Simulation on the Relative Location Recognition of Multi-Robots, Using MRDS

Wonjae CHO*, Kazuaki TANAKA**

*Kyushu Institute of Technology, wonjaein@isein.me.kyutech.ac.jp
**Kyushu Institute of Technology, kazuaki@ms.e.kyutech.ac.jp

This paper presents a simulation on the relative location recognition of multi-robots using MRDS (Microsoft Robotics Developer Studio). Multiple robots allow a more robust recognition of their relative locations in accuracy and performance compared to having just a single robot client. However, in order to experiment with a cooperative robot system that utilizes multiple robots units, there are constraints relating to large initial production costs for the robot units as well as having to secure a certain size of space for the experiment. This paper has resolved those issues by performing the multi-robot relative location recognition experiment in a simulated virtual space using MRDS, a robot application tool. Moreover, relative coordinates of robots were recognized by using omni-directional panoramic vision sensors on the simulation program.

Key Words: Multi-robot, MRDS, Panoramic vision

1. Introduction

It is important for a mobile robot to identify its current location for the autonomous execution of its given tasks in random space. While most previous studies focused on the recognition of a single robot’s location [1][2][3], recent studies are focusing more on the issue of collectively and simultaneously recognizing the locations of multi-robots for more efficient performances on their given tasks [4][5][6]. Since a cooperative robot system allows the recognition of locations by collecting multiple data sets from a number of robot clients, it is more robust in its accuracy and performance in location recognition compared to a single robot system. However, since multiple clients of robots are required for creating a cooperative robot system, it requires a large initial production cost and a certain minimum size of experiment space, the financial and spatial constraints for creating the system. While there may be various solutions to these issues, a simulation experiment in 3D-simulated space can be one of the answers. 3D simulation is a computer generated model that can apply the information and algorithms of the robots to the simulation system and operate the robots in virtual space [7]. By using simulations, the cost and time required for the development of robots can be minimized and it also provides the merit of performing the experiments in various settings. There are many types of tools for constructing simulations. Among those tools, MRDS (Microsoft Robotics Studio) allows the easy development of various settings for simulations by providing physical properties to objects by using PhysX SDK (Software Development Kit) by AGEIA, as well as by providing API (Application Programming Interface) that allows the user to directly develop various sensors. Thus, in this study, virtual multi-robots, which can operate individually, and omni-directional panoramic vision sensors were created by using the MRDS simulation tool. Then, they were used in the recognition of relative locations of robots.

This paper is organized as follows: Chapter 2 describes about the configuration of the simulation system that was created with MRDS. Chapter 3 shows the location recognitions on the simulation by using the omni-directional panoramic vision sensors. Chapter 4 summarizes the results of various simulations to verify the effectiveness of the proposed methodology. Finally, Chapter 5 states the conclusion of this paper as well as its future tasks.

2. Simulation System Configuration

MRDS is a robot application tool produced by Microsoft in 2006, which provides abstract software components that can be operated in various robot hardware platforms, allowing the users to easily develop various robot-related services [8]. Since each of the software components are loosely linked to one another, object-oriented services can be developed, improving their reusability. Moreover, it supports various sensor services (such as Web cam service, LRF service, GPS service, and Motor service), providing the optimum condition required for robot development. In terms of performance, MRDS supports CCR (Concurrency and Coordination Runtime) and DSS (Decentralized Software Service) engines. CCR supports simultaneity processing and parallel processing, and therefore, simultaneously processes the tasks performed by multi-robots, and DSS supports the communication between the robots on the network, allowing efficient collaboration [9].

2.1 System Configuration

The system can be categorized into 4 services as in Fig. 1. The Communication Service processes I/O services of the data from the actual Client by using UDP (User Datagram Protocol). The Orchestration Service recognizes the location of the counterpart Client by using the inputted sensor data values. The Runtime Environment Service is composed of CCR and DSS, which operates simultaneity processing and service connections. The Simulation Service performs the modeling of the actual robot, and displays the location of the counterpart robot that has been recognized by the Orchestration Service in the 3D simulated environment. This system configuration, once it is verified for its effectiveness on the proposed 3D simulated model, can be tested in the
actual world with the actual robot that has been created without the need for extra software development. Thus, it has the merit of being developed without relying on specific hardware. Moreover, since it is independent of the platforms, the codes can be reused and modules can be created among different services. Therefore, in this paper, the proposed model of the virtual 3D modeling robots, which are the alternatives to the actual robot clients, will be verified for its performance.

![Fig. 1 System block diagram](image)

### 2.2 Simulation Robot Configuration

Pioneer 3DX produced by Mobile Robots are used in this study as the virtual robots. Pioneer 3DX has been developed for performing robot-related tests in universities and institutes. As shown in Fig. 2, MRDS basically provides the 3D model and basic sensor services (LRF, Motor, GPS, Web Cam, etc) of Pioneer3DX. The robot sensor has created and placed the omni-directional panoramic vision sensor in the center point of the robot by inheriting CameraEntity Class that is provided by MRDS. The virtual robots can be controlled individually by using the created UI (User Interface) by modifying the DifferentialDrive Service that is provided by MRDS.

![Fig. 2 Pioneer3DX Simulation & SimpleDashBoard](image)

### 2.3 Omni-directional Panoramic Visual Configuration

A panoramic image is an image that has been produced by compiling a series of images to provide a view from all directions rather than from a single viewpoint [10]. Conventional studied methods for creating panoramic images are divided into hardware-based and software-based methods. Hardware-based methods acquire images either by using multiple cameras or by using a catadioptric omni-directional camera, making it easier to produce panoramic images than the software-based methods, thus, having an advantage over them [11]. In this study, the hardware-based method, which acquires panoramic images by using multiple cameras, has been selected, and has been operated on the 3D simulation as shown in Fig. 3.

In order to create a panoramic image as shown above, images from fixed field of view cameras need to be displayed in between fixed intervals and renewed periodically.

As shown in Fig. 4, let’s assume that 90 degree field of view cameras are creating panoramic images on the simulation. By shooting 4 images in the order of ‘front,’ ‘right,’ ‘rear,’ and ‘left’ from 1 to 4V and by successively matching (register and compile) them, a panoramic image is created. When creating a panoramic image, Arbiter Class from Microsoft.Ccr.Core Namespace is used to display the matched images simultaneously to resolve the image errors that are caused by the time difference during the matching process.

![Fig. 3 Panoramic Image on the Simulation](image)

![Fig. 4 Horizontal Plane's Field of View of the Panoramic Image](image)

### 3. Relative Location Recognition

To recognize the locations of the surrounding robots of a fixed client, the data transferred from the sensors were processed in 4 stages as shown in Fig. 5. In Stage 1, the region of interest is extracted by using binary processing and sequential labeling, which separates the objects from the background. In the second stage, the pattern most similar to that of a robot is extracted from the region of interest by template matching. In the third stage, the extracted pattern is used to calculate the relative location of the counterpart robot from the base point robot. Finally, by using the calculated locations of the surrounding robots, the mobile robot is tracked. In this paper, only the process from stages 1 through 3 is discussed.

![Fig. 5 Block Diagram depicting the Relative Location Recognition](image)
3.1 Extracting the Region of Interest

It is very important to distinguish an object from its background in order to extract the object from the input image. Binary processing changes the image into ‘0’s and ‘1’s, based on the threshold value, which is a randomly fixed constant. Binary coding can be divided into P-tile, Mode, iterative, and adaptive methods. In this study, since the brightness of the simulated space is constant and there are no obstacles, adaptive method binary coding was used for binary processing of the images. The adaptive method does not use the overall histogram of an image, but rather uses a part of the histogram to calculate the threshold value for that given portion [12]. The whole image is divided into 4 partial images that have the same sizes. All of the partial images are surveyed for their histograms and their threshold values are determined. When binary processing of the image is complete, the sequential labeling algorithm is used to bundle up a single region of pixels that form each of the objects as shown in Fig. 6. Then, the region of interest is probed once again and the connected regions of the ‘front’ and ‘left’ component images of the panoramic image are bundled up as identical regions.

3.2 Template Matching

The extracted region of interest uses the preset template pattern to find a similar pattern from the inspected image. The template pattern is created by analyzing the image patterns of Pioneer 3DX which were taken from various angles. By using the pixel coordinate information of the region of interest, the raw input image and template patterns are compared by Normalized Gray-level Correlation (NGC) [13]. Since NGC normalizes the brightness of the inspected image and the template before comparing them, it is robust to the illumination changes. The equation for NGC is as shown in Eq. (1). ‘i’ and ‘j’ are the indices for the pixel coordinates. ‘a(i, j)’ is the value of the brightness of the compared section of the region of interest after subtracting the average ‘m’. ‘b(i, j)’ is the value of brightness of the template pattern after subtracting ‘t’. The normalized image, the image that has been acquired by subtracting the average brightness value, is compared for similarity by using a vector formula for comparison.

\[
R = \frac{\sum_{i=0}^{M} \sum_{j=0}^{N} a(i, j) b(i, j)}{\sqrt{\sum_{i=0}^{M} \sum_{j=0}^{N} a(i, j)^2 \sum_{i=0}^{M} \sum_{j=0}^{N} b(i, j)^2}}
\]  

3.3 Calculating the Relative Locations of Robots

Since no landmarks or identifying points are used for setting an absolute coordinate system in this study, the locations of the surrounding robots are calculated by using relative coordinate systems. Designating the standard client’s location as the standard point (the origin point), one of the location coordinates of a counterpart robot is assumed to be another point on a single axis that connects the two points, and the addition of a perpendicular axis to this single axis creates a relative coordinate system as shown in Fig. 7. By assuming R1 as a standard point, R2 and R3 can be calculated by Eq. (2).

\[
\begin{align*}
R1 &= \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
R2 &= \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} d_2 \\ 0 \end{pmatrix} \\
R3 &= \begin{pmatrix} x_3 \\ y_3 \end{pmatrix} = \begin{pmatrix} \sqrt{d_3^2 - y_3^2} \\ \left(d_3^2 + d_3^2 - d_3^2/2d_3 \right) \end{pmatrix}
\end{align*}
\]  

4. Experiment Results

To verify the performance of the proposed methodology, a virtual experimental setting was configured as in Fig. 8. Three identical units of Pioneer3DX virtual robots were used for the simulated experiments. The virtual experimental environment was an empty space with a 10m x 10m dimension with no obstacles other than the robots themselves, having similar physical properties with the real world. The simulations were performed on a laptop with a performance of 1.86GHz CPU and 1.5GB RAM.

In the experiment, the robots were placed in 1m intervals and their locations were measured by simulated sensors. To verify the accurateness of the robots’ locations, the initial locations of the robots were modified by changing the simulation coordinate values. In Fig. 9, the images collected for the location recognition, and the identified objects in each image are displayed. The relative locations of the surrounding robots are also shown by calculating the collected data.
Table 1 shows the resulting positions of the recognition process as well as their error rates. The average error rate is 5.26. Thus, the proposed simulation method is proven to have a satisfactory performance in measuring the relative locations among cooperative robots.

Table 1. Location Recognition Results and Error Rates

<table>
<thead>
<tr>
<th>Absolute Distance (m)</th>
<th>Standard Robot</th>
<th>Measured Robot</th>
<th>Measured Distance (m)</th>
<th>Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R1</td>
<td>R2</td>
<td>0.98</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>0.99</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>R1</td>
<td>R2</td>
<td>0.96</td>
<td>4.0</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>0.99</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>R1</td>
<td>R2</td>
<td>1.90</td>
<td>5.0</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>1.96</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>R1</td>
<td>R2</td>
<td>1.93</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>1.94</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>R1</td>
<td>R2</td>
<td>1.90</td>
<td>5.0</td>
</tr>
<tr>
<td></td>
<td>R3</td>
<td>1.91</td>
<td>4.5</td>
<td></td>
</tr>
</tbody>
</table>

5. Conclusion

This paper proposed the recognition of relative robot locations by the use of panoramic vision sensor on MRDS simulations. Such simulation system allows the early discovery of kinematic errors of the robots, and reduces the development costs and time since the platforms can be reused. Moreover, it has shown that relative locations of robots can be recognized by using simulated panoramic vision cameras. However, since the current system recognizes only the locations of standstill robots, it cannot track the locations of moving robots. Therefore, a simulation system might be developed in the future that can track the locations of robots in motion.

References