コンピュータビジョンと
イメージメディア 131－6
（2002．1．17）

分散視覚システムによる複数対象同時追跡

中澤篤志＊　加藤博一＊＊　日浦慎作＊＊＊　井口征士＊＊
＊)科学技術振興事業団（東京大学生産技術研究所）
＊＊)広島市立大学情報科学部 ＊＊＊)大阪大学基礎工学研究科

概要　本論文では分散視覚システムにより広域環境内で複数対象を同時観測する手法について述べる．本システムはネットワーク結合された複数台の観測ステーションによって構成され，各々が自律的に画像処理し観測を行う．これを構成するための具体的手法として，1）リアルタイム人物トラッキング・位置検出手法，2）観測ステーション間のタスク決定アルゴリズム，3）ステーション間の対象対応付けアルゴリズム，の3点を取り上げ詳説する．本手法を検証するため，実環境で3台のステーションを用い3人物を対象とした実験，および6台のステーションで6人物を対象としたシミュレーション実験を行った．その結果，本手法の特徴やその特性を明らかにすることができた．

Tracking multiple persons using distributed vision systems

Atsushi Nakazawa* Hirokazu Kato*** Shinsaku Hiura** Seiji Inokuchi**
*Institute of Industrial Science, The University of Tokyo.
**Graduate School of Engineering Science, Osaka University.
***Faculty of Information Sciences, Hiroshima City University.

Abstract: *This paper shows the multiple targets watching method in wide-area spatial environment using the Distributed Vision Systems (DVS). DVS is constructed with some "watching stations" that consist of a camera and an image processor and a computer network connects them. This system's goal is to track multiple persons in a wide-area that cannot be watched by one visual sensor. We introduce three algorithms to realize this system, realtime human tracking method, the task decision algorithms of watching stations and the object-matching method between stations. Finally, we describe experimental results that show the validity of our approach.*

1. Introduction

Wide-area surveillance systems are very necessary in security applications, general human-computer-interaction, teleconferencing applications and Intelligent Transport Systems(ITS). However, precise information cannot often be found from a single camera/sensor because of occlusions, image resolution and noise. The solution for these problems is using multiple camera/sensors that have different viewports. Maeda[1] proposed the "image fusion systems" that employed multiple cameras and only one image processor. But it didn't run in real-time because the processor had to process real-time image sequences gathered from multiple cameras. Recently, Distributed Vision Systems (DVS) have been proposed [2][3] for wide-area and real-time observation. These systems consist of a camera and an image processor (called the "watching station"), all connected through a computer network. Since the images acquired from cameras are processed on local watching stations, real-time and wide-area scene understanding can be realized. Furthermore, if multiple stations watch one target in the same time, more precise and accurate result can be found by coordinating measurement results.

Our system's goal is tracking multiple persons walking in open indoor environment and measuring their trajectories by using a DVS. In this paper, three methods are described. The first is real-time human tracking that runs on a watching station. This method not only tracks people but also finds their positions in world-coordinates. The second is the task decision algorithm of each watching stations. To track people continuously between multiple stations, watching stations have to decide their own tasks, such as tracking, acquisition of a person or idling. Since there is some variety in relations between the stations' viewing

areas (overlapped, partly overlapped or completely separated), the task decision algorithm depends on their geometrical relations. The third algorithm is for matching multiple targets between watching stations. It often occurs that multiple watching stations track multiple targets. Without a method for object-matching, stations cannot integrate measurement results for the same target that are obtained at different stations. We address this issue by using an agent-based approach. Each agent is an observation program (process) that can track only one person. If multiple people walk in one watching station's viewing area, multiple "agents" work on the station and observe each person. Grouping the result of these agents solves the problem identification of person between stations.

2. The system's architecture overview

Fig.1 shows the DVS architecture. The DVS consists of the watching stations and the station parameter management agents (SPMA) that are connected through a network. The network connection structure is flat and not hierarchical like a client-server model, so this system is robust to network or station failure. Each watching stations has a fixed camera and ability to run multiple visual processing programs, the seeing agents. A seeing agent can acquire images from the camera and track a person, detecting their position and communicate with local/ remote agents. Other kinds of agent also work for mediating between the seeing agents. The SPMA gathers and distributes all the watching stations' system parameters (the calibration parameter and camera position of the stations), but this agent is not used for controlling the watching stations.

3. Tracking persons

In this section, the human tracking algorithms which run on the seeing agents is explained. Although there are many studies about tracking people[4][5], we designed an approach optimized for use with a DVS. Our approach needed the following features: Realtime performance, the ability to track multiple people, and a measurement result given in world-coordinates. The last requirement is for integrating results. If the result is given in screen (image) coordinates, it is of no use when it sent to other stations.

In general, finding the initial tracking position for tracking is difficult. For this problem, our tracking method uses two tasks: the acquisition task and the tracking task. The acquisition task is used for the situation that the watching station can anticipate the region where the person will appear. Our method is based on comparing the current image to many simulated images. A person is modeled by a simple 3D-ellipse and projected onto the simulated images where the person's movement is assumed. The tracking task is used when a person has been found and can be tracked over subsequent video frames.

3.1 The acquisition task

The initial position of the human body is necessary for the

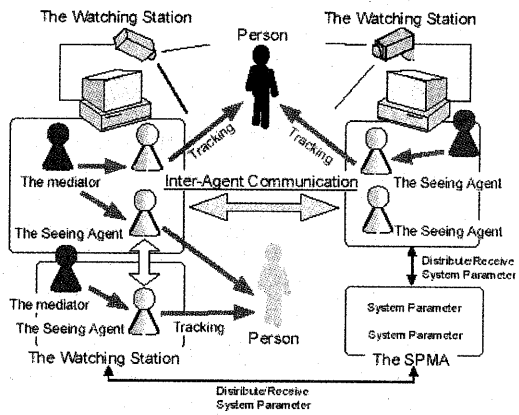


Fig.1 The DVS Architecture.

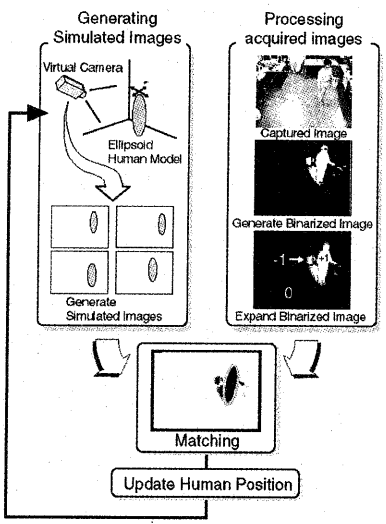


Fig.2 The tracking task flowchart.

tracking process. In most cases, a station can use messages from other watching stations to anticipate when and where a person will appear in its field of view. The station does the acquisition task by using many simulated images in its viewport. The station assumes the person will appear at one of several positions in its viewing region. If any of the simulated images match portions of the actual video frame by more than a threshold factor, it is used as an initial person position and the tracking task started.

3.2 Tracking task

The tracking task consists of four steps, as shown in Fig.2.

Step 1: A binalized image is generated by comparison the acquired image to the background image.

Step 2: To track human movement, many simulated images

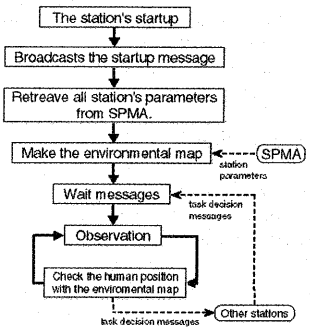


Fig.3 The task decision algorithm overview.

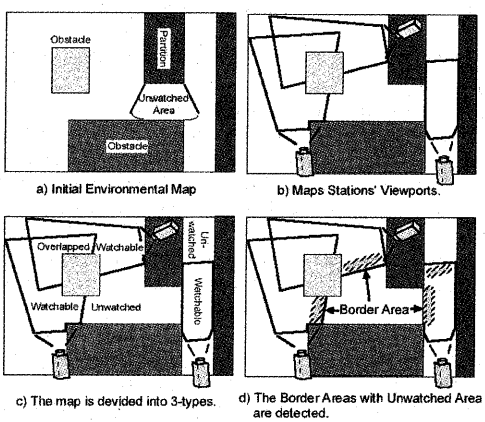


Fig.4 The environmental map generating algorithm.

are created at positions where the person might move to from the current position. A person is modeled as a 3D-ellipse and it is projected onto simulated images by using previously known camera parameters. We assume that a person will not move more than 50mm in two directions in world coordinates, so eight simulated images are created plus an additional simulated image at the current position.

Step 3: The correlation between each simulated image and the current binalized image is evaluated.

Step 4: The person's position is updated to the position with the highest similarity value. The next tracking process is now begun with this new position.

This process is iterated three times for an acquired image. For tracking multiple people, multiple ellipsoid models are projected onto the simulated images. We are able to carry out this process at 10-15 frames/sec, so the system can track people moving up to 1.5 - 2.25 m/sec.

4. The task decision algorithm of watching stations

We use multiple watching stations for tracking people

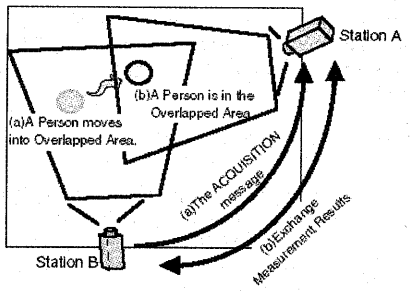


Fig.5 The task decision algorithms and the task dicision messages.
(a): A person moves into the Overlapped Area.
(b): Multiple stations are watching a person.

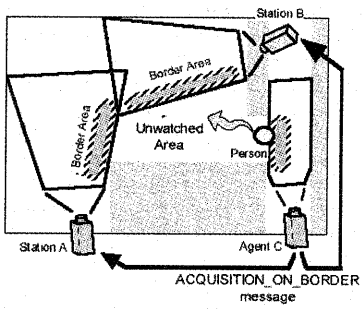


Fig.6 The task decision algorithms and the task decision messages. A person is going into the Unwatched Area.

continuously over a wide environment. Each station watches a part of the whole environment, so the stations' watching areas can be overlapped or separated. We don't use a single central-control station that would make our approach susceptible to a station or network falure. In our algorithm, each watching station decides its own current task according to the messages (not the "commands") received from the neighboring stations. Fig.3 shows an overview of the task decision algorithm.

4.1 Making the environmental map

When a watching station starts, it makes an "the environmental map" according to its own and the other stations' camera positions and calibration parameters. A world environmental map is prepared for every watching station (fig.4a). In this map, the position and attribute of the objects in the environment are described. After the station starts up, it broadcasts a "startup message" and receives all the other stations' camera positions and calibration parameters from the SPMA. Then the station maps the other station's watching areas and its own watching area onto the world environmental map (fig.4b). Then the environmental map is divided into three types of areas (fig.4c): regions that only one station can watch (the watchable area), regions that multiple stations can watch (the overlapped area) and those that no station can watch (the unwatched area). Using

this final map, each station can determine other stations whose watching regions are overlapping or neighboring, bordering their own watching area and the neighboring unwatched areas (fig.4d).

4.2 The task decision messages

After the environmental map is made, each station starts to wait for the task decision messages. There are 3 kinds of the messages, the ACQUISITION message, the ACQUISITION_ON_BORDER message and the STOP_ACQUISITION message. If a station receives the ACQUISITION message, it starts the acquisition task at the position described in the acquisition message packet (fig.5(a)-Station A). If a station receives the ACQUISITION ON_BORDER message, it starts the acquisition task on the border area which is described in the message packet (fig.6-Station A,B). A stations stops the acquisition task when it receives the STOP_ACQUISITION message.

Inversely, a station sends messages when it is tracking a person or loses track of a person. If the station is tracking a person whose position is in an overlapped area, it sends the ACQUISITION message to those stations whose watching areas are overlapping it's own (fig.5(a)-Station B). If it fails to track and no other stations are watching, the station judges that the person has moved into an unwatched area. Then it determines the area (the nearest from the position) and sends an ACQUISITION_ON_BORDER message to the neighboring stations (fig.6-Station C). If any stations then find a person appearing from the unwatched area, it sends a STOP_ACQUISITION message to the stations that are doing the acquisition task.

If multiple stations track the same target, the stations exchange measurement results (fig.5(b)). Then they can estimate a more accurate position by using all the results. If one station fails to track and others are successfully tracking, the station tries to resume tracking at the position that is received from other stations.

4.3 Detecting stations' startup/failure

In our distributed system, it cannot be supposed that all the stations will be working all time. So the systems have to detect a station's startup or failure. The startup detection is realized by receiving the startup message. A failure can be detected when exchanging the task decision messages. If a station receives a task decision message, it replies with an acknowledgement. If the sender does not receive this acknowledgement, it supposes the receiver has failed it and broadcasts a failure announcement. When stations receive a startup or failure announcement, they reconstruct their environmental map.

4.4 The station parameter management agent (SPMA)

The SPMA is used for gathering and distributing all the station parameters. Receiving a startup message, SPMA replies by sending its network address to the sender, then the sender can retrieve all the station parameters from the SPMA. The SPMA also updates all the station parameters when it receives startup and failure announcements.

4.5 The advantages

The features of this algorithm are:

1. No central control station is used,
2. The task allocation messages are transmitted between geometrically local stations.
3. Station's startup or failure is allowed while the systems is running.

Due to these features, our algorithm is robust against station or network failure. Furthermore using effective local communications reduces the network bandwidth.

5. The expansion for tracking multiple persons

In section 4, our wide-area tracking method is described. We now describe an expansion of this method for tracking multiple people. Nishio[6] presented a system for tracking multiple cars passing trough a intersection using multiple image-processing stations. In this system, an observation program(agent) works on one of the station. Each agent corresponds a car one-to-one and achieves observation by sending commands to many image-processing stations. This system works well in simple noise-free environment. Because an agent has an target's observation result that is acquired from one image-processing station's observation result . So if the result includes error, agent's observation is affected by it. We are addressing for much complicated indoor environment: the watching areas are overlapping or separated, occlusion, noises and etc. Furthermore, we don't use central-control station for the purpose of the system's robustness.

Ukita[2] proposed a multiple target tracking system where watching stations have a pan-tilt-zoom camera, so each station can observe a much wider area than ours. Their system's goal is to observe a specific target precisely by using zoom function. Thus, it may occur to exist unwatched targets in the watching area.

Our system's goal is to track all people in the watching area. We achieve this by expanding wide-area watching method described in section 4. In the following sections, two methods are shown: a new software architecture for tracking multiple people on a single stations and the object-matching method used between stations.

5.1 The expansion of systems software architecture: The agent approach

We employ new software architecture for multiple target tracking. In this architecture multiple observation programs (the seeing agent) works in parallel on each station (fig.1). Although a seeing agent can track only one person, multiple target tracking is possible if multiple agents are used. So we can easily expand

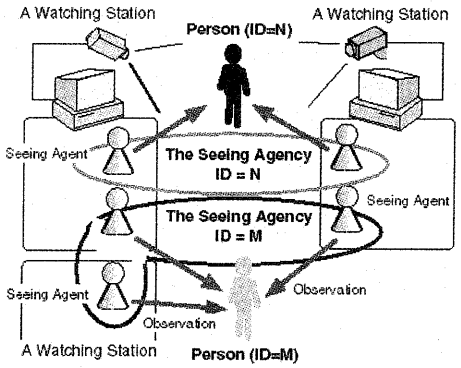


Fig.7 The object-matching between stations is realized by making agents' group("the agency").

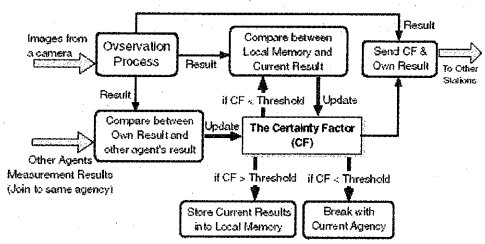


Fig.8 The seeing agent's architecture.

our system to track multiple targets and ideally our system can track a larger number of people than the number of watching stations.

5.2 Matching persons between stations

To achieve robust tracking of multiple targets, a new object-matching algorithm between stations is necessary. This is because object mismatching often occurs if multiple targets come close on the acquired images (fig.13(b)). This mismatching produces error in object position estimation, and problem in resuming of a station's tracking. We address this issue by grouping seeing agents with similar measurement results into an agent group ("the agency"). This idea is illustrated in figure 7. An agency has an ID unique to the systems. There is a one-to-one correspondence between agency groups and individual tracked people, so the tracked people have the same ID. When mismatching occurs, the agency is broken up and a new agent group is generated.

The accuracy of our grouping algorithm result is found from a certainty factor that every seeing agent maintains. The architecture of the seeing agent is shown in fig. 8. An agent acquires the target's position and its color while executing the tracking task. If an agent joins an existing agency, it compares its own measurement result with other agent's in the same agency. If their difference is smaller than a threshold, the certainty factor increases. On the other hand, if their difference is larger than the threshold, the certainty factor decreases. The certainty factor is kept by every agent and indicates the certainty that the agent's target is the person with a given ID. Consequently, if the target might be mismatched, this value is small. The certainty factor is also affected by image processing events and the difference between the current color and past measurement results stored in the local agent. If the certainty factor is less than a threshold, the agent breaks with the agency and continues to observe the target independently.

An independently-working agent follows the instructions of "the mediator". The mediator always watches the agent's observation result and determines which agency the agent should join.

5.2.1 Update the certainty factor

The certainty factor c(t) is updated by following:

$$c_t = c_{t-1} + k_1 \cdot eval\_event + k_2 \cdot eval\_comparison + k_3 \cdot eval\_knowledge$$
$$k_1, k_2, k_3 : const.(> 0)$$

Each evaluation value is set to -1, 0 or 1 according to following rules.

eval_event: If any events occur while processing images (such as two persons coming close and their image regions overlapping), this value is set to -1. Otherwise, this value is set to 0.

eval_comparison: The difference between the agent's own measurement results (position and color) and other agent's measurement results is calculated, provided that the certainty factor of the other agent is more than the agent's own certainty factor. If the difference is larger than a threshold this value is set to -1, less than threshold this value is set to 1, otherwise this value is set to 0.

eval_knowledge: If the agent's own certainty factor is less than a threshold, the difference between the current color and the target's colors found in the past is calculated. If the result is smaller than the threshold this value is set to +1, otherwise to 0. If the agent's own certainty factor is more than the threshold, the current result is stored in the agent's memory.

5.2.2 Send to other stations

If an agent is in an agency, it sends its own certainty factor and current measurement result to all the other agents that are in the same agency. This is done by using the ACQUISITION message (described in section 4). If multiple agents watch a person, they exchange their measurement results with the ACQUISITION message. We expand this protocol to contain sender's Agent ID, Agency ID and the certainty factor. Each agents has a unique agent ID, so if an agent receives a ACQUISITION message packet, it can identify the sender agent and its joining agency. Due to this exchanging process, the certainty factors of all agents can be updated.

5.2.3 The mediator

Each station has a mediator that makes agents join an existing agency or groups agents into a new agency. If there are any local agents that are working independently, the mediator compares between the agent's measurement result (position and color) and the all other agents' and agencies' measurement results that are on remote stations. The mediator has some "certainty factor lists" and the comparison results are accumulated in one of the them. This comparison rule is same as *the eval_comparison*. If any certainty factor in the list grows higher than the threshold, the mediator instructs the agent to join an existing or new agency. Each local seeing agents is only on one certainty factor list, so if multiple local agents are watching independently, the mediator uses different certainty factor lists.

6. Experiment

We tested our system in three ways. The first experiment was in a wide-area environment that included unwatched areas. One station watches a hallway while the other two watch the inside of an adjacent room with their watching regions overlapped. Fig.9 shows the experiment environment and trajectories that the two target persons walked.

The environmental map that stations generated is shown in fig.10. We can confirm they detected the unwatched areas and acquisition areas. Initially, station 3 and 4 do the acquisition task at the border of their watching areas. When persons 1 and 2 go into the watching areas of station 3 and 4, the stations find and track them. At first, person 1 goes into the unwatched area between the hallway and the room. Station 1 does the acquisition border task according the task decision message from station 4. In the same way, station 1 also finds the person 2. When the person 1 is in the overlapped area of station 1 and 2, station 2 does the acquisition task at the position sent from station 1, and they both track the person at the same time. The final detected trajectories are shown in fig.11. Stations sometimes lose people in the unwatched area, but they can resume tracking by using task allocation messages and acquisition tasks.

Experiment 2 was for testing our object-matching algorithm. Fig.12 shows the camera arrangement of this experiment. Three cameras were watching the room while two persons entered through the door and walked around. Fig.13 shows the tracking result from station 2. As shown in fig.13(b), the human image regions overlap many times. This event causes the object-mismatching between the stations. Initially, two seeing agents are generated on each station: Agent-A (in Station 1,2,3) is tracking person 1, while Agent-B (in Station 1,2,3) is tracking person 2. However at the point 5 (in fig.12), the two people come close and object-mismatching between the stations occurs. Fig.14 indicates this event: the detected trajectories from stations 2 and 3 are completely different at the point 5. Using this figure, we can confirm the mismatching has occured on station 3. The transition of the certainty factors (CF) and the agent grouping results are shown in fig.15,16. We notice that all the agent's CF

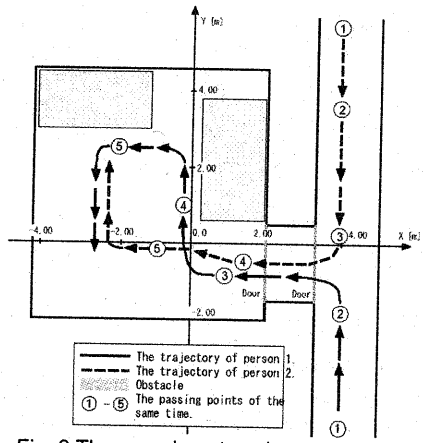


Fig. 9 The experiment environment and the trajectories of two persons walked.

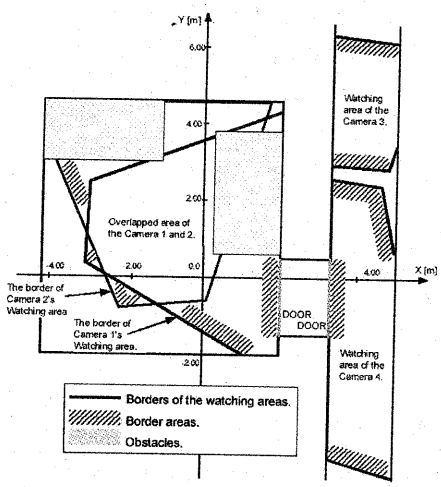


Fig. 10 The environmental map that stations generated.

decrease at the point 5. This is affected by the overlaps the human's image region, and we also find that both agent's CF at station 3 continue to decrease after this event, while those at station 2 increase. This phenomenon indicates that the agents at station 3 are tracking different people before and after the people have passed point 5 (Their targets have been swapped). Then, their certainty factor has decreased because their targets' colors and positions are completely different from agents that join the same agency.

At point 6, agents at station 3 had broken away from the agency, then the mediator made them join the correct agency. Finally, the correct matching (Station1-AgentA,Station2-AgentA, Station3-Agent B), (Statin1-AgentB,Station2-AgentB, Station3-AgentA) has been achieved. In above two experiments, the

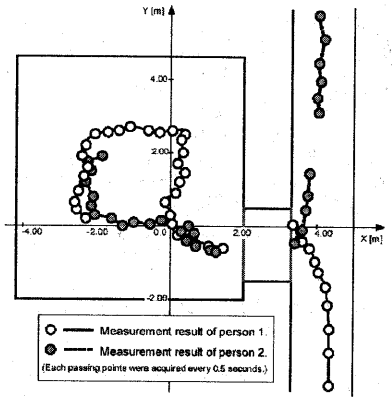


Fig. 11 The measurement result by 4 stations.

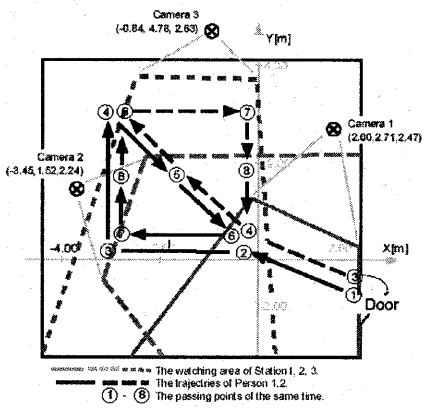
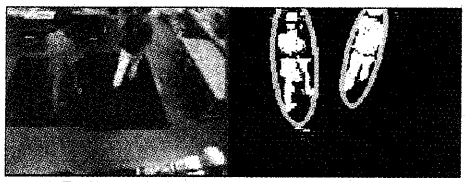


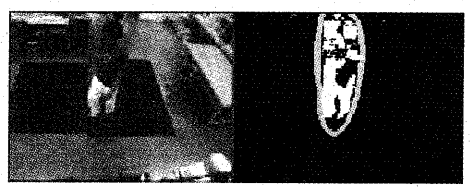
Fig. 12 The camera arrangement and trajectory of two persons (experiment 2).

stations consist of a PC (Pentium III-500MHz, Linux) with a CCD camera attached. They can acquire a 240x180 pixel video image and are connected to a 100Mbps network.

The last experiment is the computer simulation to evaluate our object-matching algorithm. In this experiment, 6 people are supposed to walk around where 6 stations are watching (Fig. 17). Two algorithms are tested, proposed algorithm and the conventional information-centerized algorithm. In the latter one, all information acquired by the stations (position and color, each one includes noise) is brought together in one place and integrated, then the integrated results are distributed for each stations again. The experiment results are shown in fig.18. We can notice that the trajectories of the person 2 and 4 are not obtained correctly in the information-centerized algorithm. This is because whether the systems has only a presumed result or many ones. In our algorithm, each agent have their measurement results independently, therefore a system has many presumed results for each tracking targets. We think this 'diversity' contributed to the tracking robustness.



(a)Tracking result when human image regions are separated.



(b)Tracking result when human image regions are overlapped.

Fig. 13 Tracking results on Station 2 in experiment 2.

7. Conclusion

We have described a distributed vision system that can watch multiple people over a wide-area environment. Three methods are used for this system; real-time tracking working on watching stations, a task decision algorithm used by multiple stations and an object-matching method used between stations. For the tracking algorithm, we developed a real-time 3D-model based tracking method. For the task decision algorithm, tasks are chosen according to the stations' geometric relationship and messages passed between the stations. For the object-matching method, we use a new software architecture in which many software agents works in parallel on each watching station. Grouping these agents solves the object matching between stations. Three experiments show that this system works well for tracking multiple people in the wide-area, and its robustness by the comparison of conventional algorithm and proposed one.

References

[1] T.Maeda, H.Kato and S.Inokuchi, "Image Fusion System for Object Tracking", Proc. of the Japan U.S.A. Symposium on Flexible Automation pp.365-368

[2] N.Ukita, T.Matsuyama, "Incremental Observable-Area Modeling for Cooperative Tracking", Proc. of 15th ICPR, Vol.4, pp.192-196

[3] A.Nakazawa, H.Kato, S.Inokuchi, "Human Tracking using Distributed Vision Systems", Proc. of 14th ICPR, pp.593-596

[4] Jakob Segen and Sarma Pingali, "A Camera-Based System for Tracking People in Real Time", Proc. of 13th ICPR, pp.63-67

[5] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, "Pfinder: Real-time tracking of the human body", IEEE Trans. on Pattern Analysis and Machine Intelligence Vol.19, No.7, pp.94-115

[6] Shuichi Nishio, Yuichi Ohta, "Tracking of Vehicles at an Intersection by Integration of Multiple Image Sensors", Proc. of IAPR Workshop on Machine Vision Applications (MVA'92), pp.321-324

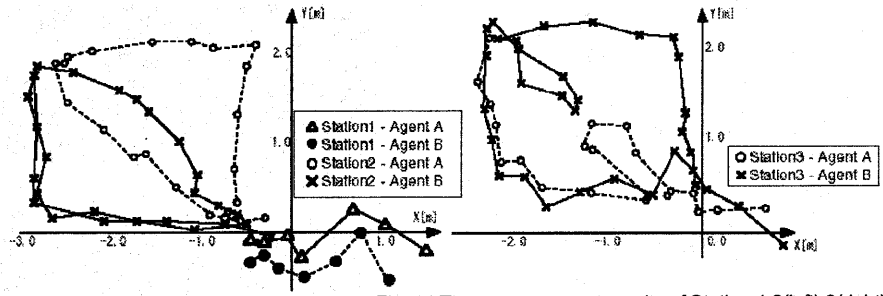


Fig. 14 The measurement results of Station 1,2(left),3(right) in exp.2.

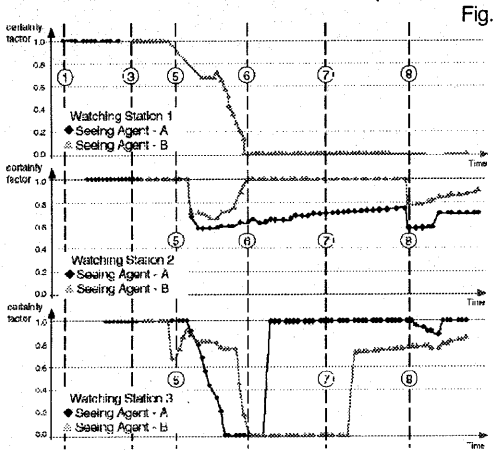


Fig. 15 The transition of the certainity factor in exp.2.
①- ⑧corresponds to the same time in fig.12.

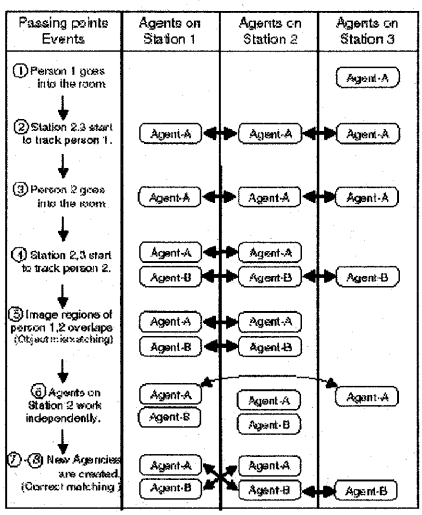


Fig.16 The grouping result of the agencies.
① - ⑧ corresponds the same time in fig.12,15.

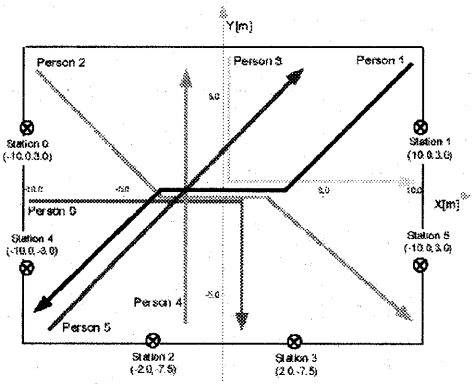
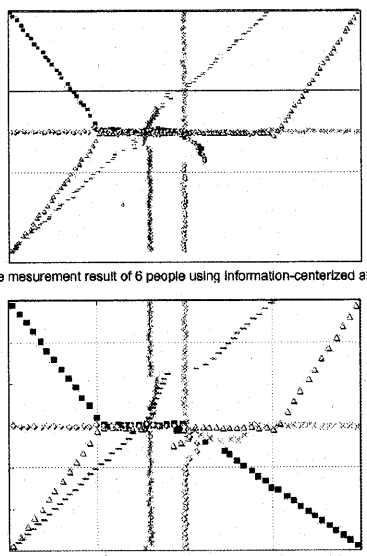


Fig. 17 The camera arrangement and trajectory of 6 stations/people (experiment 3).



a)The mesurement result of 6 people using information-centerized algorithm.

b)The mesurement result of 6 people using proposed algorithm.

Fig. 18 The simulation results of two object-matching algorithms (experiment 3)