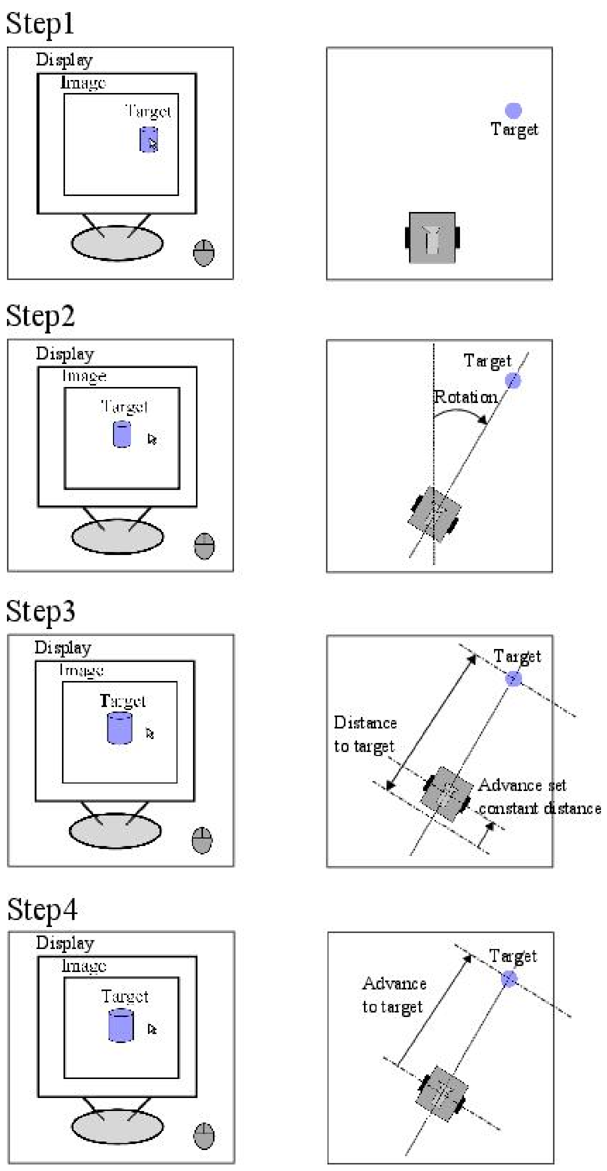

Fig. 2. Flow of robot movement

- [Step 1]The users can see the images sent from the robot on a display. They can then point to a position to which they want the robot to move to.
- [Step 2]The interface calculates the direction to the target, and turns the robot to face to the target. The instruction of the target is only once, therefore, the interface tracks the target on the images to keep target location.
- [Step 3]The robot advances a set constant distance, and the interface calculates the distance to the target.
- [Step 4]The robot moves to the target position.

In this way, there are no meaningless movements because we separate the rotation and forward movement operations.

## III. 3D Position Calculation

### A. Flow of Algorithm

Fig. 3 shows the flow of process in the proposed interface. We will discuss each method in detail later.

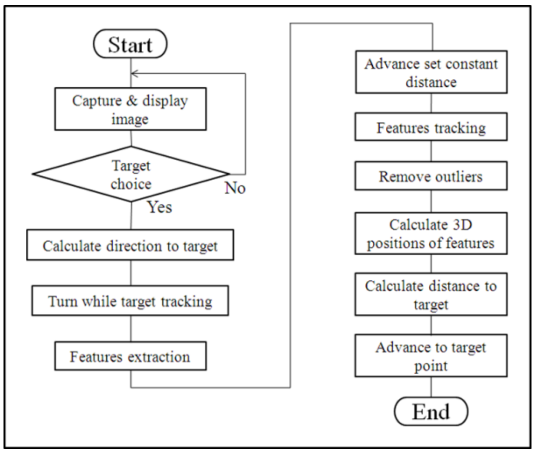
Start
Capture & display image
Target choice
No
Yes
Calculate direction to target
Turn while target tracking
Features extraction
Advance set constant distance
Features tracking
Remove outliers
Calculate 3D positions of features
Calculate distance to target
Advance to target point
End

Fig. 3. Flow of algorithm

## B. Direction Calculation

Fig. 4 shows positions relationship between the target and the camera during target instruction in a pinhole camera model. Horizontal axis shows the $X$ axis, vertical axis shows the $Z$ axis. $s_x$ is physical size of image sensor on vertical direction.

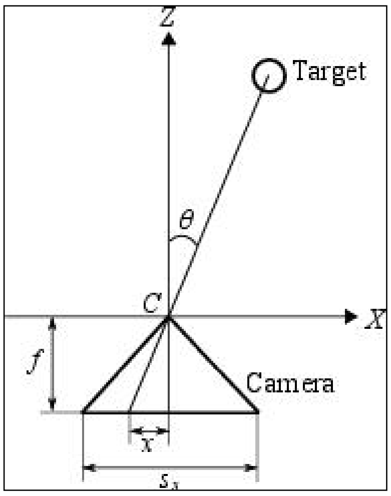

Fig. 4. Pinhole camera model in direction calculation

The angle $\theta$ to the target based on the optical axis is calculated by eq. 1.

$$\theta = arctan\frac{x}{f} \tag{1}$$

where, $x$ is the value of normalized image coordinate, $f$ is focal length.

Therefore, the interface calculates the target direction and rotates the robot by the following algorithm.

1. The image coordinate $(u, v)$ clicked by the user is set as target center, and it is transformed to normalized image coordinate $(x, y)$.
2. The direction $\theta$ is calculated from its coordinate x and eq. 1.
3. The interface rotates the robot $\theta$ degrees to face to the target. At this time the interface must keep track of the target on the images, because the target instruction is only once. The tracking method used is the KLT-Tracker algorithm.
4. After rotation, the direction is re-calculated to confirm whether the robot rotated accurately. If the robot could not rotate exactly, the robot adjusts itself.

## C. Distance Calculation

The distance to the target point is calculated by motion stereo vision. The algorithm is as follows.

[Feature Point Extraction]

First, feature points are extracted by KLT corner detector using the image (Image 1) captured after rotation.

[Feature Point Tracking]

Secondly, the robot advances a set constant distance, and captures Image 2. On the Image 2, the feature points are tracked using a KLT-Tracker.

[Outlier Removal]

The tracking of the feature points sometimes failed, and these points cannot reconstruct the 3D location exactly. Therefore, we should detect and remove these points. This algorithm is as follows.

- If optical flow (movement vector of feature point on image before and after motion) is too short or long, the feature point is removed as a failed tracking point.
- A straight line ($y=ax+b$) is calculated.
- Repeat 1 and 2 for every feature points.
- Average coordinate of intersection points for each optical flow is calculated as disappearance point (Fig. 5).
- Distances between the disappearance point and the straight lines $d$ are calculated (Fig. 6). If $d$ is larger than threshold value, the feature point is removed as outlier.

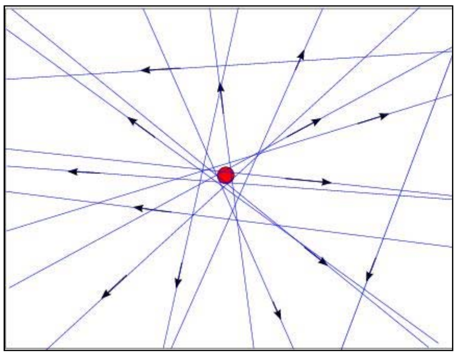

Fig. 5. Disappearance point

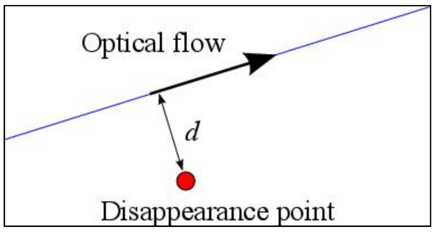

Fig. 6. Remove outlier

[3D Position Calculation of Feature Point]

3D locations are calculated from the correspondence coordinates of the feature points that were successfully tracked. In this step, eq. 2 is used.

$$C = \|X - s_1 X\widetilde{m_1}\|^2 + \|X - s_2 R\widetilde{m_2} - t\|^2 \qquad (2)$$

where, $\widetilde{m_1}$ and $\widetilde{m_2}$ are extended vectors of the projected point, and $s_1$, $s_2$ are scalar expressing the vector length.

[Calculation of Distance to Target Point]

The distance to the target is calculated from the 3D position of the feature points. In this method, we assume that feature points of the target are located in constant distance from the pointed coordinate, and a median of these 3D positions $Z$ is the distance to the target.

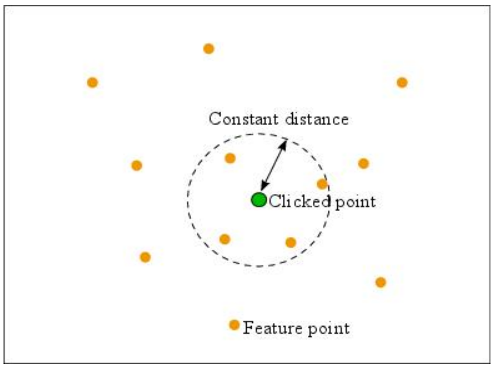

Fig. 7. Calculation of distance to target

## IV. Experiments

Before mounting the proposed interface on the robot system, its effectiveness is confirmed. First, the direction is calculated and then the distance calculation is performed.

### *A. Common Conditions*

In this work, captured image size is 640x480 pixels and frame rate is 15 fps. Additionally, intrinsic parameters are pre-calculated by camera calibration. Moreover, information about background is unknown and projected scene is a real environment.

### *B. Direction Calculations*

[Conditions]

The direction to the target is approximated to within one degree, because the robots with higher precision are few. Additionally, the calculation area is set to 15 degrees right and left of the optical axis. This is because in many cases that horizontal angle of view of standard cameras is about 40 degrees.

[Results]

The experimental results are shown in Table 1. The directions are calculated in 5 degrees steps. From the result, it can be confirmed that the direction could be calculated accurately using the proposed method.

### *C. Distance Calculations*

[Conditions]

In this experiment, the camera captures the images while being manually moved. This situation is shown as Fig. 8. The target is a box put on the floor about 1000 mm away from the first camera position. We set 300 mm as the motion. Actual captured images are shown as Fig. 9 and 10.

TABLE I: Results of Direction Calculation

| True value [mm] | Calculated value [mm] |
|---|---|
| 15 | 15 |
| 10 | 10 |
| 5 | 5 |
| 0 | 0 |
| -5 | -5 |
| -10 | -10 |
| -15 | -15 |

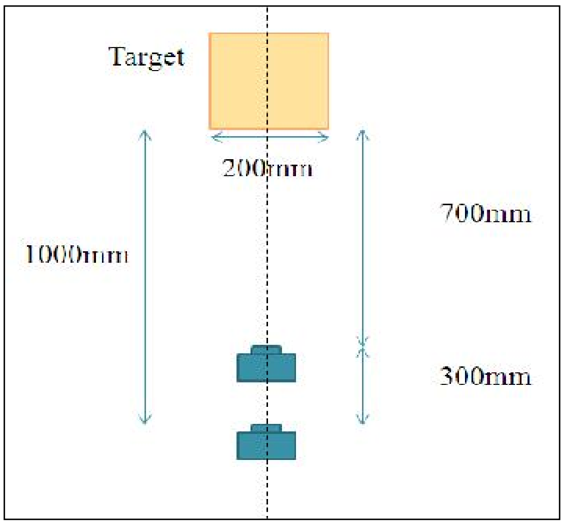

Fig. 8. Situation of experiment

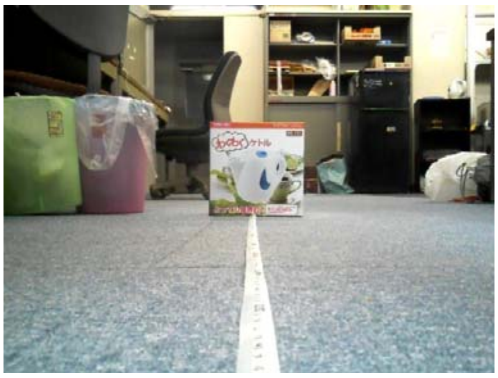

Fig. 9. Image 1 (before motion)

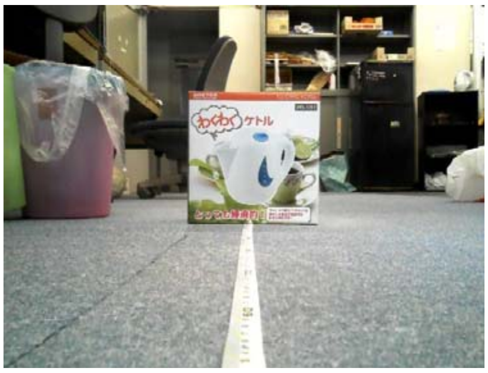

Fig. 10. Image 2 (after motion)

[Feature Points Extraction]

From image 1, the effective feature points to track are extracted using KLT corner detector. The results are shown in Fig. 11. Red points are the extracted feature points.

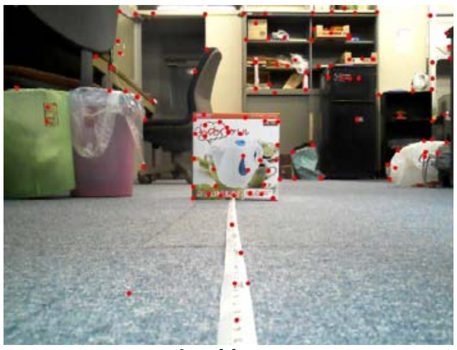
Fig. 11. Results of features extraction

[Feature Points Tracking]

After motion, the feature points extracted in image 1 are tracked in image 2. The results are shown as Fig. 12. Red points are the extracted feature point locations. Aqua points are the tracked feature point locations. Red lines show the optical flow.

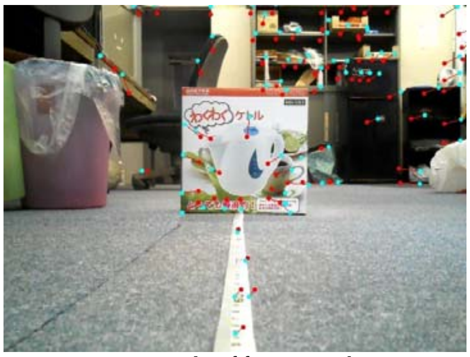
Fig. 12. Results of features tracking

[Removal of Outliers]

The results of the tracking contain some false tracked feature points. Therefore, we detected and removed these feature points as shown Fig. 13. The yellow circle is the average point of the intersections, and white circle is a set constant distance from it. We set 50 pixels as the constant distance in this experiment. As shown, green lines do not pass through the set circle and represents failed tracking. Conversely, red lines are successfully tracked. The feature points inside of white circle are the target feature points.

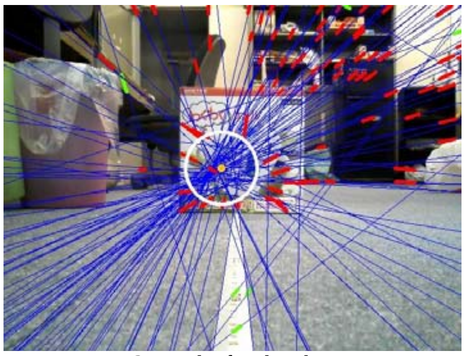
Fig. 13. Result of outlier detection

[3D Position Calculations of Feature Points]

Fig. 14 shows the results of 3D reconstruction before removing outliers. In addition, the results after that are shown in Fig. 15. Fig. 16 shows the 3D location of the feature points with real objects location marked in. Horizontal axis represents X-axis, and vertical axis represents Z-axis.

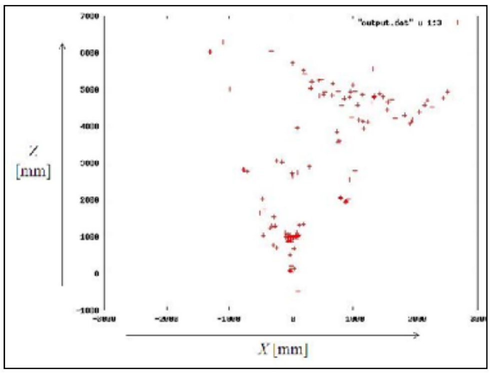

Fig. 14. Results of 3D feature positions (before remove outlier)

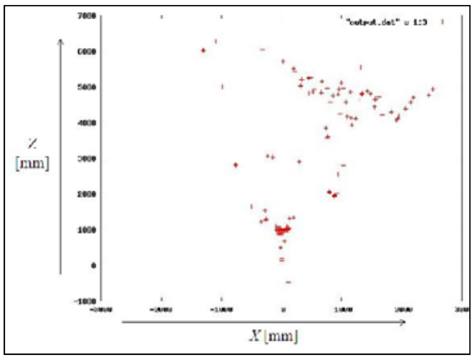

Fig. 15. Results of 3D feature positions (after remove outlier)

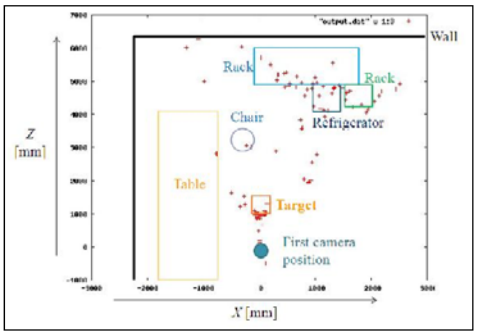

Fig. 16. Results of 3D position with real objects

[Calculation of Distance to Target Point]

The median of the feature point locations Z in the target area (blue circle) was 991mm (Fig. 17). This is the distance to the target. The true value of the distance is 1000mm; therefore, calculation accuracy is 99.1%.

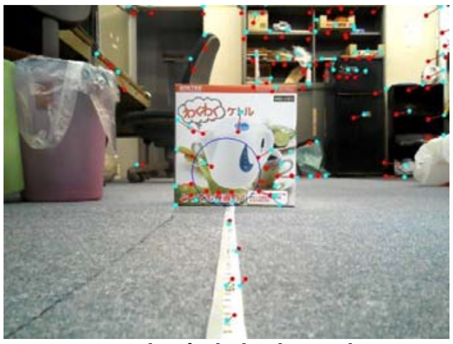
Fig. 17. Results of calculated target distance

[Consideration]

The feature points that could not be correctly tracked by the KLT-Tracker can be removed using the proposed outlier's removal method. The 3D positions of these points are also calculated with high accuracy. Additionally, although the 3D positions Z in the target area were variable, the distance to the target can be calculated with high accuracy using the median value. Therefore, from these experiments, the effectiveness of the proposed method is confirmed in a real environment. However, when the interface is mounted on the robot, it is supposed that the motion amount (camera movement value) has some errors due to tires slip. Moreover, we set the target area to a constant distance from the pointed coordinate. It is possible that the median is 3D position outside of target such as background, etc. Conversely, if the target area is small, there might be no feature points detected. Thus, we need to mount the robot system taking into account these considerations.

## V. Mounting on Robot System

In the previous experiments, two images are captured from the camera by hand, and the effectiveness of the proposed method in each step was confirmed for feature points extraction, tracking, 3D position reconstruction, etc. In this section, we investigate the effectiveness of these methods using actual robot system.

### A. Robot System

Fig. 18 is the robot used in this experiment. The robot can connect with the terminal by serial communication, and the commands from terminal side (interface) to the robot are sent using a Bluetooth connection. As the camera mounted on the robot, we use a web camera. The captured images are sent to the terminal side at 15fps, and these are processed in the terminal side at real-time.

As the mobile actuator, the robot has two stepping motors left and right side. The stepping motors are rotated by a driving pulse; therefore, it does not need a feedback circuit.

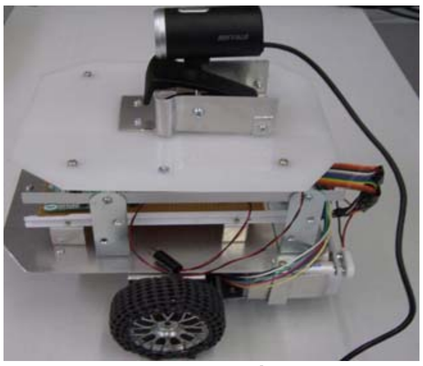
Fig. 18. Our robot

### B. Experiments

The target conditions of this experiment are the same before. The floor is a carpet. Table 2 shows the results of the distance calculation to the target for 10 trials. In addition, the detailed results of one trial are as shown in Fig. 19 to 23. In the figures of the 3D position reconstruction, horizontal axis is *X-axis*, and vertical axis is *Z-axis*.

TABLE II: Results of Direction Calculation

| Trial | Calculated Distance [mm] | Error [mm] |
|---|---|---|
| 1 | 885.0 | -115.0 |
| 2 | 856.0 | -144.4 |
| 3 | 1014.8 | 14.8 |
| 4 | 955.0 | -45.0 |
| 5 | 1024.9 | 24.9 |
| 6 | 1091.5 | 91.5 |
| 7 | 1024.5 | 24.5 |
| 8 | 753.3 | -246.7 |
| 9 | 742.1 | -257.9 |
| 10 | 871.5 | -128.5 |

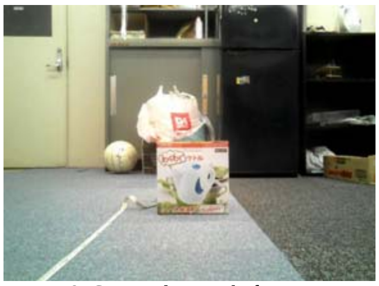
Fig. 19. Captured image before motion

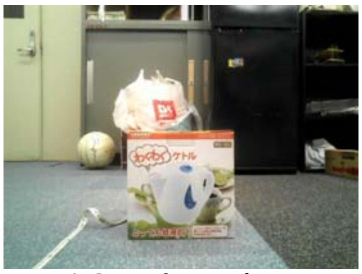
Fig. 20. Captured image after motion

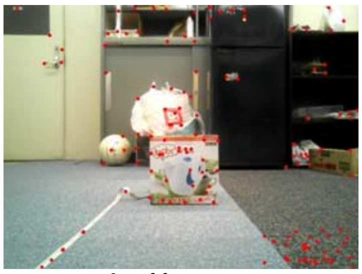
Fig. 21. Results of feature point extraction

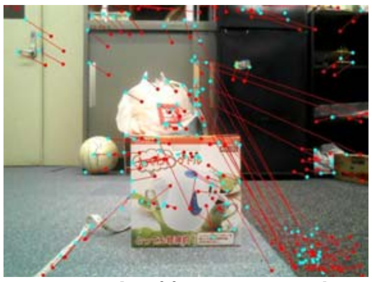
Fig. 22. Results of feature points tracking

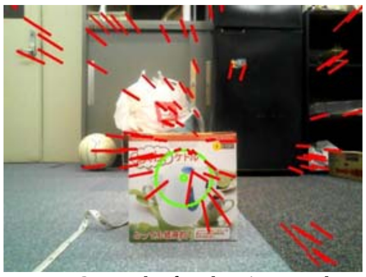
Fig. 23. Result of outliers' removal

### C. Consideration

From the results, we recognized that the distance can be calculated with high accuracy in some cases. However, in other cases, the calculation accuracy is low. The causes are thought to be as follows. The first cause is the effect of the gaze movement (movement advancing toward one certain point). In the stereo vision, reconstruction accuracy in the case of the camera set at lateral direction from objects side view is better than front-back direction, because of the influence of the errors in capturing using image sensor. Therefore, the results were influenced by gaze movement. However, we consider that the effect is reduced if we use the median of Z value as the distance to the target. The second cause is a problem in the outlier's removal method. In the experiments, the outliers are removed using the distance between disappearance point and lines passing through the optical flow. However, even if the tracking failed, the feature points near disappearance point cannot be removed by the method. Thus, we need to add to the method, information such as a direction of the optical flow. The third cause is a problem of the robot motion. At the beginning, the robot rotates just a little on circumjacent Y-axis. In this case, disappearance point is out of alignment on *u* axis of the image (Fig. 23). In the calculation, we assume that these rotations did not occur. Therefore, we think that just a little rotation affected the reconstruction accuracy. To solve the problem, a method which calculates the out of alignment of the motion from the disappearance point could be effective.

## VI. Conclusion

In this paper, we proposed a simple interface, completely easy to use and versatile for mobile robots equipped with a single camera. Instead of the user, the interface can move the robot to the user selected target position autonomously. To achieve this, the interface needs to know the 3D location of the target, and in this work, we proposed its calculation using a motion stereo vision method. We then confirmed the effectiveness of the method by the experiments, and constructed the interface based on the method. Finally, we mounted the interface on the robot system, and performed the experiments in a real environment. From the results, although some problems still exist, we recognized that the proposed interface can be used in real environments.

As future works, the following three items will be considered. First, the solution for the motion error using the disappearance point should be introduced. Secondly, the target area should be optimized using a target recognition method. Finally, we should confirm the proposed interface in other environments such as various kinds of targets, cameras and robots. Additionally, if the interface is combined with existing technology, its application can be expanded. For example, the operability of the remote-controlled robots can be improved. Moreover, in this paper the object which has texture (can extract feature points from target on image) is used as the target. To introduce a technology for floor detection, the floor with no texture can be used as the target.

## References

[1] A. Ikezoe, K. Muranaga, H. Nakamoto, and M. Nagase, "Development of RT-Middleware for Real-Time OS," *The Japan Society of Mechanical Engineers*, 2008.

[2] A. Ikezoe, H. Nakamoto, and M. Nagase, "OpenRT Platform/RT-Middleware for VxWorks," *The Japan Society of Mechanical Engineers*, 2010.

[3] R. Hartley, R. Gupta and T. Chang, "Stereo from Uncalibrated Cameras," *CVPR*, pp. 761-764, 1992.

[4] K. Kanatani and H. Mishima, "3-D Reconstruction from Two Uncalibrated Views and Its Reliability Evaluation," *Transactions of Information Processing Society of Japan*, vol. 42, no. SIG6 (CVIM 2), pp. 1-8, 2001.

[5] K. Yamada, Y. Kanazawa, K. Kanatani and Y. Sugaya, "Latest Algorithm for 3-D Reconstruction from Two Views," *Information Processing Society of Japan*, CVIM 2009-CVIM-168 (15), pp. 1-8, 2009.

[6] K. Yamazaki, H. Yaguchi, N. Hatao, Z. Nikolaus and M. Inaba, "MonoSLAM in indoor environment by wide-angle Camera," *JSME Conference on Robotics and Mechatronics*, 2P1-G13, 2008.

[7] K. Yamazaki, M. Tomono, T. Tsubouchi and S. Yuta, "Object Shape Reconstruction and Pose Estimation by a Camera Mounted on a Mobile Robot," *Intelligent Robots and Systems*, vol. 4, pp. 4019-4025, 2004.

[8] A. Yamashita, T. Harada, R. Kawanishi and T. Kaneko, "Three-Dimensional Environment Modeling from Images Acquired with an Omni-Directional Camera Mounted on a Mobile Robot by Structure from Motion," *Transactions of the Japan Society of Mechanical Enginners*, vol. C 73, no. 726, pp. 512-519, 2007, in Japanese.

**Stephen Karungaru** received a PhD in Information System Design from the Department of information science and Intelligent Systems, University of Tokushima in 2004. He is currently an associate professor in the same department. His research interests include pattern recognition, neural networks, evolutionary computation, image processing and robotics. He is a member of IACSIT, RISP, IEEE and IEEJ.

**Atsushi Ishitani** received a master degree from the Department of Information Science and Intelligent Systems, University of Tokushima in 2012.

**Takuya Ishiatani** received a master degree from the Department of Information Science and Intelligent Systems, University of Tokushima in 2012.

**Minoru Fukumi** received the B.E. and M.E. degrees from the University of Tokushima, in 1984 and 1987, respectively, and the doctor degree from Kyoto University in 1996. Since 1987, he has been with the Department of Information Science and Intelligent Systems, University of Tokushima. In 2005, he became a Professor in the same department. He received the best paper awards from the SICE in 1995 and Research Institute of Signal Processing in 2011 in Japan, and best paper awards from some international conferences. His research interests include neural networks, evolutionary algorithms, image processing and human sensing. He is a member of the IEEE, SICE, IEEJ, IPSJ, RISP and IEICE.