An Algorithm to Determine Neighbor Nodes for Automatic Human Tracking System

H. Kakiuchi
Melco Power Systems Co., Ltd.
Kobe, Japan
Kakiuchi.Hiroto@zs.MitsubishiElectric.co.jp

T. Kawamura, T. Shimizu, and K. Sugahara
Graduate School of Engineering, Tottori University
Tottori, Japan
{kawamura, tadaaki, sugahara}@ike.tottori-u.ac.jp

Abstract— **The determination algorithm of neighbor video cameras by deployed position and view distance of video camera is introduced in this paper. This algorithm is utilized to determine the neighbor video cameras in the Automatic Human Tracking System. The Automatic Human Tracking System is enhanced Video Monitoring System utilizing the Mobile Agent Technologies. Information of the neighbor video cameras is needed in the Video Monitoring System during the transition of video display. And the mobile agent in the Automatic Human Tracking System also utilizes such information to pursue the human efficiently. On the event of modifying the installed cameras in the system the following complications occur: Updating of the camera information and computation of the neighbor cameras. As such the Automatic Human Tracking System have inherited the above problems. For this reasons, the algorithm is developed to solve the mentioned problems.**

I. Introduction

In the Automatic Human Tracking System, the mobile agent in [1], [2], [3] and [4], a mobile agent pursues a human captured by a video camera. The video camera together with server computer is deployed at the monitoring position therefore the video camera and the server is treated as a single entity in this paper. The Automatic Human Tracking system is introduced in the next section.

When a person is to be pursued, a mobile agent is generated for the particular person. The mobile agent will follow the next location of the person utilizing the neighbor camera/server location information after verifying the feature of the person within the server [5]. If there is no neighbor camera/server location information, the mobile agent will not be able to pursue the person likewise it will not be able to switch the video display in the Video Monitoring System, or there will be a need to deploy new mobile agents on the servers which is not efficient. And it is difficult to determine the neighbor server in IP network but it is enable to search the neighbor server by defining the relation between IP address and physical position of video camera into the system configuration file.

The neighbor camera/server information is called neighbor camera node/nodes in this paper. The neighbor camera node is determined by the location and view distance of the video camera. The neighbor camera nodes differ by the difference of view distance and view overlap of the video camera even if video camera's locations are same. Therefore it is necessary to update the information of cameras on the event that the number of installed cameras will be modified.

The algorithm to determine neighbor nodes resolves the above problems.

II. Automatic Human Tracking System

The system configuration of the Automatic Human Tracking System is shown in Fig.1. It assumes that the system is used in a building, the user capture an image of the face and the body of a person by the video camera, the user registers the image into the system, and person with un-registered image cannot roam inside the building.

This system is composed of Agent Monitoring terminal, Agent Management server, Video Recording server, and Feature Extraction server with video camera. The Agent Monitoring terminal is used for registering the person, for confirming the current location staying the agent, and for displaying video of the person captured. The Agent Management server records the agent's tracking information, provides the information to the terminal, and requests video image to the Video Recording server by an operation from the terminal. The Video Recording server records all video images and provides the images to the terminal by request from the Agent Management Server. The Feature Extraction server sets up by the video camera, analyzes the person image, and extracts the feature information from the image. An agent pursues the person by using the feature information and the neighbor nodes information.

The processing flow of the proposed system is the following:

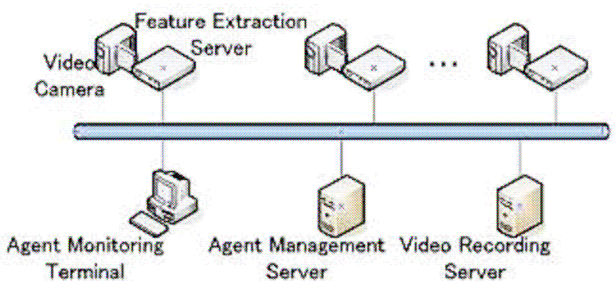

Figure 1. System configuration.

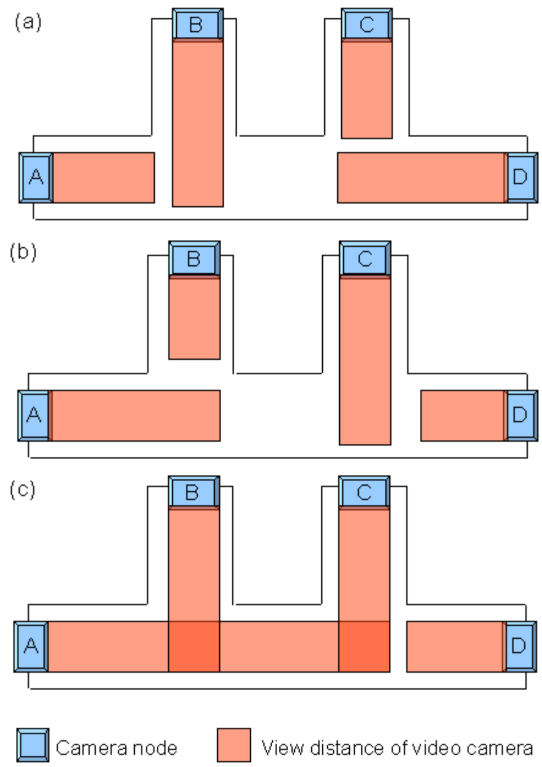

Figure 2. Example that view distance influences.

i. The Video Recording server records video images from all video cameras at all time.

ii. The user selects the person on the screen of the Agent Monitoring terminal, and extracts the pursued person's feature information as electronic.

iii. A mobile agent is generated for the person and registered as the mobile agent information including the feature information into the system.

iv. The mobile agent is deployed on the first Feature Extraction server, and begins pursuing the person.

v. When the mobile agent finds the person, the mobile agent notifies the information such as the video camera number, the discovery time, and the mobile agent identifier first to the Agent Management server.

vi. When the person moves to the area of the next video camera, the mobile agent transfers to the next video camera near the person.

vii. If the mobile agent finds the person, the mobile agent notifies the information as stated on item v.

The mobile agent repeats above process from step v. to vi. until the person goes out a building.

III. ALGORITHM TO DETERMINE NEIGHBOR NODES

A. Proprem on Determining Neighbor Video Cameras

A scenario in which neighbor video camera location is influenced by view distance is described in Fig. 2. Fig. 2 shows 3 diagrams portraying a floor plan with 4 video cameras each. And each video cameras view distances are different. It assumes that person to be pursued starts to move from the location of video camera A. The neighbor of video camera A in object (a) of Fig. 2 is video camera B and, not C and D. In object (a) of Fig. 2, video camera C and D are not considered as neighbor of video camera A, because video camera B blocks the view of video camera C and D. And the person being pursued can be captured at an earlier time on video camera B. But in the case of object (b) of Fig. 2, the neighbor of video camera A are video camera B and C but not camera D. Thus neighbor video camera's location indicates the difference by view distance of video camera. The case of object (c) in Fig. 2 becomes more complicated. The neighbor of video camera A in object (c) of Fig. 2 is video camera B, C, and D. And video camera B is not considered as neighbor of video camera C. It is because video camera A exists as neighbor between video camera B and C. When it is assumed that a person move to D from A, the person is sure to be captured with video camera in order of video camera A, B, A, and C.

On this scenario it indicates that determining the neighbor definition is influenced from the change of view distance and it becomes more complicated as the number of video camera increases.

B. Overview of the Algorithm

The algorithm to determine neighbor nodes that was developed can easily determine the neighbor video camera's location without the influence of view distance and modification of the information of the currently installed camera. It is set on the nodes to compute neighbor video cameras on the diagram and the diagram is expressed as graph. The node is defined as the followings:

Camera Node: the location of video camera is labeled as camera node and the nodes are defined as $\mathbf{A}=\{a_1, a_2, \ldots, a_p\}$.

Non-camera Node: the nodes are defined as $\mathbf{V}=\{v_1, v_2, \ldots, v_q\}$. Conditions of a non-camera node is stated below:

- Either of crossover, corner, terminal of passage.
- The position where video camera is installed.
- The end point of the view distance of video camera.

In addition, the point where the above conditions have overlapped is treated as one node. When the view distance of the video camera has reached non-camera node, the non-camera node is defined as the neighbor of the camera node. When two non-camera nodes are next to each other on a course, it is defined that those nodes are neighbor. Fig. 3 shows the example of applying these definitions and shows view distance of video camera.

The algorithm to determine neighbor nodes in this paper uses adjacent matrix. Two kinds of the adjacent matrix are needed in the algorithm. One is adjacent matrix **X** made from camera nodes' location as row and non-camera nodes' location as column. Another one is adjacent matrix **Y** made from non-camera nodes' location as row and column. The neighbor information of video camera is calculated from the connection information of non-camera node by using adjacent matrix **X** and **Y**. However neighbor information between video camera B and C are miscalculated in the situation like (c) of Fig.2. Because non-camera nodes that exist in the view distance of video camera B and C are neighbor. Therefore, when the non-camera node that crosses each other the view distance of two or more video camera exists mutually, it is necessary to break the connection between those nodes. It explains details in the definition of adjacent matrix **Y**.

The below is the algorithm to determine neighbor nodes:

- Put camera nodes and non-camera nodes on the diagram.
- Make adjacent matrix **X** from camera nodes' location and non-camera nodes' location, and make adjacent matrix **Y** from non-camera nodes' location. Adjacent matrix **X** indicates that row is camera node and column is non-camera node. Adjacent matrix **Y** indicates that row and column is non-camera node. And, it makes adjacent matrix Y to resolve the problem of (c) in Fig.2.
- Calculate adjacent matrix **X'** and **Y'** by excluding unnecessary non-camera nodes from adjacent matrix **X** and **Y**.
- Calculate neighbor's location matrix by multiplying adjacent matrix and transposed matrix $\mathbf{X'}^{\mathbf{T}}$. This neighbor's location matrix is neighbor's nodes information.

Unnecessary non-camera node is a non-camera node in which it has no camera node as a neighbor. Adjacent matrix **X**', **Y**' is computed without unnecessary nodes, and using the procedure stated in *"D. Removing of Unnecessary Non-camera Node"*. It would be better to include the unnecessary nodes in the diagram from the beginning. Since the risk of committing an error will be very high as the diagram becomes large. Including of the unnecessary nodes from the beginning and remove it at the end.

C. Consideration for Determination Algorithm

The diagram on Fig. 4 is expressed in Fig. 3 as graph. In Fig. 4, adjoining nodes are connected with the line. TABLE I is adjacent matrix **X** that was digitalized based on Fig. 4 and TABLE II is temporary adjacent matrix **Y**, which was digitalized, based on Fig. 4 before satisfying all conditions. TABLE III is adjacent matrix **Y** after all conditions are satisfied.

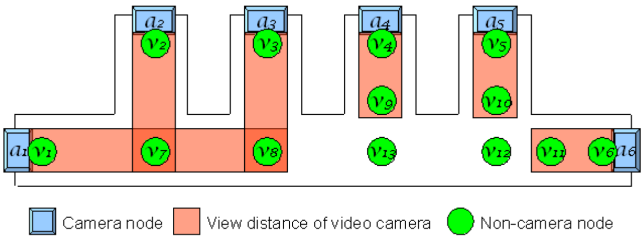

Figure 3. Figure that sets non-camera nodes.

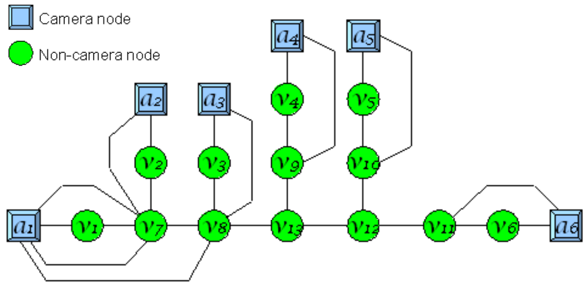

Figure 4. Graph of Fig.3.

TABLE I. ADJACENT MATIRX X WITH CAMERA NODES AND NON-CAMERA NODES

X	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13
a1	1	0	0	0	0	0	1	1	0	0	0	0	0
a2	0	1	0	0	0	0	1	0	0	0	0	0	0
a3	0	0	1	0	0	0	0	1	0	0	0	0	0
a4	0	0	0	1	0	0	0	0	1	0	0	0	0
a5	0	0	0	0	1	0	0	0	0	1	0	0	0
a6	0	0	0	0	0	1	0	0	0	0	1	0	0

TABLE II. ADJACENT MATRIX Y WITH NON-CAMERA NODES AND NON-CAMERA NODES

Y	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13
v1	0	0	0	0	0	0	1	0	0	0	0	0	0
v2	0	0	0	0	0	0	1	0	0	0	0	0	0
v3	0	0	0	0	0	0	0	1	0	0	0	0	0
v4	0	0	0	0	0	0	0	0	1	0	0	0	0
v5	0	0	0	0	0	0	0	0	0	1	0	0	0
v6	0	0	0	0	0	0	0	0	0	0	1	0	0
v7	1	1	0	0	0	0	0	1	0	0	0	0	0
v8	0	0	1	0	0	0	1	0	0	0	0	0	1
v9	0	0	0	1	0	0	0	0	0	0	0	0	1
v10	0	0	0	0	1	0	0	0	0	0	0	1	0
v11	0	0	0	0	0	1	0	0	0	0	0	1	0
v12	0	0	0	0	0	0	0	0	0	1	1	0	1
v13	0	0	0	0	0	0	0	1	1	0	0	1	0

TABLE III. ADJACENT MATRIX Y WITH NON-CAMERA NODES AND NON-CAMERA NODES

Y	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13
v1	0	0	0	0	0	0	1	0	0	0	0	0	0
v2	0	0	0	0	0	0	1	0	0	0	0	0	0
v3	0	0	0	0	0	0	0	1	0	0	0	0	0
v4	0	0	0	0	0	0	0	0	1	0	0	0	0
v5	0	0	0	0	0	0	0	0	0	1	0	0	0
v6	0	0	0	0	0	0	0	0	0	0	1	0	0
v7	1	1	0	0	0	0	0	0	0	0	0	0	0
v8	0	0	1	0	0	0	0	0	0	0	0	0	1
v9	0	0	0	1	0	0	0	0	0	0	0	0	1
v10	0	0	0	0	1	0	0	0	0	0	0	1	0
v11	0	0	0	0	0	1	0	0	0	0	0	1	0
v12	0	0	0	0	0	0	0	0	0	1	1	0	1
v13	0	0	0	0	0	0	0	1	1	0	0	1	0

TABLE I is adjacent matrix **X** consisted of p rows q columns which are $|\mathbf{A}| = p$ and $|\mathbf{V}| = q$. Element x_{ij} is defined as (1) in adjacent matrix **X**.

$$x_{ij} = \begin{cases} 1 & \text{There is the line which links camenra node } a_i \text{ and conjugation node } v_j. \\ 0 & \text{Other.} \end{cases} \quad (1)$$

TABLE II is adjacent matrix **Y** consisted of q rows q columns which are $|\mathbf{V}| = q$. Element y_{ij} is defined as (2) in adjacent matrix **Y** before satisfying all conditions.

$$y_{ij} = \begin{cases} 1 & \text{There is the line which links conjugation node } v_i \text{ and conjugation node } v_j. \\ 0 & \text{Other.} \end{cases} \quad (2)$$

Consider the problem of (c) in Fig.2 with Fig.3. Video cameras, a_2 and a_3, have same problem in Fig.3. When paying attention to non-camera nodes, v_7 and v_8, here, these nodes are adjacent with plural camera. This consists equations that summation of column v_7 in X and summation of column v_8 in X each other are larger than 1, if adjacent matrix X is used. And $y_{78} = y_{87} = 1$ also consists because neighbor of v_7 is v_8, if adjacent matrix Y. On this case, the neighbor relationship of non-camera nodes, v_7 and v_8, can be cut, and the neighbor relationship of camera nodes, from a_1 to a_3, and non-camera nodes, v_7 and v_8, can be left if it is assumed $y_{78} = y_{87} = 0$.

If the conditions (3) are satisfied, then Element y_{ij} and Element y_{ji} is replaced as $y_{ij} = y_{ji} = 0$.

$$\sum_{n=1}^{m} x_{ni} > 1, \sum_{n=1}^{m} x_{nj} > 1, y_{ij} = y_{ji} = 1 \quad (3)$$

TABLE III is adjacent matrix Y after satisfying the conditions (3) and resolved the problem of (c) in Fig.2. Summation of column of v_7 and summation of column of v_8 each other in adjacent matrix X are lager than 1. Therefore y_{78} and y_{87} are replaced by 0.

Matrix **Z** is made from transposed matrix **X**. It is possible to consider that x_{ij} indicates neighbor from camera node a_i to non-camera node v_j, and z_{ji} indicates neighbor from non-camera node v_j to camera node a_i. Therefore, $x_{ij} \times z_{ji} = 1$.

Next considering the case as shown on Fig. 5, the camera node a_i is neighbor to camera node a_j via non-camera node v_n.

The relation between camera node a_i and non-camera node v_n consists of $x_{in} = 1$ and the relation between non-camera node v_n and camera node a_i consists of $z_{nj} = 1$. In addition, it is considered that $x_{in} \times z_{nj} = 1$ consists, if camera node a_i can reach camera node a_j via non-camera node v_n. Therefore, it is possible to derive the relation between camera a_i and camera a_j with the above arithmetic expression, and it is possible to define that camera node a_i is neighbor to camera node a_j. If it is assumed that the element b_{ij} of adjacent matrix **B** indicates relation between camera node a_i and camera node a_j, the arithmetic expression via non-camera node v_n ($n = 1 \ldots m$) is possible to express as (4). However, if $i = j$ consists, then it makes $b_{ij} = 0$ because neighbor relation as $i = j$ indicates neighbor to camera itself.

$$b_{ij} = \sum_{n=1}^{m} x_{in} z_{nj} \quad \begin{cases} \geq 1 & a_i \text{ is adjacent to } a_j. \\ = 0 & a_i \text{ is not adjacent to } a_j. \end{cases} \quad (4)$$

Considering the case in Fig. 6, camera node a_i is neighbor to non-camera node v_j via non-camera node v_n.

The relation between camera node a_i and v_n consists of $x_{in} = 1$ and the relation between non-camera node v_n and non-camera node v_j consists of $y_{jn} = 1$ and transposed y_{jn} is also 1. In addition, it is considered that $x_{in} \times y_{jn} = 1$ consists, if camera node a_i can reach non-camera node v_j via non-camera node v_n. Therefore, it is possible to derive the relation between camera node a_i and non-camera node v_j with the above arithmetic expression, and it is possible to define that camera node a_i is neighbor to non-camera node v_j. If it is assumed that adjacent matrix element c_{ij} indicates relation between camera node a_i and non-camera node v_j, the arithmetic expression via non-camera node v_n ($n = 1 \ldots m$) is possible to express as (5).

$$c_{ij} = \sum_{n=1}^{m} x_{in} y_{jn} \quad \begin{cases} \geq 1 & a_i \text{ is adjacent to } v_j. \\ = 0 & a_i \text{ is not adjacent to } v_j. \end{cases} \quad (5)$$

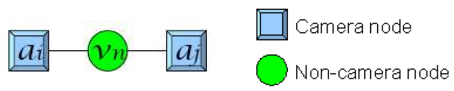

Figure 5. Two camera nodes via one non-camera node.

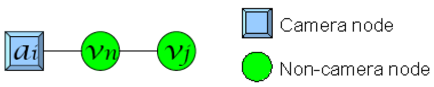

Figure 6. Camera node and non-camera node via one non-camera node.

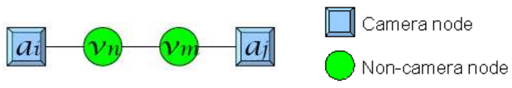

Figure 7. Two camera nodes via two non-camera nodes.

Considering the case on Fig. 7, the camera node a_i is neighbor to camera node a_j via two non-camera nodes, v_n and v_m.

If it is assumed that the element of adjacent matrix **D** is d_{ij}, it is possible to derive (6) under applying the result of Fig. 6.

$$d_{ij} = \sum_{n=1}^{m} c_{in} z_{nj} \begin{cases} \geq 1 & a_i \text{ is adjacent to } a_j . \\ = 0 & a_i \text{ is not adjacent to } a_j . \end{cases} \quad (6)$$

Than the above, when it calculates adjacent matrix **E** via *n* or more modes, it can use (7).

$$\mathbf{E} = \mathbf{X}(\mathbf{Y})^{n-1}\mathbf{X}^{\mathbf{T}} \begin{cases} \geq 1 & a_i \text{ is adjacent to } a_j . \\ = 0 & a_i \text{ is not adjacent to } a_j . \end{cases} \quad (7)$$

n is number by way of non-camera nodes.

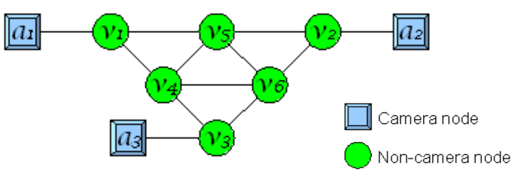

Figure 8. Graph that non-camera nodes are looped.

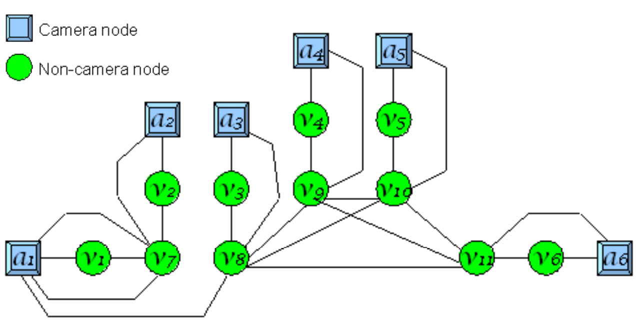

Figure 9. Graph without unnecessary non-camera nodes.

It considers the value of *n* on the above (7). It is difficult to decide whether or not the non-camera nodes are *n* and in the case on where it belongs to a loop as shown in Fig. 8. If there is no existence of a loop the problem will not occur. Then adjacent matrix **X'** and **Y'** are computed so that a diagram may be constituted from camera nodes and non-camera nodes without unnecessary non-camera nodes. Fig. 9 is the graph re-calculated from Fig. 3.

D. Removing of Unnecessary Non-camera Node

Consider the adjacent matrix **X'**, **Y'** computed without unnecessary nodes. The matrix is calculated with the following procedure.

In the case of adjacent matrix **X**, the procedure is stated below:

- Searches unnecessary non-camera node v_n in which camera node is not neighbor.
- Removes the column of the node v_n.

Adjacent matrix **X'** is computed from adjacent matrix **X** without the unnecessary nodes.

TABLE IV. UNNECESSARY NODES IN ADJACENT MATRIX **X**

X	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13
a1	1	0	0	0	0	0	1	1	0	0	0	0	0
a2	0	1	0	0	0	0	1	0	0	0	0	0	0
a3	0	0	1	0	0	0	0	1	0	0	0	0	0
a4	0	0	0	1	0	0	0	0	1	0	0	0	0
a5	0	0	0	0	1	0	0	0	0	1	0	0	0
a6	0	0	0	0	0	1	0	0	0	0	1	0	0

TABLE V. ADJACENT MATRIX **X'**

X	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11
a1	1	0	0	0	0	0	1	1	0	0	0
a2	0	1	0	0	0	0	1	0	0	0	0
a3	0	0	1	0	0	0	0	1	0	0	0
a4	0	0	0	1	0	0	0	0	1	0	0
a5	0	0	0	0	1	0	0	0	0	1	0
a6	0	0	0	0	0	1	0	0	0	0	1

TABLE VI. UNNECESSARY NODES IN ADJACENT MATRIX **Y**

Y	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13
v1	0	0	0	0	0	0	1	0	0	0	0	0	0
v2	0	0	0	0	0	0	1	0	0	0	0	0	0
v3	0	0	0	0	0	0	0	1	0	0	0	0	0
v4	0	0	0	0	0	0	0	0	1	0	0	0	0
v5	0	0	0	0	0	0	0	0	0	1	0	0	0
v6	0	0	0	0	0	0	0	0	0	0	1	0	0
v7	1	1	0	0	0	0	0	0	0	0	0	0	0
v8	0	0	1	0	0	0	0	0	0	0	0	0	1
v9	0	0	0	1	0	0	0	0	0	0	0	0	1
v10	0	0	0	0	1	0	0	0	0	0	0	1	0
v11	0	0	0	0	0	1	0	0	0	0	0	1	0
v12	0	0	0	0	0	0	0	0	0	1	1	0	1
v13	0	0	0	0	0	0	0	1	1	0	0	1	0

TABLE VII. MERGED RESULT FROM TABLE VI

Y	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}
v_1	0	0	0	0	0	0	1	0	0	0	0	0	0
v_2	0	0	0	0	0	0	1	0	0	0	0	0	0
v_3	0	0	0	0	0	0	0	1	0	0	0	0	0
v_4	0	0	0	0	0	0	0	0	1	0	0	0	0
v_5	0	0	0	0	0	0	0	0	0	1	0	0	0
v_6	0	0	0	0	0	0	0	0	0	0	1	0	0
v_7	1	1	0	0	0	0	0	0	0	0	0	0	0
v_8	0	0	1	0	0	0	0	1	1	1	1	1	1
v_9	0	0	0	1	0	0	0	1	1	1	1	1	1
v_{10}	0	0	0	0	1	0	0	1	1	1	1	1	1
v_{11}	0	0	0	0	0	1	0	1	1	1	1	1	1
v_{12}	0	0	0	0	0	0	0	1	1	1	1	0	1
v_{13}	0	0	0	0	0	0	0	1	1	1	1	1	1

TABLE VIII. ADJACENT MATRIX **Y'**

Y'	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}
v_1	0	0	0	0	0	0	1	0	0	0	0
v_2	0	0	0	0	0	0	1	0	0	0	0
v_3	0	0	0	0	0	0	0	1	0	0	0
v_4	0	0	0	0	0	0	0	0	1	0	0
v_5	0	0	0	0	0	0	0	0	0	1	0
v_6	0	0	0	0	0	0	0	0	0	0	1
v_7	1	1	0	0	0	0	0	0	0	0	0
v_8	0	0	1	0	0	0	0	1	1	1	1
v_9	0	0	0	1	0	0	0	1	1	1	1
v_{10}	0	0	0	0	1	0	0	1	1	1	1
v_{11}	0	0	0	0	0	1	0	1	1	1	1

When the data on TABLE I is considered for an example, the unnecessary non-camera nodes will be highlighted as shown on TABLE IV. The columns v_{12} and v_{13} represent the unnecessary non-camera nodes, and adjacent matrix **X'** become as TABLE V.

In the case of adjacent matrix **Y**, non-camera node v_n is extracted to compute adjacent matrix **X'**. The procedure is stated below:

- Search unnecessary non-camera node v_n in which camera node is not neighbor.
- Merge column of unnecessary non-camera node v_n and column of non-camera node v neighbor to v_n with OR.
- Merge row of unnecessary non-camera node v_n and row of non-camera node v neighbor to v_n with OR.
- Remove the row and column of the node v_n.

Adjacent matrix **Y'** is computed from adjacent matrix **Y** without the unnecessary nodes.

When TABLE III is considered for an example, the unnecessary non-camera nodes will be highlighted as shown on TABLE VI. The result of merge is TABLE VII. The rows and columns of v_{13} and v_{14} represent the unnecessary non-camera nodes. And the adjacent matrix **Y'** become TABLE VIII. It makes y'_{ij} $(i = j)$ to 0, because neighbor relation as $i = j$ indicates neighbor to itself.

E. Expression for Neighbor Definition

It is possible to derive neighbor definition as (8) from the result of the foregoing paragraphs. However, if $i = j$ then $\mathbf{E}_{ij} = 0$ because neighbor relation $i = j$ indicates neighbor to camera itself.

$$\mathbf{E} = \mathbf{X'Y'X'}^{\mathrm{T}} \begin{cases} \geq 1 & a_i \text{ is adjacent to } a_j . \\ = 0 & a_i \text{ is not adjacent to } a_j . \end{cases} \quad (8)$$

IV. EXAMINATION

The algorithm was tested on the diagram as shown on Fig.10. Three kind of view distance of camera is applied to Fig. 10 on this test. The applied Figures are shown from diagram A to C on from Fig. 11 to Fig. 13. The arrows on the diagrams indicate neighbor relation between camera nodes confirmed through observation. And calculated results are shown in TABLE IX, X, and XI. When the three calculated results and the arrows on the diagram A, B, and C are compared. The correct results were obtained.

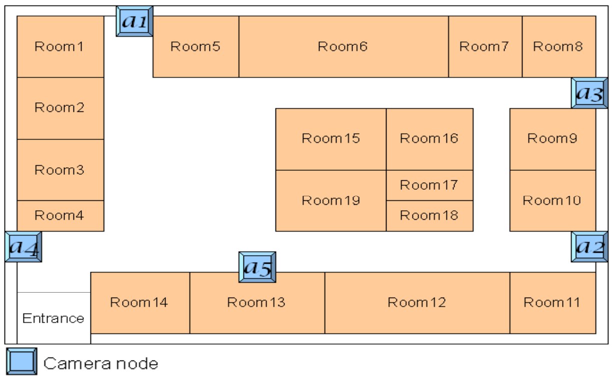

Figure 10. Base diagram for examination.

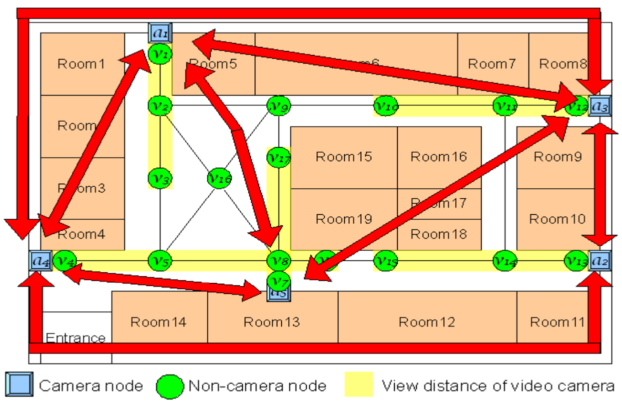

Figure 11. Diagram A.

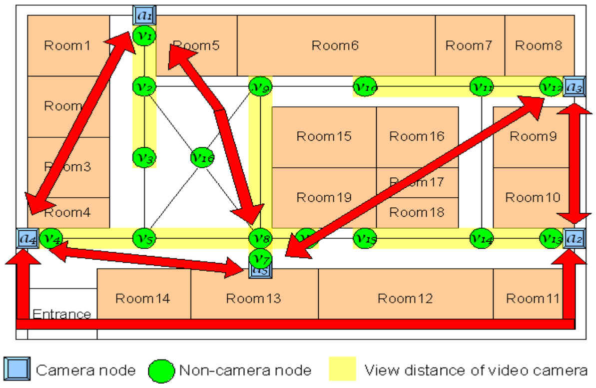

Figure 12. Diagram B.

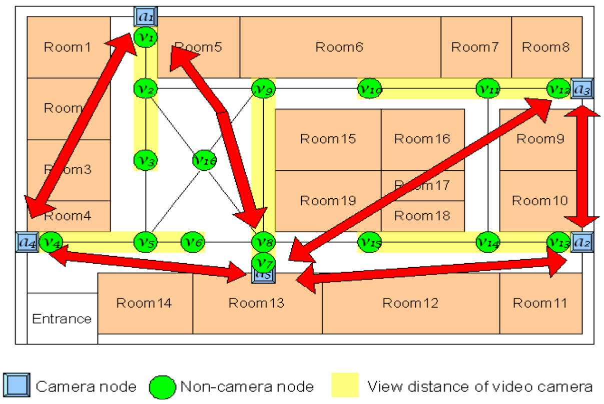

Figure 13. Diagram C.

TABLE IX. CALCULATED ADJACENT MATRIX **E** FOR DIAGRAM **A**

E	a1	a2	a3	a4	a5
a1	0	0	1	3	2
a2	0	0	1	1	0
a3	1	1	0	2	2
a4	3	1	2	0	5
a5	2	0	2	5	0

TABLE X. CALCULATED ADJACENT MATRIX **E** FOR DIAGRAM **B**

E	a1	a2	a3	a4	a5
a1	0	0	0	3	2
a2	0	0	1	1	0
a3	0	1	0	0	1
a4	3	1	0	0	4
a5	2	0	1	4	0

TABLE XI. CALCULATED ADJACENT MATRIX **E** FOR DIAGRAM **C**

E	a1	a2	a3	a4	a5
a1	0	0	0	2	2
a2	0	0	1	0	1
a3	0	1	0	0	1
a4	2	0	0	0	3
a5	2	1	1	3	0

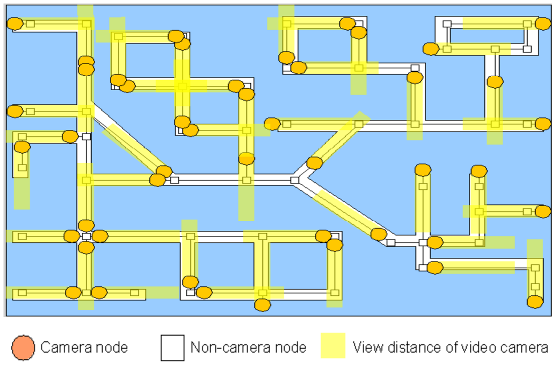

Figure 14. Complicated diagrams.

V. CONCLUSION

This algorithm is possible to determine the neighbor nodes easily, even when the view distance of the video camera changes and modification to the number of video camera installed occur. And the algorithm could determine the neighbor nodes quickly and correctly in the case of the complicated diagram in Fig.14. In addition, the mobile agent can efficiently decide unto which server the agent should transfer next correctly by utilizing the neighbor node information determined by this algorithm.

When it is considered that an agent transfers more efficiently, highly precise pursuit is realizable as using this algorithm and the information that indicates movement direction of pursued human on video camera. We will aim at the establishment of the higher-precision pursuit technique with the algorithm adding movement direction information in future.

REFERENCES

[1] 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 59th Chugoku branch union convention of the Institute of Electrical Engineers of Japan and Information Processing Society of Japan in Tottori, Tottori, 2008, pp. 485.

[2] 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 59th Chugoku branch union convention of the Institute of Electrical Engineers of Japan and Information Processing Society of Japan in Tottori, Tottori, 2008, pp. 486.

[3] 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.

[4] 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.

[5] N. Ishida, 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 59th Chugoku branch union convention of the Institute of Electrical Engineers of Japan and Information Processing Society of Japan in Tottori, Tottori, 2008, pp. 418.