2

A Construction Method for Automatic Human Tracking System with Mobile Agent Technology

Hiroto Kakiuchi, Kozo Tanigawa,
Takao Kawamura and Kazunori Sugahara
Melco Power Systems Co., Ltd/Graduate School of Tottori University
Japan

1. Introduction

Human tracking systems that can track a specific person is being researched and developed aggressively, since the system is available for security and a flexible service like investigation of human behaviour. For example, Terashita, Kawaguchi and others propose the method for tracking an object captured by simple active video camera (Terashita et al. 2009; Kawaguchi et al. 2008), and Yin and others propose the solution of the problem of the blurring of the active video camera (Yin et al. 2008). Tanizawa and others propose a mobile agent framework that can become the base of a human tracking system (Tanizawa et al. 2002). These are component technology, and they are available in construction of the human tracking system. On the other hand, Tanaka and others propose a human tracking system using the information from video camera and sensor (Tanaka et al. 2004), and Nakazawa and others propose a human tracking system using recognition technique which recognizes same person using multiple video cameras at the same time (Nakazawa et al. 2001). However, although these proposed systems are available as human tracking system, the systems are constructed under fixed camera position and unchanged photography range of camera. On the other hand, there are several researches to track people with active cameras. Wren and others propose a class of hybrid perceptual systems that builds a comprehensive model of activity in a large space, such as a building, by merging contextual information from a dense network of ultra-lightweight sensor nodes with video from a sparse network of high-capability sensors. They explore the task of automatically recovering the relative geometry between an active camera and a network of one-bit motion detectors. Takemura and others propose a view planning of multiple cameras for tracking multiple persons for surveillance purposes (Takemura et al. 2007). They develop a multi-start local search (MLS)-based planning method which iteratively selects fixation points of the cameras by which the expected number of tracked persons is maximized. Sankaranarayanan and others discuss the basic challenges in detection, tracking, and classification using multiview inputs (Sankaranarayanan et al. 2008). In particular, they discuss the role of the geometry induced by imaging with a camera in estimating target characteristics. Sommerlade and others propose a consistent probabilistic approach to control multiple, but diverse active cameras concertedly observing a scene (Sommerlade et al. 2010). The cameras react to objects moving about, arbitrating conflicting interests of target resolution and trajectory accuracy, and the cameras anticipate the appearance of new targets. Porikli and others propose an automatic

object tracking and video summarization method for multi-camera systems with a large number of non-overlapping field-of-view cameras is explained (Porikli et al. 2003). In this framework, video sequences are stored for each object as opposed to storing a sequence for each camera.

Thus, these studies are efficient as the method to track targets. 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`` (Lowe. 2004) 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. In this chapter, a construction method of human tracking system including the detection method is proposed for realistic environment using active camera like the above mentioned. And the system constructed by the method can continuously track plural people at the same time. 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" 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 chapter. The mobile agent (Lange et al. 1999; Cabri et al. 2000; Valetto et al. 2001; Gray et al. 2002; Motomura et al. 2005; Kawamura et al. 2005) can detect the target person efficiently with knowing the neighbor camera node location information.In this chapter, the algorithm which can determine the neighbor node even if the view distance of video camera changes is also proposed.

2. System configuration

The system configuration of the automatic human tracking system is shown in Fig. 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

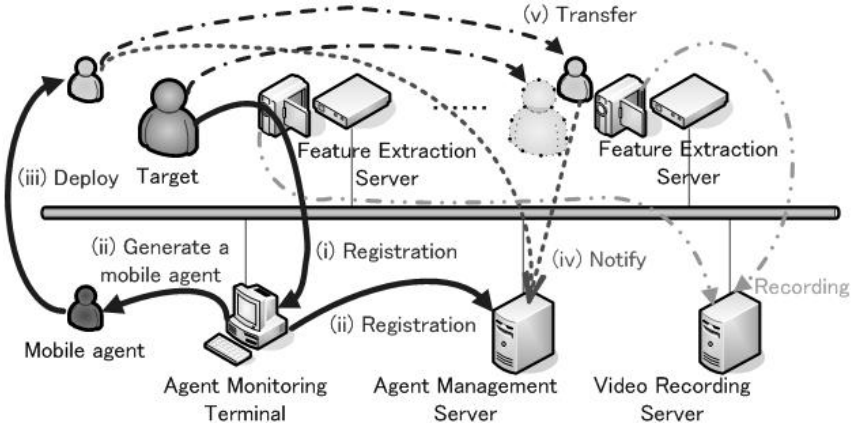

Fig. 1. System configuration and processing flow.

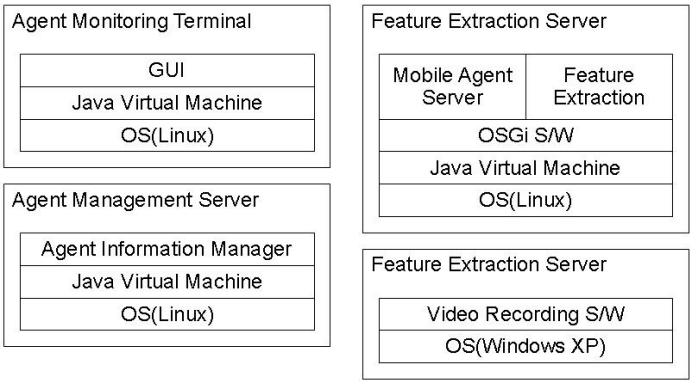

Fig. 2. System architecture.

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 Fig. 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 Fig. 2. The GUI is 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 (Open Service Gateway Initiative Alliance) S/W 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.

3. Influence by change of view distance of video camera

Here is indicated a problem that a change of view distance of video camera makes change for neighbor cameras. And a solution for the problem is also indicated.

3.1 Problem of influence by change of view distance of video camera

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 Fig.3. The upper side figures of Fig.3 show four diagrams portraying a floor plan with four video cameras each, considering 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 Fig.3 show neighbors of each video camera with arrows. The neighbor of video camera A in object (a-1) of Fig.3 is video camera B but not C and not D as the arrows in object (a-2) show. In object (a-1) of Fig.3, 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 Fig.3, the neighbors of video camera A are video camera B and C but not camera D as the arrows in object (b-2) of Fig. 3 show. In the case of object (c-1) of Fig.3, the neighbors of video camera A are all video cameras as the arrows in object (c-2) of Fig.3 show. Thus neighbor video camera's location indicates the difference in view distances of video cameras. The case of object (d-1) in Fig.3 is more complicated. The neighbors of video camera A in object (d-1) of Fig.3 are video camera B, C, and D as the arrows in object (d-2) of Fig.3 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.

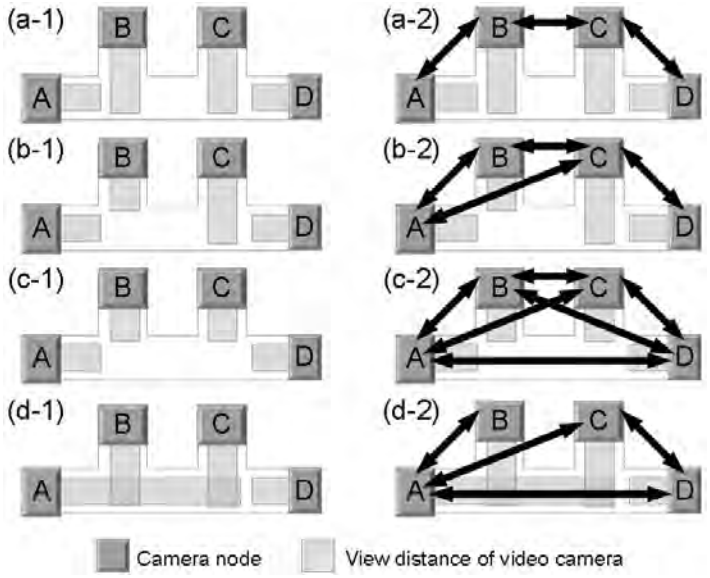

Fig. 3. Example of influence by change of view distance.

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.

3.2 Neighbor node determination algorithm to resolve the problem

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. Fig.4 shows an example of these definitions applied and shows the view distances of the video cameras.

The algorithm accomplishes neighbor node determination 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

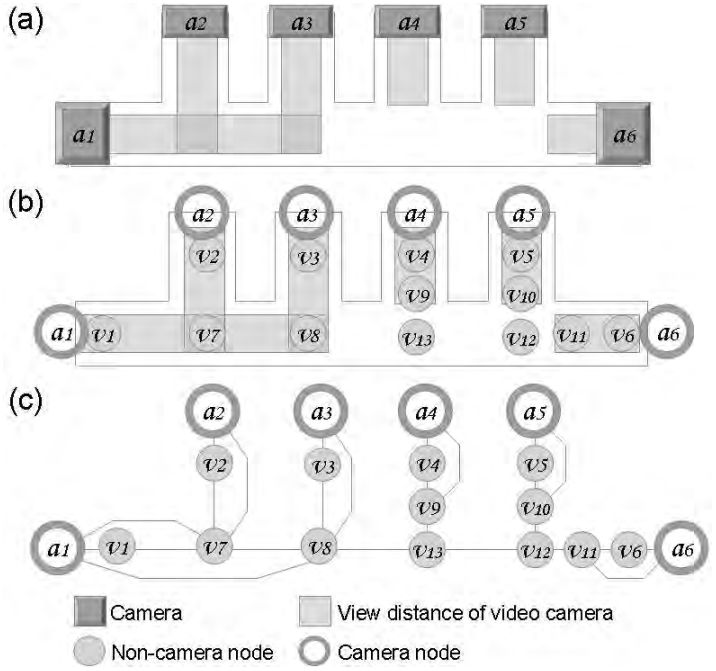

Fig. 4. Figure that sets non-camera nodes.

x_{ij} of matrix X is defined as (1). 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). The neighbor information for video cameras is calculated from the connection information of non-camera nodes by using adjacency matrix X and Y.

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

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

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

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 Fig.4. ii) Transform the diagram to a graph as shown in object (c) of Fig.4. 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 neighbor'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. Finally, matrix E which indicates the neighbor nodes is derived as (4).

$$E = X'Y'X'^T \begin{cases} \geq 1 & a_i \text{ is neighbour node to } a_j. \\ = 0 & a_i \text{ is not neighbour node to } a_j. \end{cases} \tag{4}$$

4. Human tracking method

Human tracking method consists of Follower method and Detection method. Follower method is used for tracking a moving target. Detection method is used for detecting a target when an agent has lost the target. In the tracking method, an agent has three statuses as "Catching", "Not catching" and "Lost". At first, an agent is assumed that it stays on a certain camera node. If the feature parameter the agent keeps is similar to the feature parameter extracted on the node, agent's status is indicated as "Catching". If the parameter the agent keeps is not similar to the feature parameter extracted on the node, agent's status is indicated as "Not catching". If the agent keeps "Not catching" status on a certain time, the agent decides that it lost a target, and agent's status is indicated as "Lost".

4.1 Follower method

In Follower method, an agent deploys its copies to neighbor nodes when agent's status becomes "Catching". When one of the copies has "Catching" status, all agents except that copy are removed from the system. And that copy becomes original agent. After that, the agent deploys its copies to neighbor nodes again. The follower method realizes tracking by repeating those routine.

4.2 Detection method

The detection method in this chapter is used to re-detect a target when the automatic tracking system loses the target. 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 chapter, 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 Fig. 5.

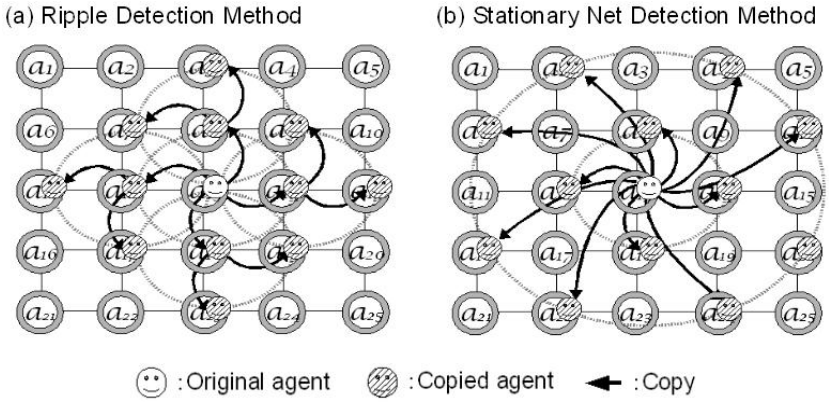

Fig. 5. Figure that sets non-camera nodes.

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 experimented 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, except 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 equation (10) in the algorithm.

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

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 experimented in search property. In the Stationary net detection method, the neighbor camera nodes are shown as (11).

$$E1 = E = X'Y'X' \tag{11}$$

When a mobile agent lost a target, copy agents are deployed to the next nodes of (11) expressed by (12), and search is started. $X'Y'^2X'^T$ shows neighbor camera nodes via two non-camera nodes, because the elements of $X'Y'^2X'^T$ 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.

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

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

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

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

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.

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

5. Experimentation

Here are two types of experiment. One is an experiment by simulator, and the other one is an experiment by real environment. In the experiment by simulator, follower method and

detection methods are experimented, and the effectiveness is verified. In the experiment by real environment, the tracking method is verified for whether the plural targets can be tracked continuously.

5.1 Experiment by simulator

Examination environment for the Ripple detection method and the Stationary net detection method is shown in Fig. 6 and Fig. 7. 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. 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 1, Table 2 and Table 3. The results of the measurement on the floor map 2 are shown in Table 4, Table 5 and Table 6. 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 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

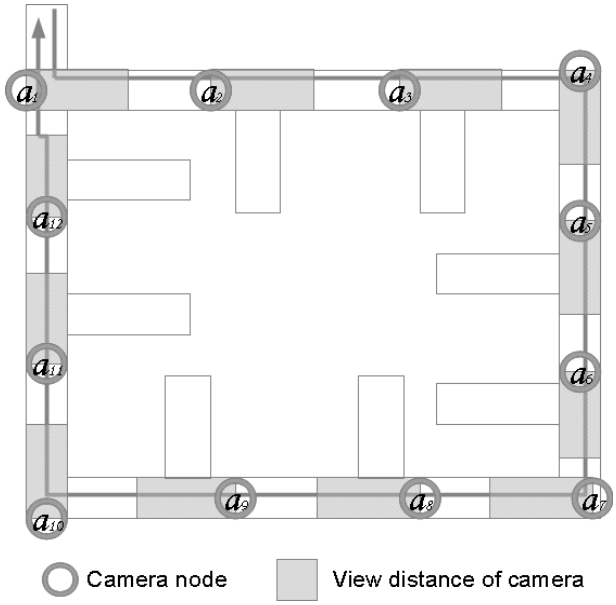

Fig. 6. Floor map 1 for experiment of detection methods.

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.

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

Table 1. Detection time on floor map 1 by walking speed 1.5*m/s*.

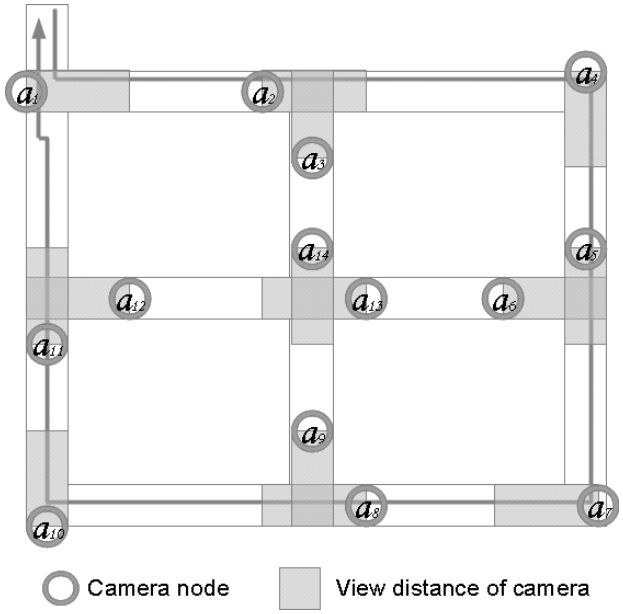

Fig. 7. Floor map 2 for experiment of detection methods.

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

Table 2. Detection time on floor map 1 by walking speed 2*m/s*.

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

Table 3. Detection time on floor map 1 by walking speed 3*m/s*.

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

Table 4. Detection time on floor map 2 by walking speed 1.5*m/s*.

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

Table 5. Detection time on floor map 2 by walking speed 2*m/s*.

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

Table 6. Detection time on floor map 2 by walking speed 3*m/s*

5.2 Experiment by real environment

Upon verification of real system, it aimed to confirm whether targets can be tracked continuously by tracking method. Therefore, in order to reduce influence by image processing performance, the image processing adopts simple processing by color vision. In this experiment, there is a miniature environment of 2*m* by 2*m* shown in Fig. 8 and Fig. 9. Here are three targets and those targets have each moving routes, red, blue, and green, shown in Fig. 8. The environment consists of seven servers with USB camera. The toys are used as targets instead of people tracked. This toy is a train toy with a sensor that recognizes a black line. And the train runs on along the line. The black line is used as a route a target walks and is drawn by hand. The toy is covered with a color paper to keep a certain accuracy of image processing as shown in Fig. 9. The moving speed of target is 6.8*m/second*. This environment has simulated a floor of 60*m* by 60*m* on the scale of 1/30; therefore the

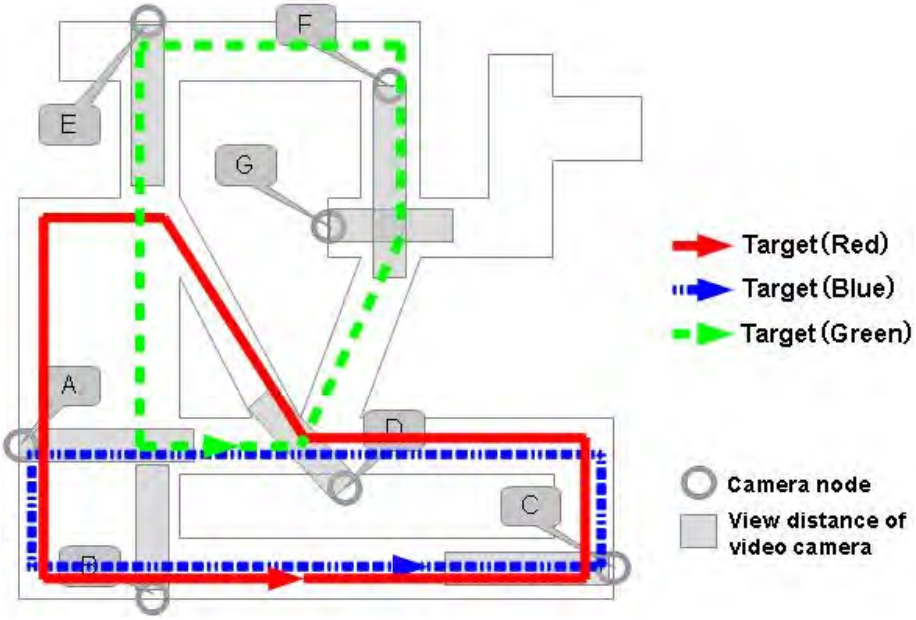

Fig. 8. Moving route of targets on experiment environment.

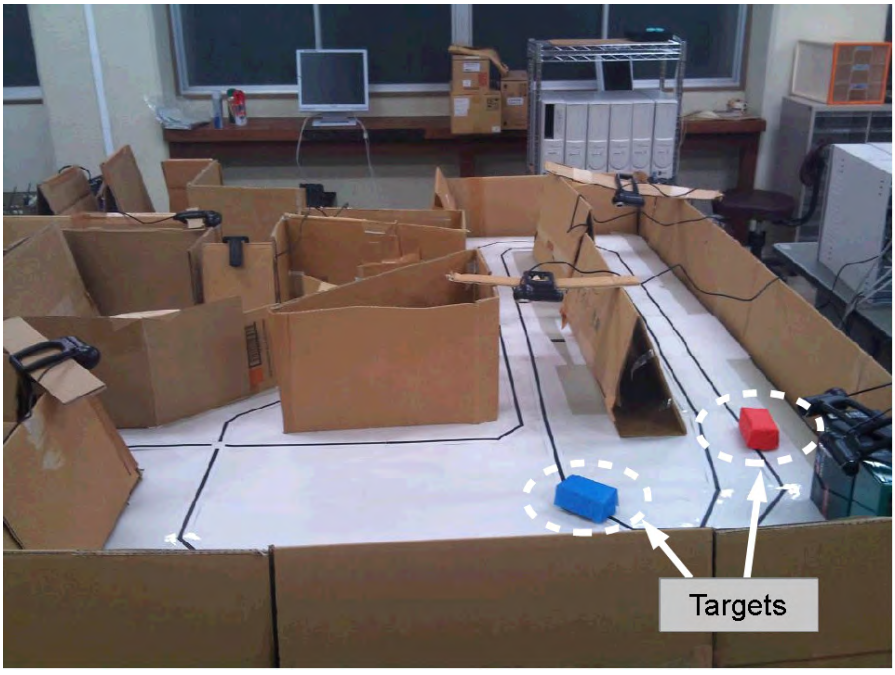

Fig. 9. Real experiment environment.

moving speed of the target is equivalent to 2.0*m/second*. Consider the moving route and the moving speed of the target, the search cycle is set to 5 seconds.

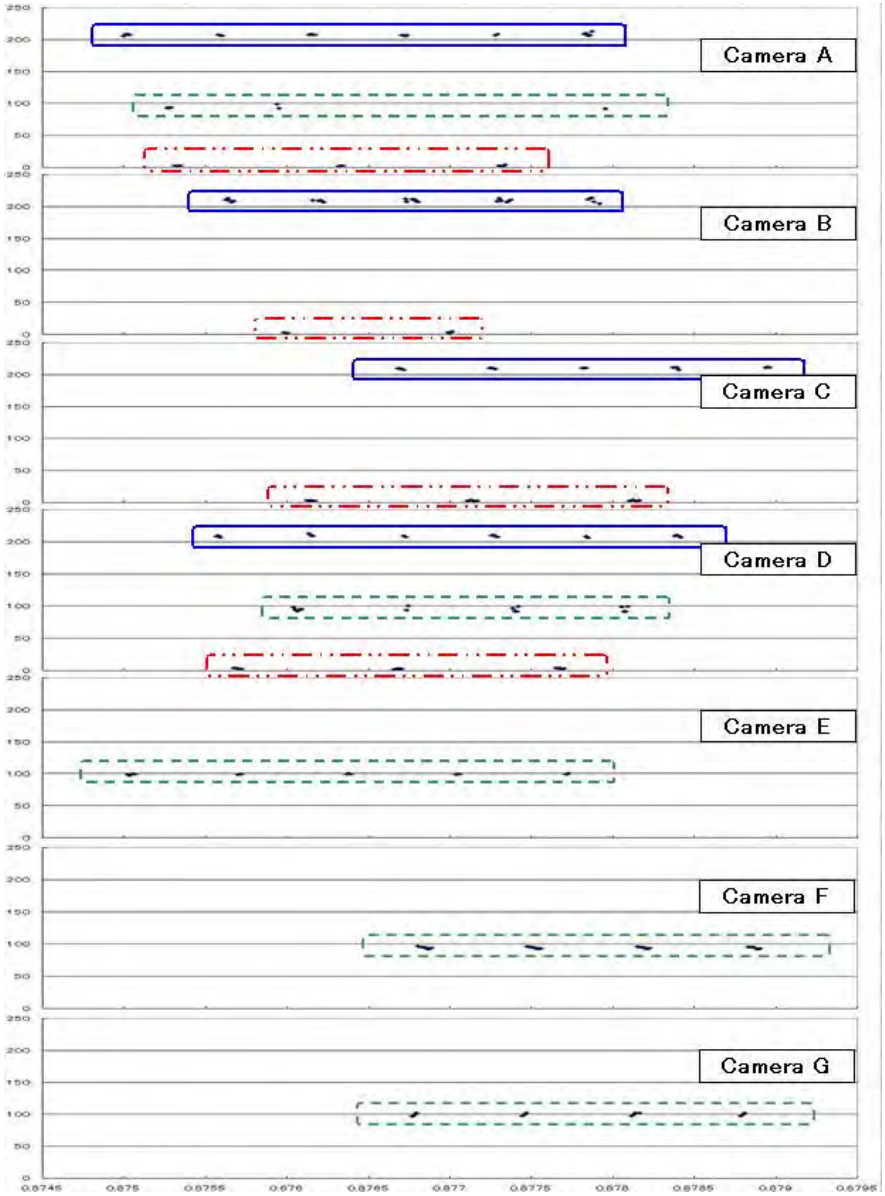

Fig. 10. Detection of target by camera on experiment environment.

Fig. 10 shows the result in the experiment. The graph of Fig. 10 is arranged in order from camera A to camera G. Horizontal axis is time and the vertical axis is the value of the

extracted H-HVS color. A solid line rectangle encloses the part where a red target was discovered, the rectangle of the short dashed line encloses the part where a blue target was discovered, and a dotted line rectangle encloses the part where a green target was discovered. The plot becomes a constant pattern if target is captured correctly, because the movement of the target is periodic. However, The illuminance was actually different according to the place that the camera installed, the color was judged the black by the shadow, and the target might not be able to be caught. Therefore, the number of agents did not increase and decrease by a constant pattern. But, the Neighbor node determination algorithm was able to calculate neighbor camera nodes. However, when the target is captured by video camera, the neighbor nodes are recomputed by the algorithm, then it was confirmed that an agent might not be deployed appropriately for a few dozens milliseconds. About this result, it is necessary to improve the computation time. In graphs of Fig. 10, the plot shows the time the target captured. It is confirmed that a target is lost temporarily from the camera capturing the target. But it is confirmed that the human tracking keeps tracking targets continuously by capturing the target by other cameras.

6. Conclusion

A construction method of automatic human tracking system with mobile agent technology is proposed using neighbor node determination algorithm. The construction method consists of the neighbor node determination algorithm and tracking methods. The neighbor node determination algorithm can compute neighbor nodes. This algorithm can be efficient to compute neighbor node and can make the system robust even if view distance of camera is changed. The tracking methods consist of follower method and detection method. The follower method can identify feature of a target locally. The detection method can search a lost target but a search cycle has to be within *walking speed* × *distance between cameras*. The detection method 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 lost the target. In addition, from the experiment 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.

The effectiveness of proposed tracking method was experimented using simulator and real environment. In the experiment using simulator, the tracking methods are experimented by the walking speed of a target. In the detection methods, consideration is added about the propriety of the parameter n which gives the number of non-camera nodes. Aimed to confirming behavior of the automatic human tracking system, the system in a real environment uses the simple image processing which can identify the color information of a target except the influence by the accuracy of image processing. And the follower method and the detection method are confirmed to be effective by a toy instead of a targeted person and to be able to construct the automatic human tracking system.

We will research more efficient detection to improve the automatic human tracking system. In addition, the accuracy of image processing has to be improved more to track a target more accurately. We are considering to improve our tracking system by combining effective

studies and to improve image processing program by using PCA-SIFT (Ke et al. 2004) or SURF (Bay et al. 2008) algorithm.

7. Acknowledgment

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

8. References

Terashita, K.; Ukita, N. & Kidode, M. (2009). Efficiency improvement of probabilistic-topological calibration of widely distributed active cameras, *IPSJ SIG Technical Report*, vol. 2009–CVIM–166, pp. 241–248

Kawaguchi, Y.; Shimada, A.; Arita, D. & Taniguchi, R. (2008). Object trajectory acquisition with an active camera for wide area scene surveillance, *IPSJ SIG Technical Report*, vol. 2008–CVIM–163, pp. 1306–1311

Yin, H. & Hussain, I. (2008). Independent component analysis and nongaussianity for blind image deconvolution and deblurring, *Integrated Computer-Aided Engineering*, vol. 15, No. 3, pp. 219–228

Tanizawa, Y.; Satoh, I. & Anzai, Y. (2002). A User Tracking Mobile Agent Framework "FollowingSpace", *Information Processing Society of Japan*, Vol. 43, No. 12, pp. 3775-3784

Tanaka, T.; Ogawa, T.; Numata, S.; Itao, T.; Tsukamoto, S. & Nishio, S. (2004). Design and Implementation of a Human Tracking System Using Mobile Agents in Camera and Sensor Networks, *IPSJ Workshop of Groupware and Network services 2004*, pp. 15-20

Nakazawa, A.; Hiura, S.; Kato, H. & Inokuchi, S. (2001). Tracking Multiple Persons Using Distributed Vision Systems, *Information Processing Society of Japan*, Vol. 42, No. 11, pp. 2699-2710

C. R. Wren, U. M. Erdem, and A. J. Azarbayejani, Automatic pan-tilt-zoom calibration in the presence of hybrid sensor networks, *Proceedings of the Third ACM International Workshop on Video Surveillance & Sensor Networks (VSSN'05)*, pp. 113–120, Nov. 2005.

N. Takemura & J. Miura, View planning of multiple active cameras for wide area surveillance, *Proc. IEEE International Conference on Robotics and Automation (ICRA'07)*, pp. 3173–3179, Apr. 2007.

Aswin C. Sankaranarayanan, Ashok Veeraraghavan & Rama Chellappa, Object Detection, Tracking and Recognition for Multiple Smart Cameras, *Proceedings of the IEEE*, vol. 96(10), pp. 1606-1624, Oct 2008

Sommerlade, E. & Reid, I. , Probabilistic Surveillance with Multiple Active Cameras, *Robotics and Automation (ICRA), 2010 IEEE International Conference on* , vol., no., pp.440-445, 3-7 May 2010

Alexandros Iosifidis & Spyridon G. Mouroutsos, A Hybrid Static/Active Video Surveillance System, *Antonios Gasteratos International Journal of Optomechatronics*, 1559-9620, Volume 5, Issue 1, Pages 80 – 95, January 2011

F. Porikli and A. Divakaran, Multi-Camera Calibration, Object Tracking and Query Generation, *Proc. International Conference on Multimedia and Expo (ICME'03)*, pp. 653–656, July 2003.

Lange, D. B. & Oshima, M. (1999). Seven good reasons for mobile agents, *Communications of the ACM*, vol. 42, no. 3, pp. 88–89

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

Motomura, S.; Kawamura, T. & Sugahara, K. (2005). Maglog: A mobile agent framework for distributed models, *Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems*, pp. 414–420

Kawamura, T.; Motomura, S. & Sugahara, K. (2005). Implementation of a logic-based multi agent framework on java environment, *Proceedings of IEEE International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, pp. 486–491

Cabri, G.; Leonardi, L. & Zambonelli, F. (2000). Mobile-agent coordination models for internet applications," *Computer*, vol. 33, no. 2, pp. 82–89, February 2000

Valetto, G.; Kaiser, G. & Kc, G. S. (2001). A mobile agent approach to process-based dynamic adaptation of complex software systems, *Lecture Notes in Computer Science*, vol. 2077, pp. 102–116

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

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

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

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