Neighbor Camera Node Determination Algorithm for Automatic Human Tracking System based on Mobile Agent Technologies

K. Tanigawa*, K. Takahashi**, T. Kawamura** and K. Sugahara**.

Abstract — New method to construct automatic human tracking systems based on mobile agent technologies is discussed. In the proposed system, relationships between video camera nodes according to their locations become important problems. An algorithm to determine neighbor video camera nodes from deployed camera positions according to their view area is introduced. The proposed algorithm is utilized to generate the neighbor video camera information in automatic human tracking system. The automatic human tracking system is enhanced by the video monitoring system utilizing the mobile agent technologies. Mobile agent is suitable for distributed processing and parallel processing, since mobile agent can migrate and run on the distributed computers. In our system, one mobile agent is deployed for one tracking object. When a camera misses the object that one agent is tracking, the agent migrates to neighbor camera nodes. Here, we need to compute the neighbor camera nodes but its computation is very complicated due to their deployed positions and view area. In addition, the deployed position and the view area of each video camera change due to unexpected circumstances. In this paper, we propose a robust computation algorithm which is not influenced in any circumstances, and evaluate it.

Keywords — Location matrix, adjacency matrix, automatic human tracking system, mobile agent technologies

I. Introduction

In recent years, video monitoring systems are deployed into many companies or public offices. Their video monitoring systems are efficient to track a suspicious person, however, their systems are not suitable to track two or more person simultaneously because of following problems:

- In the conventional video monitoring systems, all the tracking is processed in one centralized server. This limits the number of people the system can track simultaneously because of its computational power;
- When a suspicious person walks out of a camera view area, operators have to switch cameras to find the suspicious person. The operator may confuse which camera he/she changes to;
- An operator cannot track a lot of people simultaneously because of his/her workload;
- To identify a person is difficult for an operator.

A number of researches try to solve these problems. K.Terashita et.al. and Y.Kawaguchi et.al proposed in [1] and [2] using pan/tilt cameras to reduce the workload of operators. These approaches, however, do not consider the case of tracking numerous people. Efficient camera positioning at a particular monitoring station are discussed in [3],[4] and [5], however, these approaches require installing a lot of cameras. Image processing is important to identify a suspicious person and requires much computational power.

In this paper, we propose a mobile agent-based system to track suspicious people. The mobile agent technologies are suitable for distributed processing and parallel processing, since mobile agents can transport agent programs and run on distributed computers. In our system, we deploy one mobile agent [6],[7] per one suspicious person. The mobile agent has the feature information of a tracking object, and migrates from a camera to a camera. Since the mobile agent runs on a camera, the mobile agent uses computational resource on that camera. This naturally actualizes load balancing, thus, enables to track numerous people simultaneously. We call our video monitoring system “Automatic Human Tracking System [8], [9]”.

At the start of tracking, an operator generates a mobile agent that has the feature information of a suspicious person. The mobile agent migrates from a camera to a camera for tracking the suspicious person. When the mobile agent migrates to other camera, it has to know the information of neighbor cameras. Though the video cameras are connected on an IP network, it is difficult to determine neighbor video cameras on the IP network. Therefore, we connect the physical position of video camera with the IP address by using a system configuration file.

The neighbor camera node (the neighbor camera is called “neighbor camera node” in this paper.) is determined by the location and view area of video cameras. In addition, the deployed position and the view area of each video camera would change due to unexpected circumstances. In this paper, we propose a robust computation algorithm that is not influenced by any circumstances, and evaluate it.

* K. Tanigawa is with Melco Power Systems Co., Ltd, Kobe, Hyogo 652-8555 Japan (e-mail: Tanigawa.Kozo@zx.MitsubishiElectric.co.jp).

**K. Takahashi, T. Kawamura and K. Sugahara are an associate professor and professors of Graduate School of Engineering, Tottori University, Tottori 680-8552 Japan (e-mail: takahashi/kawamura/sugahara@ike.tottori-u.ac.jp).

II. Automatic Human Tracking System

The overview of our automatic human tracking system is shown in Fig.1. Our system consists of an agent monitoring terminal, an agent management server, a video recording server, video cameras and feature extraction servers.

The agent monitoring terminal is used to generate a mobile agent that has the feature information of a suspicious person, the management of mobile agent locations, and for the display of pictures captured by video cameras. The agent management server records the agent's tracking information, provides the agent status information to the agent monitoring terminal, and requests a video image to the video recording server. The video recording server records all video images and provides the images to the agent monitoring terminal by a request from the agent management server. The feature extraction server is set up by the video camera, analyzes the person image and extracts the feature information from the image.

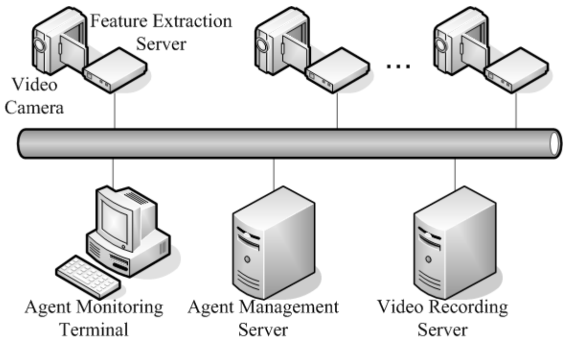

Fig. 1 System overview

A mobile agent tracks a person by using feature information and the neighbor nodes information. The processing flow of the proposed system is as follows:

1) The feature extraction servers analyze the person image and extract his/her feature information.
2) The video recording server records video images captured by all video cameras from feature extraction servers.
3) An operator selects a person that he/she wants to track on the screen of the agent monitoring terminal.
4) An agent monitoring terminal generates a mobile agent with the mobile agent information including the feature information of the suspicious person.
5) The mobile agent migrates to the feature extraction server and starts to track the suspicious person.
6) When the mobile agent finds its tracking person through the feature extraction server, the mobile agent notifies it to the agent management server with information such as the video camera number, discovery time, and mobile agent identifiers.
7) When the person moves out of the camera, the mobile agent migrates to neighbor camera nodes.

By repeating from step 6) and 7), the mobile agent can track the suspicious person.

III. Algorithm to Determine Neighbor Nodes

A. Problem on Determining Neighbor Video Cameras

Cameras can be classified into fixed cameras and PTZ cameras. PTZ cameras have the functions of Pan, Tilt and Zoom, whereas fix cameras have only Zoom function. When a camera is panned, tilted or zoomed, the view area of the camera changes. Then, the camera may miss a suspicious person. Therefore, the change of the view area should be taken into consideration for the calculation of neighbor camera nodes. Furthermore, the view area of the camera may change frequently by pan, tile and/or zoom, so that, dynamic and faster calculation of neighbor camera nodes are required.

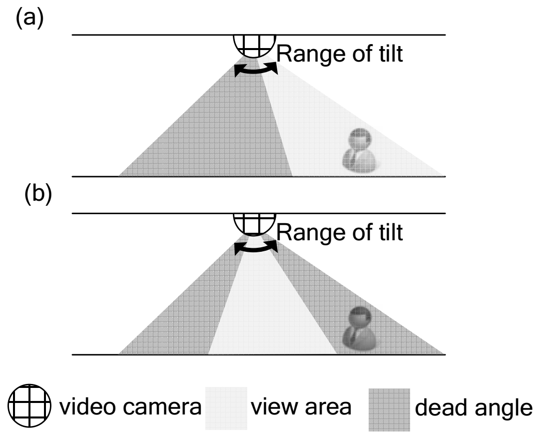

Fig.2 The view areas of a camera when the camera is tilted

Fig. 2 shows the difference of a view area when a camera is tilted. The camera holds a suspicious person in the situation of Fig. 2 (a), but it misses the suspicious person after the camera is tilted, as shown in Fig. 2 (b).

It is possible to calculate view area of all the ranges of Pan, Tilt, and Zoom as shown in Fig. 3. However, it requires high experiences and knowledge.

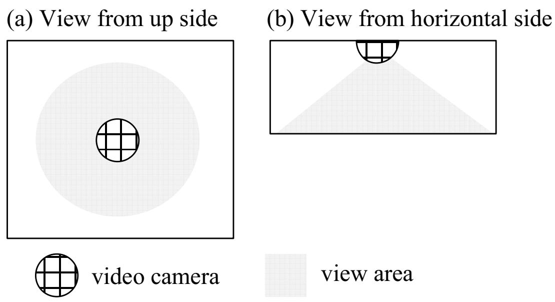

Fig.3 View area of all ranges of PTZ

The algorithm proposed in this paper can determine the neighbor camera node without the influence of view area. In our algorithm, we calculate possible view area of when the camera is panned, tilted and/or zoomed, shown in Fig. 3. This calculation is difficult, but we elaborated it. Moreover, a definition of the information on the currently installed camera can be made intuitively.

B. Overview of the Algorithm

The algorithm to determine neighbor nodes in this paper uses a location matrix. We use two location matrices: one is location matrix M made from visible blocks and invisible blocks; the other is location matrix X made from the view block of each camera. The visible, invisible, view blocks are defined as follows:

Visible Block: Blocks in which people can walk are labeled as *visible block*.

Invisible Block: Blocks in which people cannot walk are labeled as *invisible block*

View Block: Blocks that a camera is viewing are labeled as *view blocks*. We represent view blocks as **V**=$\{v_1, v_2, ..., v_p\}$.

We also define a camera block to help the calculation of matrix X and M.

Camera Block: A block in which a camera locates is labeled as *camera block*. We represent camera blocks as **A**=$\{a_1, a_2, ..., a_p\}$.

We put following constraints from above definitions:

- All blocks are classified into visible blocks or invisible blocks;
- One camera may have two or more view block;
- A block can be both a camera block and a view block;
- A block may be a view block of two or more cameras;
- If a camera block can also be a view block

The algorithm to determine the neighbor nodes is as follows:

1) A floor is divided into blocks, and each block is set up as a visible or an invisible block. Passages, entrances and so on are labeled as visible block. Other blocks are labeled as invisible blocks.
2) Blocks in which a camera is set up are labeled as camera blocks.
3) The view area of a camera is measured. Blocks that the camera is viewing are set as view blocks with the camera identifier.
4) Location matrix *M* is made from visible and invisible blocks.
5) Location matrix *X* is made from view blocks.
6) We calculate location matrix *Y* from the location matrix *M* and *X*.
7) We can get an adjacency matrix by the multiplication of location matrix *X* and *Y*. The adjacency matrix shows the neighbor camera node information.

Mobile agents know the neighbor camera nodes and can track their targeted suspicious person.

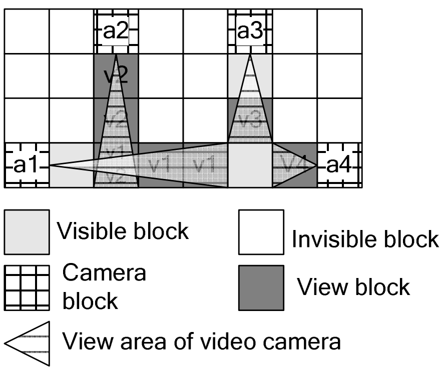

Fig.4 Example of applied definitions and view area

C. Consideration for Determination Algorithm

Neighbor node determination algorithm can easily determine the neighbor video camera's location without regarding to the influence of view distances and any modification of the information of the currently installed cameras. The modification information is set on the system that is expressed as a matrix using a diagram. The elements of a matrix are called as blocks, and blocks are used to compute neighbor video camera's information in our algorithm. Blocks are defined as visible blocks, invisible blocks, camera blocks and view blocks. Camera blocks are the location of video cameras that are labeled as camera nodes. The blocks and nodes are defined as $A = \{a_1, a_2, ..., a_p\}$. This block and node is also a server with video camera. View block nodes are defined as $V = \{v_1, v_2, ..., v_q\}$.

Figure 4 shows an example of these definitions applied and shows the view area of the video cameras. The algorithm accomplishes neighbor camera node determination by using location matrix. The algorithm uses three kinds of location matrix. The 1st is a location matrix *M* made from view blocks and invisible blocks. Element m_{ij} of matrix *M* is defined as (1). The 2nd is a location matrix *X* made from view blocks. Element X_{ij} of matrix *X* is defined as (2). And the last is a location matrix *Y* made from the calculation results of location matrix *M* and *X*. Element y_{ij} of matrix *Y* is defined as (3). The neighbor information for video cameras is calculated from the connection information of view blocks by using location matrix *X* and *Y*.

$$m_{ij} = \begin{cases} -1, & \text{Invisible block.} \\ 0, & \text{Visible block.} \end{cases} \quad (1)$$

$$x_{ij} = \begin{cases} 1, & \text{View block .} \\ 0, & \text{Not a view block.} \end{cases} \quad (2)$$

$$y_{ij} = \begin{cases} 1, & \text{There is a connection point with other cameras.} \\ 0, & \text{There is no connection.} \end{cases} \quad (3)$$

The algorithm to determine the neighbor nodes is:

1) Divide a floor into a number of blocks suitable for floor layout.
2) Set visible block, invisible blocks, and camera blocks on the diagram as shown in Fig.4.
3) Calculated view blocks from camera specifications.
4) Generate a location matrix *M* from visible blocks and invisible blocks, and generate a location matrix *X* from view blocks for each camera. For example, in Fig. 4, the location matrix M is (4) and the location matrix X of camera *a1* is (5).
5) Calculate location matrix *X'* which is the composition of all location matrix *X* except the target camera.
6) Generate a location matrix *Y* by checking the location matrix *X'* of the target camera. The example of camera *a1* of Fig.4 is shown in (6).

7) Calculate neighbor's adjacency matrix by multiplying location matrix and matrix *Y*. This neighbor's adjacency matrix is the neighbor's camera node information. The results of the example of Fig.4 are shown in Tables I.

$$M = \begin{pmatrix} -1, -1, 0,-1,-1, 0,-1,-1 \\ -1, -1, 0,-1,-1, 0,-1,-1 \\ -1, -1, 0,-1,-1, 0,-1,-1 \\ 0, 0, 0, \ 0, 0, 0, 0, 0 \end{pmatrix} \quad (4)$$

$$X = \begin{pmatrix} 0, 0, 0, 0, 0, 0, 0, 0 \\ 0, 0, 0, 0, 0, 0, 0, 0 \\ 0, 0, 0, 0, 0, 0, 0, 0 \\ 0, 0, 1, 1, 1, 0, 0, 0 \end{pmatrix} \quad (5)$$

$$Y = \begin{pmatrix} 0, 0, 0, 0, 0, 0, 0, 0 \\ 0, 0, 0, 0, 0, 0, 0, 0 \\ 0, 0, 0, 0, 0, 1, 0, 0 \\ 0, 0, 1, 0, 0, 0, 1, 0 \end{pmatrix} \quad (6)$$

TABLE I
CALCULATED ADJACENCY MATRIX

E	a1	a2	a3	a4
a1	0	1	1	1
a2	1	0	0	0
a3	1	0	0	1
a4	1	0	1	0

IV. EXAMINATION

We evaluate the proposed algorithm on the floor plan as shown in Fig.5. We assume each camera has three kinds of view distance. Figures 6 to 8 show the situation at each time period. The arrows connecting cameras show that the cameras are each others neighbors. The results on the evaluation are shown in Tables II, III and IV.

The view area of the camera (a3) changes the left side in Fig.6 to the right side in Fig.7. As the result, the camera (a1) and the camera (a2) are connected, but the camera (a2) and the camera (a4) is disconnected. The view distance of the camera (a5) changes from Fig. 6 to Fig. 8. As the result, the camera (a1) and the camera (a2) are connected, but the camera (a2) and the camera (a4) is disconnected.

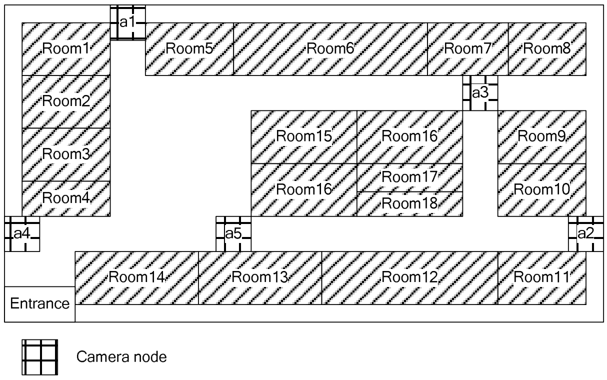

Fig.5 Base floor plan for examination

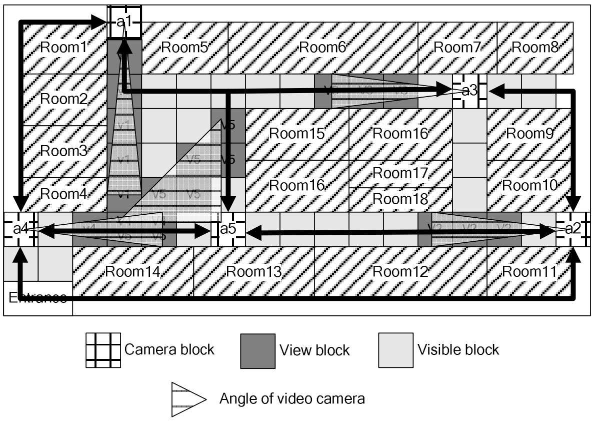

Fig. 6 Diagram A

TABLE II
CALCULATED ADJACENCY MATRIX FOR DIAGRAM A

E	a1	a2	a3	a4	a5
a1	0	0	1	1	1
a2	0	0	1	1	1
a3	1	1	0	0	1
a4	1	1	0	0	1
a5	1	1	1	1	0

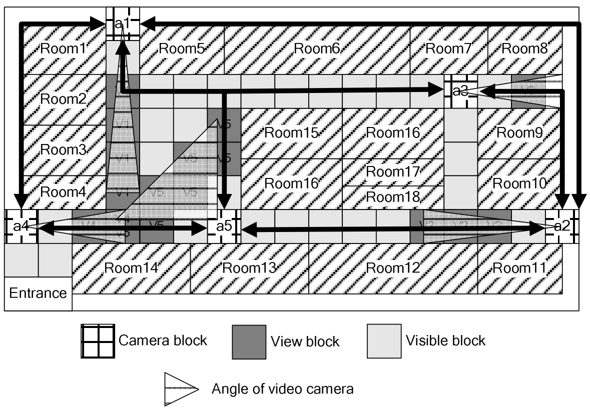

Fig. 7 Diagram B

TABLE III
CALCULATED ADJACENT MATRIX FOR DIAGRAM B

E	a1	a2	a3	a4	a5
a1	0	1	1	1	1
a2	1	0	1	0	1
a3	1	1	0	0	1
a4	1	0	0	0	1
a5	1	1	1	1	0

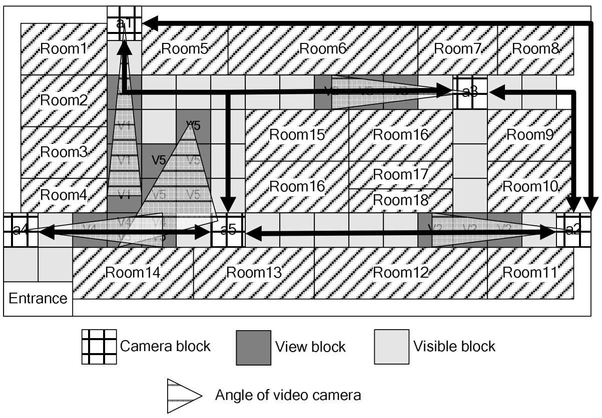

Fig. 8 Diagram C.

TABLE VI
CALCULATED ADJACENCY MATRIX FOR DIAGRAM C

E	a1	a2	a3	a4	a5
a1	0	1	1	1	1
a2	1	0	1	0	1
a3	1	1	0	0	1
a4	1	0	0	0	1
a5	1	1	1	1	0

V. CONCLUSION

In our system, the computation of image processing is distributed by using mobile agent technologies. Since the mobile agent tracks a target object automatically, the workload of an operator is reduced. For the automatic tracking, we proposed the algorithm that intuitively determines the neighbor nodes, even when various kinds of cameras are installed in the system. The algorithm considers the moving direction of a suspicious person and the view area of each camera. The evaluation results show mobile agents can migrate among camera nodes and efficiently track a suspicious people.

REFERENCES

[1] K. Terashita, N. Ukita, M. Kidode, "Efficiency Improvement of Probabilistic-Topological Calibration of Widely Distributed Active Cameras," IPSJ SIG Technical Report, 2009-CVIM-166, 2009, pp. 241-248.

[2] Y. Kawaguchi, A. Shimada, D. Arita, R. Taniguchi, "Object Trajectory Acquisition with an Active Camera for Wide Area Scene Surveillance," IPSJ SIG Technical Report, 2008-CVIM-163, 2008, pp. 1306-1311.

[3] Y. Yao, C.-H Chen, B. Abidi, D. Page, A. Koschan, M. Abidi, "Sensor Planning for Automated and Persistent Object Tracking with Multiple Cameras," CVPR2008, Jun. 2008.

[4] Y. Yao, C.-H Chen, B. Abidi, D. Page, A. Koschan, M. Abidi, "Sensor Planning for PTZ Cameras Using the Probability of Camera Overload," ICPR2008, Dec. 2008.

[5] U.M. Erdem, S. Sclaroff, "Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements," CVIO2006, Vol. 103, No.3, Sep. 2006, pp. 156-169.

[6] H. Kakiuchi, Y. Hamada, T. Kawamura, T. Shimizu, K. Sugahara, "To realize Automatic Human Tracking System based on Mobile Agent Technologies," in Proceedings of the 59^{th} Chugoku branch union convention of the Institute of Electrical Engineers of Japan and Information Processing Society of Japan in Tottori, Tottori, 2008, pp. 485.

[7] Y. Hamada, S. Iwasaki, H. Kakiuchi, T. Kawamura, K. Sugahara, "Pursuit methods for Automatic Human Tracking System based on Mobile Agent Technologies," in Proceedings of the 59^{th} Chugoku branch union convention of the Institute of Electrical Engineers of Japan and Information Processing Society of Japan in Tottori, Tottori, 2008, pp. 486.

[8] S. Motomura, T. Kawamura, K. Sugahara, "Maglog: A Mobile Agent Framework for Distributed Models," in Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems, Phoenix, Arizona, USA, 2005, pp. 414-420.

[9] T. Kawamura, S. Motomura, K. Sugahara, "Implementation of a Logic-based Multi Agent Framework on Java Environment," in Proceedings of IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems (Henry Hexmoor (eds.)), Waltham, Massachusetts, USA, 2005, pp. 486-491.

[10] N. Ishibashi, Y. Hamada, H. Kakiuchi, T. Shimizu, T. Kawamura, K. Sugahara. "Feature extraction method for Automatic Human Tracking System based on Mobile Agent Technologies," in Proceedings of the 59^{th} Chugoku branch union convention of the Institute of Electrical Engineers of Japan and Information Processing Society of Japan in Tottori, Tottori, 2008, pp. 418.

K. Tanigawa was born in 1972. He graduated Hyogo technical High school, Kobe, Japan, in 1991, and has joined Melco Power Systems Co., Ltd, Kobe, Japan from 1991. He was a team leader in Engineering Section 3 of the company in 2001. He is now researching the mobile agent technologies and automatic human tracking system as a member of Tottori University.

K. Takahashi Received the B.E., M.E. and D.E. degree in Computer Science and Electrical Engineering, Kyushu University in 1999, 2001 and 2004, respectively. From 2004 to 2010, he worked at Institute of Systems & Information Technologies/KYUSHU as a researcher. Now he is an associate professor of Department of Information and Electronics at Tottori University. His research interests include security, ubiquitous computing and multi-agent system. He is a member of IEICE, IPSJ and IEEJ.

T. Kawamura was born in 1965. He obtained his B.Eng., M.Eng. and Ph.D. degrees in Computer Engineering from Kobe University, Japan in 1988, 1990 and 2002, respectively.

Since 1994 he had been in Tottori University as a research associate and has been in the same University as an associate professor in the Faculty of Engineering since 2003. His current research interests include mobile-agent systems and distributed systems. Dr. Kawamura is a member of IPSJ and IEICE.

K. Sugahara received the B.Eng. degree from Yamanashi University in Japan in 1979 and M.Eng. degree from Tokyo Institute of Technology, Japan, in 1981. In 1989, he received the D.Eng. degree from Kobe University, Japan.

From 1981 to 1994, he was staff of the Department of Electronic Engineering, Kobe City College of Technology. In 1994, he joined Tottori University as an associate professor of the Department of Electrical and Electronic Engineering and he is a professor of the Department of Information and Knowledge Engineering. His current interest lies in the fields of mobile-agent system applications, computer architectures and hardware realizations of 1D and multidimensional signal processing algorithms. Prof. Sugahara is a member of IEEE, IEICE and IPSJ.