Verification of Detection Methods for Robust Human Tracking System

Hiroto Kakiuchi
System Engineering Department,
Melco Power Systems Co., Ltd.
Kobe, Japan
Email: Kakiuchi.Hiroto@zs.MitsubishiElectric.co.jp

Takao Kawamura, Toshihiko Sasama, and Kazunori Sugahara
Graduate School of Engineering,
Tottori University.
Tottori, Japan
Email: {kawamura,sasama,sugahara}@ike.tottori-u.ac.jp

***Abstract*—Much recent research is concerned with overcoming limitations of existing video surveillance systems, particularly for use in automatic human tracking systems. This paper presents detection methods which detect a lost target person during track in automatic human tracking system. The detection methods utilize an algorithm which determines the position of neighbors in a system of video cameras. By utilizing this deployed position and the view distance of video cameras, this algorithm also determines the interrelationship cameras in such a network must have in an automatic human tracking system. The system is enhanced by a video monitoring system utilizing mobile agent technologies. Mobile agents are suitable for distributed processing and parallel processing, since they can monitor their own behavior and run on distributed computers. Multiple mobile agents in the system can track numerous people using information gathered from several neighboring video cameras at the same time. Searching a target person at random is irrational when a system is losing the target; additionally, difficulty of detection can arise if the deployed position and/or the view distance of video cameras vary due to other circumstances. Therefore, a robust computation not influenced by these circumstances is needed, and detection methods utilizing the above algorithm were developed to solve these concerns and to improve reliability of this system by re-detecting the lost target.**

Keywords-Detection Method; Human Tracking; Mobile Agent.

I. INTRODUCTION

Video surveillance systems are seeing widespread use in the remote monitoring of people. Principally, the video surveillance system is used as a security system because of its ability to track a particular person. If the function of the video surveillance system is extended to track numerous people, the demands of the system are extended in various ways. Two common examples of such uses are to search for lost children and to gather/analyze consumers' route pattern for marketing research of a retail establishment. Such video surveillance systems are referred to as "automatic human tracking systems" in this paper. Our aim is to show how automatic human tracking systems can be improved by resolving some of the problems of conventional video surveillance system.

Currently, existing video surveillance systems have many limitations to their capabilities. In one case, systems have difficulty isolating a number of people located at different position at the same time and track those people automatically. In another, the number of possible targeted people is limited by the extent of users' involvement in manually switching the view from one video camera to another. Although approaches do exist to increase the efficiency of identifying and tracking particular people in a system comprised of numerous surveillance positions, these approaches demand an increase in the workload of the user since it demands users to identify the target.

Some researchers have suggested solutions to the above problems. The first approach was to use an active camera to track a person automatically [1][2], thus the camera moves in a synchronized motion along with the projected movement of the targeted person. Since a method for correcting blurring image [3] is proposed, the active camera is available. This approach is capable of locating and tracking small number of people, but improvements must be made to facilitate the locating and tracking of larger numbers of people. Another common approach was to position the camera efficiently at strategic surveillance locations [4]. This is not possible in some situations due to the number of cameras that would be necessary for full coverage, and in such cases this approach is not feasible due to limited resources. A third approach involved the implementation of sensors to efficiently track a target with multiple cameras [5][6]. A fourth approach observes a target with multiple cameras called "Watching Station" [7]. These solution also meet with resource and local restrictions such as installation barriers and the amount of area to be monitored. A fifth approach discriminates a target from color of hair, skin and clothing [8]. This solution has weak from change of hue by shade, and has possibility of mistaken recognition.

A better approach to identify and track numerous targeted people at the same time involves image processing and installation of video cameras at any designated location. However, the concern then becomes the appropriateness of using a single server when locating numerous people, since the image processing increases server load. As such, a new type of system that is capable of more efficiently identifying and locating people must be developed. In this proposed system, utilizing mobile agent technologies, the ratio of mobile agents and tracked targets is directly proportional

[9][10][11][12]. According to many studies, an agent-based approach is appropriate for distributed systems and parallel processing [13][14][15], since mobile agents can transfer copies of themselves to other servers in the system. By working cooperatively, such a multi-agent system would be effective [16]. With distributed processing, mobile agent technologies are more effective and efficient than conventional video surveillance systems, assuming that a large number of servers with video camera are installed. If one mobile agent can track one person, then multiple mobile agents can track numerous people at the same time, and the server balances the load process of the operating mobile agent on each server with a camera. A video surveillance system enhanced with mobile agent technologies is called "Automatic Human Tracking System" [17][18]. In such a system, a mobile agent tracks a person captured by a video camera and a server process the data. The video camera and the server are treated as a single entity since the video camera and the server are deployed at the surveillance position. Upon initialization of a person as a target to track, a mobile agent is generated for that particular person. After verifying the features of the person [19], the mobile agent tracks the movement of the person by utilizing the neighbor camera/server location information.

In the automatic human tracking system, tracking function must be robust even if the system loses a target person. Present image processing is not perfect because a feature extraction like SIFT [20] has high accuracy but takes much processing time. The trade-off of accuracy and processing time is required for such a feature extraction algorithm. In addition, the speed a person walks is various and the person may be unable to be captured correctly in cameras. Therefore, it is necessary to re-detect a target person as tracking function even if the system loses the target. We propose two types of detection method to detect a target person in this paper. The detection methods compensate for the above weakness of feature extraction as a function of system. The detection methods also utilize "neighbor node determination algorithm" [21] to detect the target efficiently. The algorithm can determine neighbor camera/server location information without the location and view distance of video camera. Neighbor camera/servers are called "neighbor camera node/nodes" in this paper. The mobile agent can detect the target person efficiently with knowing the neighbor camera node location information.

On the following sections, Section II will be describing about overview of automatic human tracking system, Section III contains the overview of Neighbor node determination algorithm, Section IV explains the two types of detection method to detect a target person when the system is losing the target, Section V is the results of examination using the detection methods, and Section VI is the conclusion of the detection methods and feature subjects.

II. Overview of Automatic Human Tracking System

The system configuration of the automatic human tracking system is shown in Figure 1. It is assumed that the system is installed in a given building. Before a person is granted access inside the building, the person's information is registered in the system. Through a camera an image of the persons face and body is captured. Feature information is extracted from the image by SIFT and registered into the system. Any person who is not registered or not recognized by the system is not allowed to roam inside the building. This system is composed of an 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 target person's information, retrieving and displaying the information of the initiated mobile agents, and displaying video of the target entity. The agent management server records mobile agents' tracking information history, and provides the information to the agent monitoring terminal. The video recording server records all video images and provides the images to the agent monitoring terminal via request. The feature extraction server along with the video camera analyzes the entity image and extracts the feature information from the image.

A mobile agent tracks a target entity using the feature information and the neighbor nodes information. The number of mobile agents is in direct proportion to the number of the target entities. A mobile agent is initialized at the agent monitoring terminal and launched into the feature extraction server. The mobile agent extracts the features of a captured entity and compares it with the features already stored by the agent. If the features are equivalent, the entity is located by the mobile agent.

The processing flow of the proposed system is also shown in Figure 1. (i) First, a system user selects an entity on the screen of the agent monitoring terminal, and extracts the feature information of the entity to be tracked. (ii) Next, the feature information is used to generate a mobile agent per target which is registered into the agent management server. (iii) Then the mobile agent is launched from the terminal to the first feature extraction server. (iv) When the mobile agent catches the target entity on the feature extraction server, the mobile agent transmits information such as the video camera number, the discovery time, and the mobile agent identifier to the agent management server. (v) Finally, the mobile agent deploys a copy of itself to the neighbor feature extraction servers and waits for the person to appear. If the mobile agent identifies the person, the mobile agent notifies the agent management server of the information, removes the original and other copy agents, and deploys the copy of itself to the neighbor feature extraction servers again. Continuous tracking is realized by repeating the above flow.

The system architecture is shown in Figure 2. The GUI is

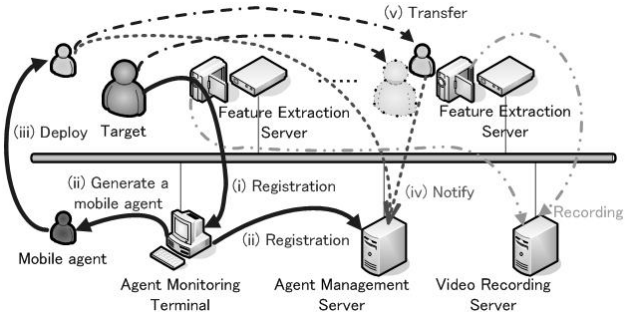

Figure 1. System Configuration and Process Flow.

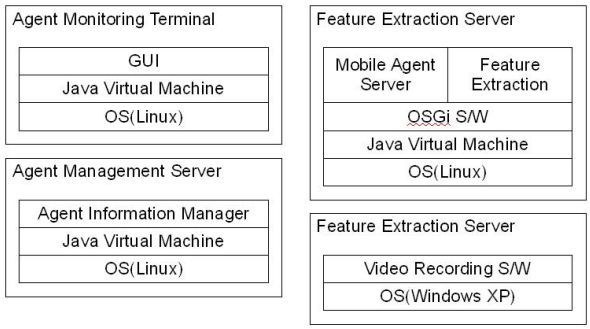

Figure 2. System Architecture.

operated only on the agent monitoring terminal. The GUI is able to register images of the entities and monitor the status of all the mobile agents. The mobile agent server is executed on the feature extraction server and allows the mobile agents to execute. The Feature extraction function is able to extract features of the captured entities, which is then utilized in the tracking of those entities as mobile agents. OSGi S/W [22] acts as a mediator for the different software, allowing the components to utilize each other. The Agent information manager manages all mobile agent information and provides the information to the agent monitoring terminal. The Video recording S/W records all video, and provides the video movie to agent monitoring terminal. Each PC is equipped with an Intel Pentium IV 2.0 GHz processor and 1 GB memory. The system has an imposed condition requirement that maximum execution time of feature judgment is 1 second and maximum execution time of mobile agent transfer is 200 milliseconds.

In this system, a simulator is also currently being developed in Java Language. The simulator consists of an image processing simulator and simulator tools. The simulator tools are an editor for the creation of target simulation routes and a simulation feature data creator. The simulator tools are shown in Figure 3 and Figure 4. Since it is difficult to place many cameras, the image processing simulator is performed on the feature extraction server instead of a genuine image processing function. In addition, this simulator changes a target entity's feature to a walking target entity by using a

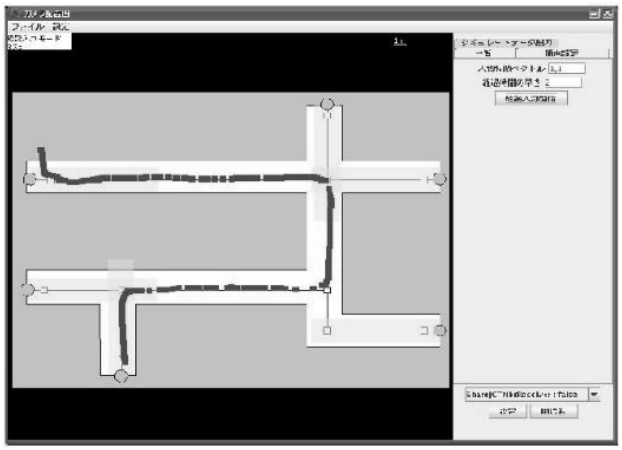

Figure 3. Editor of Simulation Route.

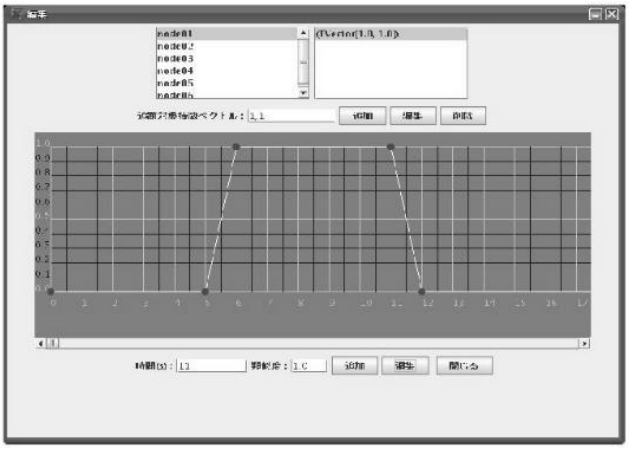

Figure 4. Creator of Simulation Feature Data.

simulation agent. The simulation agent is also a mobile agent that simulates the movement of a target entity and changes the target entity feature. The movement of the target entity is digitized by the editor of route simulation and the target entity features are digitized by the simulation feature data creator.

III. Overview of Neighbor Node Determination Algorithm

If a mobile agent tracks a target entity, the mobile agent has to know the deployed location of the video cameras in the system. However the abilities of the neighbor cameras are also determined by their view distances. A problem caused by a difference in the view distances can occur. This problem occurs when there is a difference in expected overlap of a view or an interrupt of view.

A scenario in which a neighbor video camera's location is influenced by view distance is shown in Figure 5. The upper side figures of Figure 5 show four diagrams portraying a floor plan with four video cameras each, considering

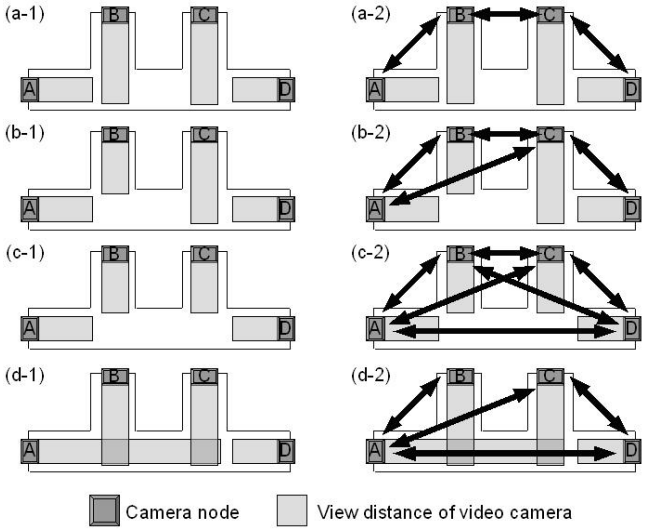

Figure 5. Influence by Change of View Distance of Video Cameras.

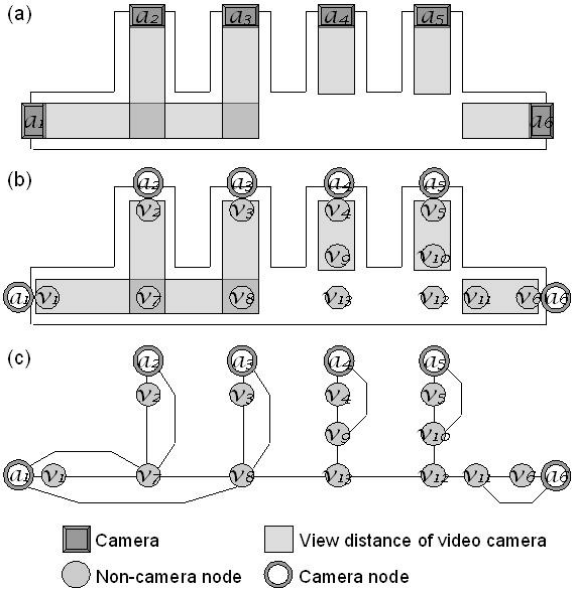

Figure 6. Figure that sets Non-camera Nodes.

the view distances of each video camera are different and assuming that the target entity to be tracked moves from the location of video camera A to video camera D. The underside figures of Figure 5 show neighbors of each video camera with arrows. The neighbor of video camera A in object a-1 of Figure 5 is video camera B but not C and not D as the arrows in object a-2 show. In object a-1 of Figure 5, video camera C and D are also not considered neighbors of video camera A, because video camera B blocks the view of video camera C and D. And the target entity can be captured at an earlier time on video camera B. But in the case of object b-1 of Figure 5, the neighbors of video camera A are video camera B and C but not camera D as the arrows in object b-2 of Figure 5 show. In the case of object c-1 of Figure 5, the neighbors of video camera A are all video cameras as the arrows in object c-2 of Figure 5 show. Thus neighbor video camera's location indicates the difference in view distances of video cameras. The case of object d-1 in Figure 5 is more complicated. The neighbors of video camera A in object d-1 of Figure 5 are video camera B, C, and D as the arrows in object d-2 of Figure 5 show. And video camera B is not considered the neighbor of video camera C. It is because video camera A exists as a neighbor between video camera B and C. When it is assumed that a target entity moves from A to D, the target entity is sure to be captured by video camera A, B, A, and C in that order.

This scenario indicates that the definition of "neighbor" cannot be determined clearly because the determination of the neighbor definition is influenced by the change of view distance and it becomes more complicated as the number of video cameras increases.

Neighbor node determination algorithm can easily determine the neighbor video camera's location without regard to the influence of view distances and any modification of the information of the currently installed cameras. The modification information is set in the system to compute neighbor video cameras on the diagram, which is expressed as a graph. Nodes are used to compute neighbor video camera's information in this algorithm. The nodes are defined as camera node and non-camera node. Camera node is the location of video camera that is labeled as camera node. The nodes are defined as $A = \{a_1, a_2, ..., a_p\}$. This node is also a server with video camera. Non-camera node is defined as $V = \{v_1, v_2, ..., v_q\}$. The conditions of a non-camera node are stated below; i) either of crossover, corner, terminal of passage, ii) the position where a video camera is installed, or iii) the end point of the view distance of a video camera. In addition, the point where the above conditions are overlapped is treated as one node. When the view distance of the video camera reaches a 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, those nodes are specified as neighbors. Figure 6 shows an example of these definitions applied and shows the view distances of the video cameras.

The algorithm accomplishes this using an adjacency matrix. Two kinds of adjacency matrix are used by the algorithm. One is an adjacency matrix X made from camera nodes' locations as rows and non-camera nodes' locations as columns. Element x_{ij} of matrix X is defined as (1). Table I is the adjacency matrix X which is a table representation based on the object (c) of Figure 6.

$$x_{ij} = \begin{cases} 1 & \textit{There is the line which links} \\ & \textit{camera node } a_i \textit{ and non-camera node } v_j. \\ 0 & \textit{There is no link.} \end{cases} \quad (1)$$

Table I
ADJACENCY MATRIX X WITH ELEMENT x_{ij}

X	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}
a_1	1	0	0	0	0	0	1	1	0	0	0	0	0
a_2	0	1	0	0	0	0	1	0	0	0	0	0	0
a_3	0	0	1	0	0	0	0	1	0	0	0	0	0
a_4	0	0	0	1	0	0	0	0	1	0	0	0	0
a_5	0	0	0	0	1	0	0	0	0	1	0	0	0
a_6	0	0	0	0	0	1	0	0	0	0	1	0	0

Table II
ADJACENCY MATRIX Y WITH ELEMENT y_{ij}

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	0	0	0	0	0	1
v_9	0	0	0	1	0	0	0	0	0	0	0	0	1
v_{10}	0	0	0	0	1	0	0	0	0	0	0	1	0
v_{11}	0	0	0	0	0	1	0	0	0	0	0	1	0
v_{12}	0	0	0	0	0	0	0	0	0	1	1	0	1
v_{13}	0	0	0	0	0	0	0	1	1	0	0	1	0

Another one is as adjacency matrix Y made from non-camera nodes' location as rows and columns. Element y_{ij} of matrix Y is defined as (2). Table II is the adjacency matrix Y which is a table representation based on the object (c) of Figure 6.

$$y_{ij} = \begin{cases} 1 & \textit{There is the line which links} \\ & \textit{two non-camera nodes, } v_i \textit{ and } v_j. \\ 0 & \textit{There is no link or (3) is satisfied.} \end{cases} \quad (2)$$

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

The neighbor information for video cameras is calculated from the connection information of non-camera nodes by using adjacency matrix X and Y.

Below is the algorithm to determine neighbor nodes: i) Set camera nodes and non-camera nodes on the diagram as shown in object (b) of Figure 6. ii) Transform the diagram to a graph as shown in object (c) of Figure 6. iii) Generate an adjacency matrix X from camera node locations and non-camera node locations on the graph, and generate an adjacency matrix Y from non-camera node locations on the graph. Adjacency matrix X indicates that rows are camera nodes and columns are non-camera nodes. Adjacency matrix Y indicates that rows and columns are non-camera nodes, which results in adjacency matrix Y resolving an overlap problem of view distances between video cameras. iv) Calculate adjacency matrix X' and Y' by excluding unnecessary non-camera nodes from adjacency matrix X and Y. v) Calculate neighbor's location matrix by multiplying adjacency matrix and transposed matrix X'^T. This neighbor's location matrix is the neighbour's node information. An unnecessary non-camera node is a non-camera node which has no camera node as a neighbor. Adjacency matrix X' and Y' are computed without unnecessary nodes, and using the procedure shown later. There are reasons why it might be better to include the unnecessary nodes in the diagram from the beginning as we have done. Since the risk of committing an error will be higher as the diagram becomes larger, we include the unnecessary nodes from the beginning and remove them at the end. Table III is the adjacency matrix X' without unnecessary nodes from the adjacency matrix X, and Table IV is the adjacency matrix Y' without unnecessary nodes from the adjacency matrix Y. Finally, matrix E which indicates the neighbor nodes is derived as (4). In addition, e_{ij} which is a value of element of E indicates number of route for reaching from camera node a_i to camera node a_j.

Table III
ADJACENCY MATRIX X'

X'	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}
a_1	1	0	0	0	0	0	1	1	0	0	0
a_2	0	1	0	0	0	0	1	0	0	0	0
a_3	0	0	1	0	0	0	0	1	0	0	0
a_4	0	0	0	1	0	0	0	0	1	0	0
a_5	0	0	0	0	1	0	0	0	0	1	0
a_6	0	0	0	0	0	1	0	0	0	0	1

Table IV
ADJACENCY 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

$$E = X'Y'X'^T \begin{cases} \geq 1 & a_i \textit{ is neighbor node to } a_j \\ = 0 & a_i \textit{ is not neighbor node to } a_j \end{cases} \quad (4)$$

IV. DETECTION METHODS

The detection method in this paper is used to re-detect a target when the automatic tracking system loses the target.

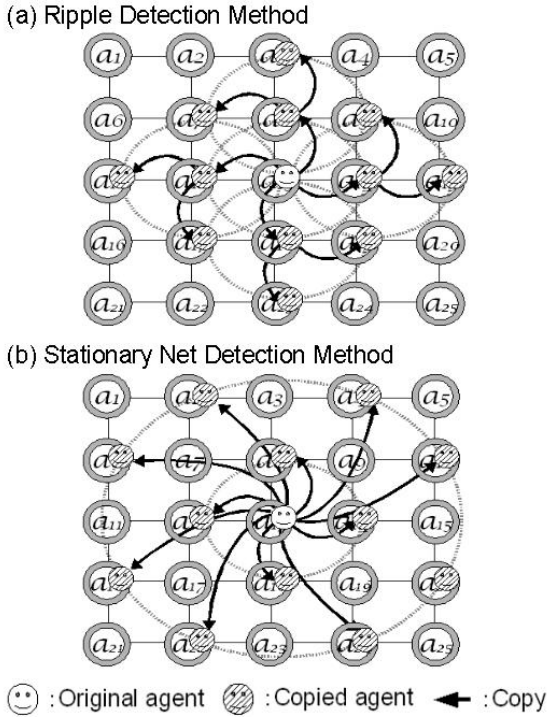

Figure 7. Detection Methods.

This method improves the tracking function, because an individual can not be accurately identified in the current image processing. As such the reliability of the system is further improved, because it enhances the continuous tracking function and re-detection of the target even if a target is lost for a long period of time. In this paper, if a target is not captured within a certain period of time, the mobile agent then concludes that the target is lost. On such case the system can also conclude that the target is lost.

We are proposing two types of detection method: (a) "Ripple detection method" and (b) "Stationary net detection method". These methods are shown in Figure 7.

Ripple detection method widens a search like a ripple from where an agent lost a target to give top priority to re-detect. This method has a feature that the discovery time becomes shorter and usual tracking can resume more quickly, if the target exists near where the agent lost. In addition, this method deletes other agents immediately after discovering the target, and suppresses the waste of the resource. The Ripple detection method is developed and is examined in search propriety. In the Ripple detection method, the neighbor camera nodes are shown as (5).

$$E1 = E = X'Y'X'^{T} \tag{5}$$

When a mobile agent lost a target, copy agents are deployed to the next nodes of (5) expressed by (6), and search is started. E^2 shows next neighbor camera nodes, because the elements of E^2 larger than 1 can be reached if the elements are larger than 1. Therefore, excepting neighbor node information E of camera nodes, automatic human tracking system uses a minimum resource by deploying copy agents.

$$E2 = E^2 - E1 = E^2 - E \tag{6}$$

Similarly, it becomes like (7) and (8) to calculate the next camera node further.

$$E3 = E^3 - (E2 + E1) = E^3 - E^2 \tag{7}$$

$$E4 = E^4 - (E3 + E2 + E1) = E^4 - E^3 \tag{8}$$

As mentioned above, the equation (9) is derived when deploying agents efficiently to the n next camera nodes. n is larger than 2 and is incremented one by one when this equation is used for detection.

$$En = E^n - \sum_{m=1}^{n-1} Em = E^n - E^{n-1} \tag{9}$$

Stationary net detection method widens a search like setting a stationary net with the Neighbor node determination algorithm from where an agent lost a target to give top priority to re-detect. This method uses (10) in the algorithm.

$$E = X'(Y')^n X'^{T} \begin{cases} \geq 1 & a_i \textit{ is neighbor node to } a_j \\ = 0 & a_i \textit{ is neighbor node to } a_j \end{cases} \tag{10}$$

In this equation, adjacency matrix E indicates the node that can reach via n non-camera nodes and n is always set to $n \geq 2$. In this method, the coefficient n is set to $n = 4$ because camera nodes are set with a certain interval. The interval between cameras in the real system may be close, but in that case, number of non-camera nodes between the cameras decreases. Therefore it is enough interval to re-detect a target if n consists of $n \geq 4$. This method has a feature that agents are deployed to neighbor camera nodes via n next non-camera nodes and catch a target like a stationary net. In addition, this method also deletes other agents immediately after discovering the target, and suppresses the waste of the resource. The Stationary net detection method is developed and is examined in search property. In the Stationary net detection method, the neighbor camera nodes are shown as (11).

$$E1 = E = X'Y'X'^{T} \tag{11}$$

When a mobile agent lost a target, copy agents are deployed to the next nodes of (11) expressed by (12), and search is started. $X'Y'^{2}X'^{T}$ shows neighbor camera nodes via two non-camera nodes, because the elements of $X'Y'^{2}X'^{T}$ larger than 1 can be reached if the elements are larger than 1. If copy agents are deployed at each camera nodes via non-camera nodes more than two, detection range of target widens. And, excepting neighbor node information E

of camera nodes, automatic human tracking system uses a minimum resource by deploying copy agents.

$$E2 = X'Y'^2X'^T - E1 \tag{12}$$

Similarly, it becomes like (13) and (14) to calculate the next camera node of more wide range.

$$E3 = X'Y'^3X'^T - (E2 + E1) \tag{13}$$

$$E4 = X'Y'^4X'^T - (E3 + E2 + E1) \tag{14}$$

As mentioned above, the equation (15) is derived when deploying agents efficiently to the next camera nodes via n non-camera nodes. n is larger than 2 and is incremented one by one when this equation is used for detection.

$$En = X'Y'^nX'^T - \sum_{m=1}^{n-1} Em = X'Y'^nX'^T - X'Y'^{n-1}X'^T \tag{15}$$

V. Examination

Examination environment for the Ripple detection method and the Stationary net detection method is shown in Figure 8 and Figure 9. There are twelve camera nodes in the environment of floor map 1, and there are fourteen camera nodes in the environment of floor map 2. Here, the following conditions are set in order to examine the effectiveness of these detection methods. i) Camera nodes are arranged on latticed floor, $56m \times 56m$. ii) View distance of camera is set to $10m$ in one direction. iii) Identification of a target in the image processing does not fail when re-detecting. iv) Walking speed of the target is constant. v) Only one target is searched. vi) The target moves only forward without going back. In the case of the floor map 1, the target moves following the order of a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}, a_{12} and a_1. In the case of the floor map 2, the target moves following the order of a_1, a_2, a_4, a_5, a_7, a_9, a_{10}, a_{11} and a_1. In the examination, the time that an agent concludes a failure of tracking is same as search cycle time. The search cycle time is defined as the time concluded that an agent can not discover a target. The search cycle time is prepared using 3 patterns 12 seconds, 9 seconds and 6 seconds. Walking speed of the target is prepared using 3 patterns $1.5m/s$, $2m/s$ and $3m/s$. And search of target is prepared that an agent loses a target at a_7 and the agent starts a search in the situation that the target has already moved to a_8. Furthermore, Stationary net detection method is examined by 3 patterns $n = 2$, $n = 3$ and $n = 4$, because of confirming effectiveness by number of non-camera nodes. On each floor map, using 12 patterns of such combination by each walking speed, discovery time and the number of agents are measured. Generally, the walking speed of a person is around $2.5m/s$, and the two types of walking speed, $2m/s$ and $3m/s$, used by the target which was examined are almost equivalent to the walking speed of general person.

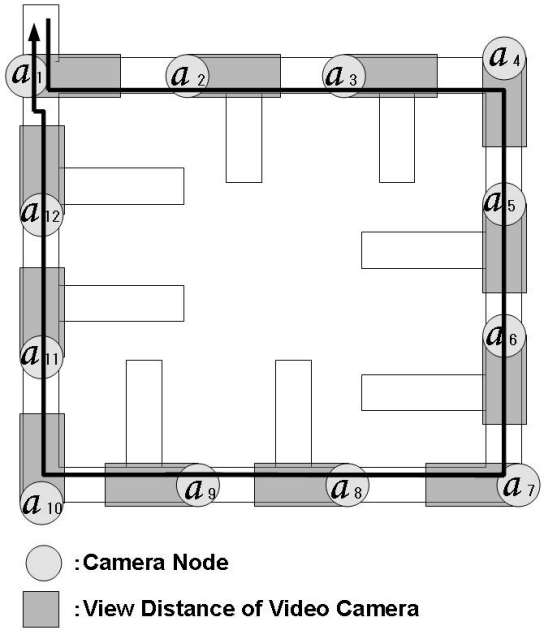

Figure 8. Floor Map 1 for Examination of Detection Methods.

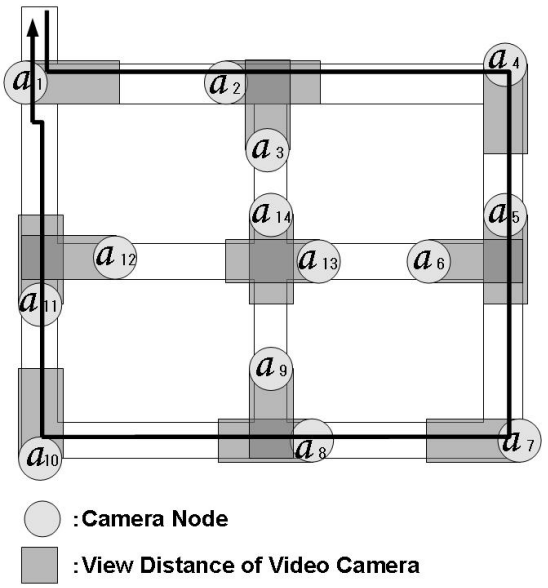

Figure 9. Floor Map 2 for Examination of Detection Methods.

And walking speed, $1.5m/s$, is very slow from the walking speed of general person.

The results of the measurement on the floor map 1 are shown in Table V, Table VI and Table VII. The results of the measurement on the floor map 2 are shown in Table VIII, Table IX and Table X. They are a mean value of 5 measurements.

The result of the Ripple detection method shows that the discovery time becomes shorter and usual tracking can resume more quickly, if the target exists near where the agent

Table V
DETECTION TIME ON FLOOR MAP 1 BY WALKING SPEED 1.5M/S

Walking Speed(1.5m/s)		Ripple	Stationary Net (n=2)	Stationary Net (n=3)	Stationary Net(n=4)
Search Cycle (12s)	Number of Agents	6	12	12	6
	Discovery Time (s)	20.6	–	–	20.5
Search Cycle (9s)	Number of Agents	6	12	6	6
	Discovery Time (s)	20.5	–	20.5	20.5
Search Cycle (6s)	Number of Agents	7	6	6	7
	Discovery Time (s)	20.5	20.5	20.5	20.5

Table VI
DETECTION TIME ON FLOOR MAP 1 BY WALKING SPEED 2M/S

Walking Speed(2m/s)		Ripple	Stationary Net (n=2)	Stationary Net (n=3)	Stationary Net(n=4)
Search Cycle (12s)	Number of Agents	6	12	12	6
	Discovery Time (s)	15.5	–	–	15.5
Search Cycle (9s)	Number of Agents	6	12	12	6
	Discovery Time (s)	15.5	–	–	15.4
Search Cycle (6s)	Number of Agents	7	12	6	7
	Discovery Time (s)	16.5	–	15.5	15.7

lost. But, if the walking speed of a target is faster, the agent will become difficult to discover the target.

The result of the Stationary net detection method shows that the agent can discover a target if coefficient n has larger value, even if the walking speed of a target is faster. And it is not enough interval to re-detect a target if n consists of $n \leq 3$ and it is not enough time to re-detect the target if the search cycle time is shorter.

From the result of measurement on the floor map 1, if the Stationary net detection method uses coefficient $n = 4$, there is not the difference of efficiency between the Ripple detection method and the Stationary net detection method. However, from the result of measurement on the floor map 2, if a floor map is complicated, the discovery time of the Stationary net detection method becomes shorter than the discovery time of the Ripple detection method and the number of agents of the Stationary net detection method becomes less than the number of agents of the Ripple detection method.

On the whole, the result of both methods shows that a number of agents decreases by searching a target near search cycle but the agents can not search the target if the search cycle time is longer than the waking speed. In addition based on the results, when the walking speed is faster, the discovery time is shortened or equal and the number of agents decreases or is equal.

Table VII
DETECTION TIME ON FLOOR MAP 1 BY WALKING SPEED 3M/S

Walking Speed(3m/s)		Ripple	Stationary Net (n=2)	Stationary Net (n=3)	Stationary Net(n=4)
Search Cycle (12s)	Number of Agents	12	12	12	12
	Discovery Time (s)	–	–	–	–
Search Cycle (9s)	Number of Agents	5.9	12	12	6
	Discovery Time (s)	11.7	–	–	11.5
Search Cycle (6s)	Number of Agents	6	12	12	6
	Discovery Time (s)	11.7	–	–	11.6

Table VIII
DETECTION TIME ON FLOOR MAP 2 BY WALKING SPEED 1.5M/S

Walking Speed(1.5m/s)		Ripple	Stationary Net (n=2)	Stationary Net (n=3)	Stationary Net(n=4)
Search Cycle (12s)	Number of Agents	14	14	10	12.2
	Discovery Time (s)	49.5	–	31.8	32.6
Search Cycle (9s)	Number of Agents	13.8	10	10	13
	Discovery Time (s)	33.7	32.3	32.3	33
Search Cycle (6s)	Number of Agents	13.9	10	13.2	14
	Discovery Time (s)	32.2	32	32.4	33

Table IX
DETECTION TIME ON FLOOR MAP 2 BY WALKING SPEED 2M/S

Walking Speed(2m/s)		Ripple	Stationary Net (n=2)	Stationary Net (n=3)	Stationary Net(n=4)
Search Cycle (12s)	Number of Agents	14	14	10	10
	Discovery Time (s)	–	–	31.1	25.3
Search Cycle (9s)	Number of Agents	14	14	10	13
	Discovery Time (s)	35.9	–	25.6	25.2
Search Cycle (6s)	Number of Agents	13.8	10	13	14
	Discovery Time (s)	24.8	25.3	24.8	25.8

Table X
DETECTION TIME ON FLOOR MAP 2 BY WALKING SPEED 3M/S

Walking Speed(3m/s)		Ripple	Stationary Net (n=2)	Stationary Net (n=3)	Stationary Net(n=4)
Search Cycle (12s)	Number of Agents	14	14	14	9.8
	Discovery Time (s)	–	–	–	18
Search Cycle (9s)	Number of Agents	14	14	14	10
	Discovery Time (s)	–	–	–	18.2
Search Cycle (6s)	Number of Agents	14	14	10.4	12.4
	Discovery Time (s)	25.9	–	18.2	19

VI. Conclusion

We propose the Ripple detection method and the Stationary net detection method. These are examined using a image processing simulator, because an individual can not be accurately identified in the current image processing. And in addition, the image processing simulator can be stabilized the accuracy of image processing and the simulator can be effective to examine each method correctly. These detection methods can search a lost target but a search cycle has to be within $(\textit{walking speed} \times \textit{distance between cameras})$. These methods can be efficient to detect a target if the search cycle is near the walking speed. A mobile agent can keep tracking a target by using these detection methods if the agent lose the target. In addition, from the examination results, the Stationary net detection method can detect a target faster than the Ripple detection method. And the Stationary net detection method can use smaller number of agents than the Ripple detection method. Because the Ripple detection method searches a target by widening a search gradually but the Stationary net detection method can widen a search efficiently by the Neighbor node determination algorithm.

We will research more efficient detection to improve the automatic human tracking system. And we are considering to compute a movement direction of a captured target, to improve efficiency of tracking by using the direction information to the tracking, and to improve the detection by using the number of route which is element of neighbor node information E. In addition, the accuracy of image processing has to be improved more to track a target more accurately. We are considering to improve image processing program by using PCA-SIFT [23] or SURF [24] algorithm.

Acknowledgment

The authors would like to thank Tadaaki Shimizu, Yusuke Hamada, Naoki Ishibashi, Shinya Iwasaki, Hirotoshi Okumura, Masato Hamamura, Shingo Iiyama in Tottori university.

References

[1] K. Terashita, N. Ukita, and M. Kidode, "Efficiency improvement of probabilistic-topological calibration of widely distributed active cameras," *IPSJ SIG Technical Report*, vol. 2009–CVIM–166, pp. 241–248, 2009.

[2] Y. Kawaguchi, A. Shimada, D. Arita, and R. Taniguchi, "Object trajectory acquisition with an active camera for wide area scene surveillance," *IPSJ SIG Technical Report*, vol. 2008–CVIM–163, pp. 1306–1311, 2008.

[3] H. Yin and I. Hussain, "Independent component analysis and nongaussianity for blind image deconvolution and deblurring," *Integrated Computer-Aided Engineering*, vol. 15, no. 3, pp. 219–228, 2008.

[4] U. M. Erdem and S. Sclaroff, "Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements," *CVIO2006*, vol. 103, no. 3, pp. 156–169, 2006.

[5] Y. Yao, C. H. Chen, B. Abidi, D. Page, A. Koschan, and M. Abidi, "Sensor planning for automated and persistent object tracking with multiple cameras," *CVPR2008*, 2008.

[6] ——, "Sensor planning for ptz cameras using the probability of camera overload," *ICPR2008*, 2008.

[7] A. Nakazawa, S. Hiura, H. Kato, and S. Inokuchi, "Tracking multiple persons using distributed vision systems," *Journal of Information Processing Society of Japan*, vol. 42, no. 11, pp. 2699–2710, November 2001.

[8] T. Darrell, G. Gordon, M. Harville, and J. Woodfill, "Integrated person tracking using stereo, color, and pattern detection," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 175–185, June 2000.

[9] D. B. Lange and M. Oshima, "Seven good reasons for mobile agents," *Communications of the ACM*, vol. 42, no. 3, pp. 88–89, 1999.

[10] R. S. Gray, G. Cybenko, D. Kotz, R. A. Peterson, and D. Rus, "D' agents: Applications and performance of a mobile-agent system," *Software: Practice and Experience*, vol. 32, no. 6, pp. 543–573, 2002.

[11] S. Motomura, T. Kawamura, and K. Sugahara, "Maglog: A mobile agent framework for distributed models," in *Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems*, 2005, pp. 414–420.

[12] T. Kawamura, S. Motomura, and 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*, 2005, pp. 486–491.

[13] G. Cabri, L. Leonardi, and F. Zambonelli, "Mobile-agent coordination models for internet applications," *Computer*, vol. 33, no. 2, pp. 82–89, February 2000.

[14] G. Valetto, G. Kaiser, and G. S. Kc, "A mobile agent approach to process-based dynamic adaptation of complex software systems," *Lecture Notes in Computer Science*, vol. 2077, pp. 102–116, 2001.

[15] N. R. Jennings, "An agent-based approach for building complex software systems," *Communications of the ACM*, vol. 44, no. 4, pp. 35–41, April 2001.

[16] D. Monticolo, V. Hilaire, S. Gomes, and A. Koukam, "A multi-agent system for building project memories to facilitate the design process," *Integrated Computer-Aided Engineering*, vol. 15, no. 1, pp. 3–20, January 2008.

[17] H. Kakiuchi, Y. Hamada, T. Kawamura, T. Shimizu, and 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*. Institute of Electrical Engineers of Japan and Information Processing Society of Japan, 2008, p. 485.

[18] Y. Hamada, S. Iwasaki, H. Kakiuchi, T. Kawamura, and 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*. Institute of Electrical Engineers of Japan and Information Processing Society of Japan, 2008, p. 486.

[19] N. Ishibashi, Y. Hamada, H. Kakiuchi, T. Shimizu, T. Kawamura, and 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*. Institute of Electrical Engineers of Japan and Information Processing Society of Japan, 2008, p. 418.

[20] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[21] H. Kakiuchi, T. Kawamura, T. Shimizu, and K. Sugahara, "An algorithm to determine neighbor nodes for automatic human tracking system," in *IEEE International Conference on Electro/Information Technology*. IEEE, 2009, pp. 96–102.

[22] Open Service Gateway Initiative Alliance, *OSGi Alliance Specifications OSGi Service Platform Release 1*, last access May 2011. [Online]. Available: http://www.osgi.org/Specifications/HomePage

[23] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," *Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 506–513, 2004.

[24] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding (CVIU)*, vol. 110, no. 3, pp. 346–359, 2008.