Real-Time Multi-Target Tracking with Cooperative Communication among Active Vision Agents

Norimichi Ukita Takashi Matsuyama
Graduate School of Informatics, Kyoto University
souhaku@vision.kuee.kyoto-u.ac.jp tm@i.kyoto-u.ac.jp

Abstract

We present a real-time cooperative multi-target tracking system. The system consists of a group of Active Vision Agents (AVAs), where an AVA is a logical model of a network-connected computer with an active camera. All AVAs cooperatively track their target objects by dynamically exchanging object information with each other. In this paper, we address the technologies employed in the system and demonstrate their effectiveness.

1. Introduction

Object detection and tracking is one of the most important and fundamental technologies to develop real-world computer vision systems: e.g., visual surveillance systems, ITS (Intelligent Transport Systems) and so on.

We propose a real-time cooperative tracking system that gazes at multiple targets simultaneously. The system consists of communicating *Active Vision Agents* (AVAs, in short), where an AVA is a logical model of a network-connected computer with an active camera. For real-time target tracking by multiple AVAs, we have to solve many problems (e.g., how to design an active camera for dynamic object detection[1] and how to realize real-time object tracking with an active camera[2]). In this paper, we focus on how to realize a real-time cooperation among AVAs.

To implement the real-time cooperation among AVAs, we propose a three-layered interaction architecture. In each layer, parallel processes exchange different kinds of information for effective cooperation. To realize a real-time information exchange and processing, we employ the dynamic memory architecture proposed in [2]. The dynamic interaction in each layer allows the total system to track multiple moving targets under complicated dynamic situations in the real world.

2. Cooperative Multi-Target Tracking

2.1. Architecture of AVA and Its Functions

Each AVA consists of a network-connected computer with a *Fixed-Viewpoint Pan-Tilt-Zoom* (FV-PTZ) camera[1]: its projection center stays fixed irrespectively of any camera rotations and zoomings. With the FV-PTZ camera, an AVA can track a moving object as follows[2]:

1. Generate a wide panoramic image of the scene; with the FV-PTZ camera, a wide panoramic image can be easily generated by mosaicing multiple images observed by changing pan, tilt and zoom parameters.
2. Extract a window image from the panoramic image according to the current pan-tilt-zoom parameters and regard it as the background image; the direct mapping exists between the position in the panoramic image and pan-tilt-zoom parameters of the camera.
3. Compute difference between the generated background image and an observed image [1] .
4. If anomalous regions are detected in the difference image, select one and control the camera parameters to track the selected target.

In our system, an agent (i.e., AVA) corresponds to a single camera. Each AVA can, therefore, control its own camera to gaze at its target. Many other multi-camera tracking systems (see [3], for example), on the other hand, defined a software agent to correspond to each detected object. This definition forces each agent to examine the object information detected by all cameras for tracking its target. In addition, multiple agents may control a camera inconsistently in tracking the target, if the system employs active cameras. As we can see, our definition of the agent has the advantage in that it has the one-to-one correspondence between the agent and the camera.

2.2. Basic Scheme for Cooperative Tracking

Our tracking system consists of multiple AVAs. The system assumes that the cameras are calibrated and densely distributed over the scene so that their visual fields are well overlapping with each other.

Followings are the basic tasks of the system:

1. Initially, each AVA independently searches for targets.
2. If an AVA detects a target, it navigates the gazes of the other AVAs towards that target.
3. A group of AVAs which gaze at the same target form what we call an *Agency* and keep measuring the 3D information of the target from multi-view images.
4. Depending on target locations in the scene, each AVA dynamically changes its target.

To realize the above cooperative tracking, we have to solve the following problems:

Multi-target identification: To gaze at each target, the system has to distinguish multiple targets.

[1] To utilize this object detection method under varying geometric and photometric environments, a robust background subtraction (see [4], for example) is required.

Real-time and reactive processing: To adapt itself to dynamic changes in the scene, the system has to execute processing in real-time and quickly react to the changes.

Adaptive resource allocation: We have to implement two types of dynamic resource (i.e., AVA) allocation: (1) To perform both target search and tracking simultaneously, the system has to preserve AVAs that search for new targets even while tracking targets, (2) To track each moving target persistently, the system has to adaptively determine which AVAs should track which targets.

We solve these problems with real-time cooperative communication among AVAs and agencies.

3 Task Specification

We realize the flexible task specification with the following three parameters, namely the *Task-Constraint*, the *Object-Importance* and the *Utility-Function*.

3.1 Task-Constraint

An AVA that searches for targets is called a *Freelancer*-AVA. An AVA belonging to an agency for tracking its target is called a *Member*-AVA.

We realize various capabilities of the system, in terms of the combination of search and tracking as follows:

Def. 1 (Search-level, Tracking-level) *The search-level and tracking-level indicate the rate of AVAs that perform search and tracking, respectively.*

$$\begin{aligned} 0 \leq \quad & \text{Search-level} \ (= N_F/N_A) \quad \leq 1 \\ 0 \leq \quad & \text{Tracking-level} \ (= N_M/N_A) \quad \leq 1, \end{aligned}$$

where N_F, N_M and N_A denote the numbers of freelancer-AVAs, member-AVAs and all AVAs, respectively.

Def. 2 (Current state (S_P, T_P)) *This parameter represents the search-level (S_P) and tracking-level (T_P) at the present time. $(S_P + T_P)$ is always 1.*

Def. 3 (Task-constraint (S_C, T_C)) *This parameter represents the minimum search-level (S_C) and tracking-level (T_C), where $0 \leq (S_C + T_C) \leq 1$. The system has to keep S_C and T_C while working. This parameter is determined by a user depending on the task given to the system.*

Each AVA dynamically changes its own role between search and tracking to adapt the current state of the system to the task-constraint.

3.2 Object-Importance

The object-importance is given to each object's category that can be distinguished by the system.

Def. 4 (Object-importance I_P) *Let I_P ($0 \leq I_P \leq 1$) denote the object-importance of the target of agency$_P$. The number of the member-AVAs in agency$_P$ (denoted by M_P) is determined by the object-importance of the target: $M_P = (\text{The total number of AVAs}) \times (I_P/S)$. Provided that S is the total sum of the object-importance $I_{1,\cdots,A}$, where A is the total number of the agencies.*

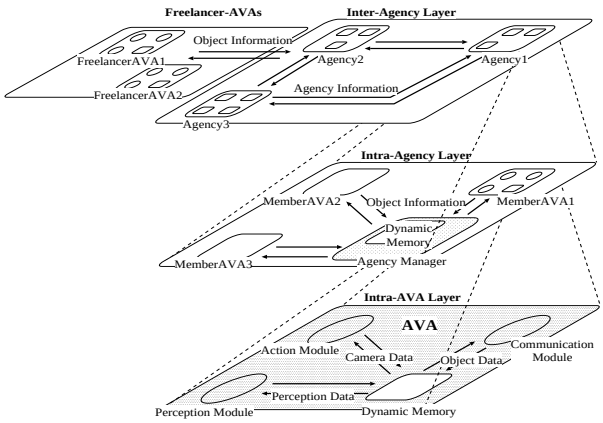


Figure 1. Three-layered dynamic interaction.

3.3 Utility-Function

Each AVA can freely change its role under the restrictions given by the task-constraint and object-importance. A guideline for the adaptive role assignment is represented by what we call the utility-function. Each AVA decides its role to increase the value of the utility-function while keeping the task-constraint and object-importance. The utility-function of our tracking system is the sum of the following search-value and tracking-value.

- **Search-value of a freelancer-AVA** is determined by a fitness of each freelancer-AVA for search.
- **Tracking-value of a member-AVA** is determined by a fitness of each member-AVA for tracking its target.

This utility-function can be designed to be adapt itself to the task given by a user (will be shown in Sec.5).

4. Three-layered Dynamic Interactions for Cooperative Tracking

In our system, parallel processes cooperatively work by dynamically interacting with each other. As a result, the system as a whole works as a tracking system. By composing the system as a group of multiple processes, we can represent the complex behavior of the total system through the interaction between processes. Designing the total system can be, therefore, reduced to designing each process. Furthermore, the states and those transitions of the system increase enormously by combining with each other. We believe that this property allows the system to cope with complicated situations in the real world in contrast to the centralized processing systems (see [5], for example).

For the system to engage in object tracking, object identification is significant. We, therefore, classify the system into three layers depending on the types of object information employed for identification. In each layer, object identification according to the type of exchanged information is established. Depending on whether or not the result of object identification is successful, a dynamic interaction protocol for cooperative object tracking is activated.

4.1. Intra-AVA layer

In the bottom layer in Fig.1, perception, action and communication modules that compose an AVA interact with each other via the dynamic memory[2] possessed by each AVA. An AVA is an augmented target tracking system proposed in [2], where the augmentation is threefold:

(1) Multi-target detection while single-target tracking

When the perception module detects N objects at $t+1$, it computes and records into the dynamic memory the 3D view lines toward the objects (i.e., $L^1(t+1), \cdots, L^N(t+1)$)[2] . Then, the module compares them with the 3D view line toward its currently tracking target at $t+1$, $\widehat{L}(t+1)$. Note that $\widehat{L}(t+1)$ can be read from the dynamic memory whatever temporal moment $t+1$ specifies. Suppose $L^x(t+1)$ is closest to $\widehat{L}(t+1)$, where $x \in \{1, \cdots, N\}$. Then, the module regards $L^x(t+1)$ as denoting the newest target view line and records it into the dynamic memory.

(2) Gaze control based on the 3D target position

The action module reads the 3D view line toward the target (i.e., $\widehat{L}(now)$) from the dynamic memory and controls the camera to gaze at the target. As will be described later, when an agency with multiple AVAs tracks the target, it measures the 3D position of the target (i.e., $\widehat{P}(t)$) and sends it to all member-AVAs. If such information is available, the action module controls the camera based on $\widehat{P}(now)$ in stead of $\widehat{L}(now)$.

(3) Incorporation of the communication module

Data exchanged by the communication module over the network can be classified into two types: detected object data (i.e., $\widehat{L}(t)$ and $\widehat{P}(t)$) and messages for various communication protocols which will be described later.

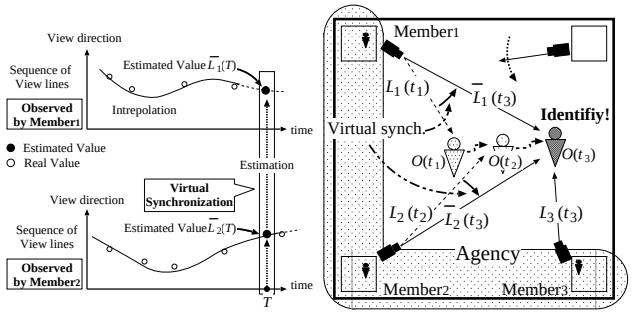
4.2. Intra-Agency layer

As defined before, a group of AVAs which track the same target form an agency. The agency formation means the generation of an *Agency Manager*, which is an independent parallel process to coordinate interactions among its member-AVAs. In our system, an agency should correspond one-to-one to a target. To make this correspondence dynamically established and persistently maintained, the following two kinds of object identification are required in the intra-agency layer (the middle layer in Fig.1).

(a) Spatial object identification

The agency manager has to establish object identification between the groups of the 3D view lines detected and transmitted by its member-AVAs. The agency manager checks distances between those 3D view lines detected by different member-AVAs and computes the 3D target position from a set of nearly intersecting 3D view lines. The manager employs what we call the *Virtual Synchronization* to virtually adjust observation timings of the 3D view lines (see 4.3 for details). Note that the manager may find none or multiple sets of such nearly intersecting 3D view lines. To cope with these situations, the manager conducts the following temporal object identification.

[2] The 3D view line determined by the projection center of the camera and an object region centroid in the observed image.



(a) (b)

Figure 2. Virtual synchronization for spatial object identification: (a) Read values from the dynamic memory, (b) Spatial identification.

(b) Temporal object identification

The manager records the 3D trajectory of its target, with which the 3D object position(s) computed by spatial object identification is compared. That is, when multiple 3D locations are obtained by spatial object identification, the manager selects the one closest to the target trajectory. When spatial object identification failed and no 3D object location was obtained, on the other hand, the manager selects such 3D view line that is closest to the target trajectory. Then the manager projects the target 3D position onto the selected view line to estimate the new 3D target position. Note that when an agency contains only a single AVA, neither spatial nor temporal object identifications succeed and hence the member-AVA just conducts appearance-based 2D tracking by itself.

4.3. Virtual Synchronization for Identification

Here we discuss dynamic aspects of the above identification processes.

(a) Spatial object identification

Since AVAs capture images autonomously, member-AVAs in an agency observe the target at different moments. Furthermore, the message transmission over the network introduces unpredictable delay between the observation timing by a member-AVA and the object identification timing by the agency manager. In [3, 6], the object information detected at t_i and t_j, where $|t_i - t_j|$ is small enough, is considered to be observed simultaneously. Such approximate methods, however, break down under complicated situations and network congestion. To solve this problem, we introduce the dynamic memory into an agency manager, which enables the manager to virtually synchronize any asynchronously observed/transmitted data. We call this function *Virtual Synchronization* by the dynamic memory.

Fig.2 shows the mechanism of the virtual synchronization. All 3D view lines computed by each member-AVA

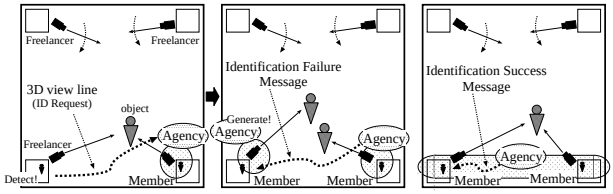


Figure 3. Agency Formation.

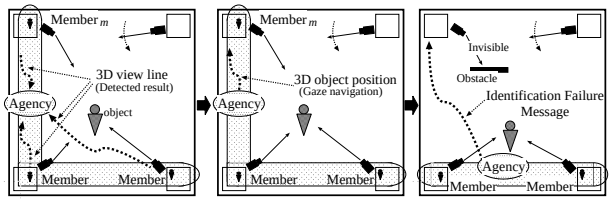


Figure 4. Agency Maintenance.

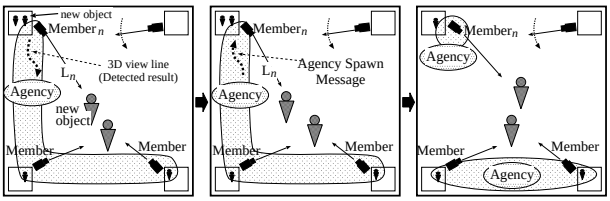


Figure 5. Agency Spawning.

are transmitted to the agency manager, which then records them into its internal dynamic memory. Fig.2 (a), for example, shows a pair of temporal sequences of 3D view line data transmitted from member-AVA$_1$ and member-AVA$_2$, respectively. When the manager wants to establish spatial object identification at T, it can read the pair of the synchronized 3D view line data at T from the dynamic memory (i.e., $\bar{L}_1(T)$ and $\bar{L}_2(T)$ in Fig.2 (a)). In the actual example shown in Fig.2 (b), the manager reads $\bar{L}_1(t_3)$ and $\bar{L}_2(t_3)$ from its dynamic memory to adjust observation timings.

(b) Temporal object identification

The virtual synchronization is also effective in temporal object identification. Let $\widehat{P}(t)$ denote the 3D target trajectory recorded in the dynamic memory and $\{P_i(T)|i = 1, \cdots, M\}$ the 3D positions of the objects identified at T. Then the manager (1) reads $\widehat{P}(T)$ (i.e., the estimated target position at T) from the dynamic memory, (2) selects the one among $\{P_i(T)|i = 1, \cdots, M\}$ closest to $\widehat{P}(T)$, and (3) records it into the dynamic memory as the target position.

4.4. Communications at Intra-Agency Layer

The following three communication protocols are activated depending on the success or failure of the above mentioned temporal object identification.

(a) Agency formation protocol

This protocol defines a new agency generation procedure by a freelancer-AVA and a participation procedure of a freelancer-AVA into an existing agency.

When a freelancer-AVA detects an object, it requests the existing agency managers to examine identification between the detected object and the target object of each agency (Fig.3, (1)). If no agency established object identification, the freelancer-AVA generates a new agency and joins into that agency (Fig.3, (2-a)). If an agency established object identification, the freelancer-AVA joins into the agency, if requested (Fig.3, (2-b)).

(b) Agency maintenance protocol

This protocol defines procedures for the cooperative tracking, the continuous maintenance of an agency and the elimination of an agency.

An agency manager repeats spatial and temporal object identifications for cooperative tracking (Fig.4 (1)). Following spatial object identification, the manager transmits the newest 3D target location to each member-AVA (Fig.4 (2)), which then is recorded into the dynamic memory of the member-AVA. If a member-AVA$_m$ cannot detect the target for a long time, the agency manager forces AVA$_m$ out of the agency to be a freelancer-AVA (Fig.4 (3)). If all member-AVAs cannot observe the target, the agency manager destroys the agency and makes all its member-AVAs become freelancer-AVAs.

(c) Agency spawning protocol

This protocol defines a new agency generation procedure from an existing agency.

After spatial and temporal object identifications, the agency manager may find such a 3D view line(s) that does not correspond to the target. This means the detection of a new object by its member-AVA. Let L_n denote such 3D view line detected by AVA$_n$ (Fig.5 (1)). Then, the manager broadcasts L_n to other agency managers to examine identification between L_n and their tracking targets. If none of the identification is successful, the manager makes AVA$_n$ quit from the current agency and generate a new agency (Fig.5 (2)). AVA$_n$ then joins into the new agency (Fig.5 (3)).

4.5. Inter-Agency layer

In multi-target tracking, the system should adaptively allocate resources: the system has to adaptively determine which AVAs should track which targets. To realize this adaptive resource allocation, the information about targets and member-AVAs is exchanged between agency managers in the top layer in Fig. 1.

The dynamic interactions between agency managers are triggered based on object identification of target 3D locations across agencies. Note that here also the virtual synchronization between a pair of 3D target locations is employed to increase the reliability of object identification. Depending on the result of this inter-agency object identification, either of the following two protocols are activated.

(a) Agency unification protocol

This protocol is activated when the inter-agency object identification is successful and defines a merging procedure of the agencies which happen to track the same object. This protocol is required to cope with failures of object identi-

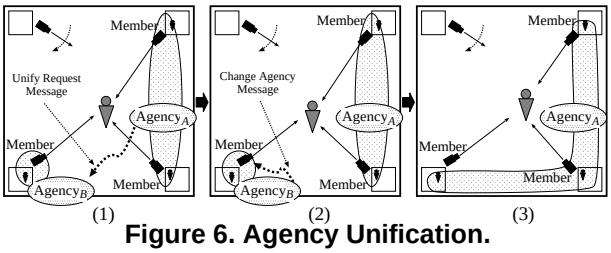


Figure 6. Agency Unification.

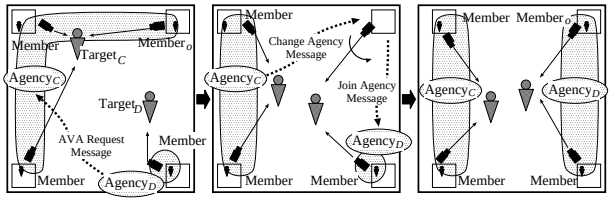


Figure 7. Agency Restructuring.

fication and discrimination: (1) asynchronous observations and/or errors in object detection by individual AVAs or (2) multiple targets which come too close to separate.

Fig.6 shows an example. When agency manager AM_A of agency_A establishes identification between its own target and the one tracked by AM_B, AM_A asks AM_B to be merged into AM_A (Fig.6(1)). Then, AM_B asks its member-AVAs to join into agency_A (Fig.6(2)). After copying the target information recorded in the dynamic memory into the object trajectory database, AM_B eliminates itself (Fig.6(3)).

(b) Agency restructuring protocol

When the inter-agency object identification fails, agency $\mathrm{manager}_j$ checks if it can activate the agency restructuring protocol taking into account the numbers of member-AVAs in agency_j and agency_i and their target locations.

In Fig.7, agency manager AM_C of agency_C sends its target information to AM_D, which fails in object identification. Then, AM_D asks AM_C to trade its member-AVA into AM_D (Fig.7(a)). When requested, AM_C selects its member-AVA and asks it to move to agency_D (Fig.7(b) (c)).

4.6. Communication with Freelancer-AVAs

An agency manager communicates with freelancer-AVAs as well as with other managers (in the top row of Fig. 1). As described in the agency formation protocol in Section 4.4, to determine whether or not generate a new agency, a freelancer-AVA activates the communication with agency managers when it detects an object. An agency manager, on the other hand, sends to freelancer-AVAs its target position when the new data are obtained. Then, each freelancer-AVA decides whether it continues to be a freelancer-AVA or joins into the agency depending on the target position and the current number of freelancer-AVAs in the system. Note that in our system a user can specify the number of freelancer-AVAs to be preserved while tracking targets.

4.7. Soundness of the Dynamic Interactions

In the proposed system, all events happened in the real world are characterized by the results of object identification. Therefore, by verifying the types of the protocols that are executed depending on the result of each object identification, we can confirm the necessity and sufficiency of the protocols for multi-target tracking.

All object identification is established when an agency received the object information from freelancer-AVAs, member-AVAs and other agencies. Table1 shows the types of the protocols that are activated according to the relations between the type of the received object information and the result of object identification. As we can see, the protocols are designed just enough in accordance with the situations in the real world.

5. Experiments

The system consists of ten AVAs. Each AVA is implemented on a network-connected PC (PentiumIII 600MHz $\times$ 2) with an active camera (SONY EVI-G20), where the perception, action, and communication modules as well as agency managers are realized as UNIX processes. Fig.9 (a) illustrates the camera layout in the room. Note that the internal clocks of all the PCs are synchronized by the Network Time Protocol to realize the virtual synchronization.

In the experiment, the system tracked two people. The task was specified as follows: tracking-level = 0.9, search-level = 0.1, object-importances of two targets were 1.0, the search-value of freelancer-AVA_f is proportional to the floor size that is visible from AVA_f, and the tracking-value of member-AVA_m is inversely proportional to the angle between the optical axis of the AVA_m's camera and the direction from the AVA_m's camera to the target.

The upper part of Fig. 8 shows the partial image sequences observed by AVA_2, AVA_5 and AVA_9. The images on the same column were taken at almost the same time. The regions enclosed by black and gray lines in the images show the detected regions corresponding to target_1 and target_2 respectively. Each figure in the bottom of Fig.8 shows the role of each AVA and the agency organization at such a moment when the same column of images in the upper part were observed. White circles denote freelancer-AVAs, while black and gray circles indicate member-AVAs belonging to agency_1 and agency_2, respectively. Black and gray squares indicate computed locations of target_1 and target_2 respectively. The system worked as follows.

a: Initially, each AVA searched for targets independently.
b: AVA_5 first detected target_1, and agency_1 was formed.
c: All AVAs except for AVA_5 were tracking target_1, while AVA_5 was searching for new targets as a freelancer-AVA.
d: Then, AVA_5 detected target_2 and generated agency_2.
e: The agency restructuring protocol balanced the numbers of member-AVAs in agency_1 and agency_2.
f: Since no AVA could distinguish two targets, the agency unification protocol merged agency_2 into agency_1.
g: Agency_1 activated the agency spawning protocol to generate agency_2 for target_2 detected again.

Table 1. Protocols activated depending on the result of object identification.

Received object information	Identification success	Identification failure
3D view lines of detected objects from a freelancer-AVA	Agency Formation	Agency Formation
3D view line of the target object from a member-AVA	Agency Maintenance	Agency Maintenance and Spawning
3D view lines of non-target objects from a member-AVA	Agency Maintenance	Agency Spawning
3D point of the target object from an agency	Agency Unification	Agency Restructuring

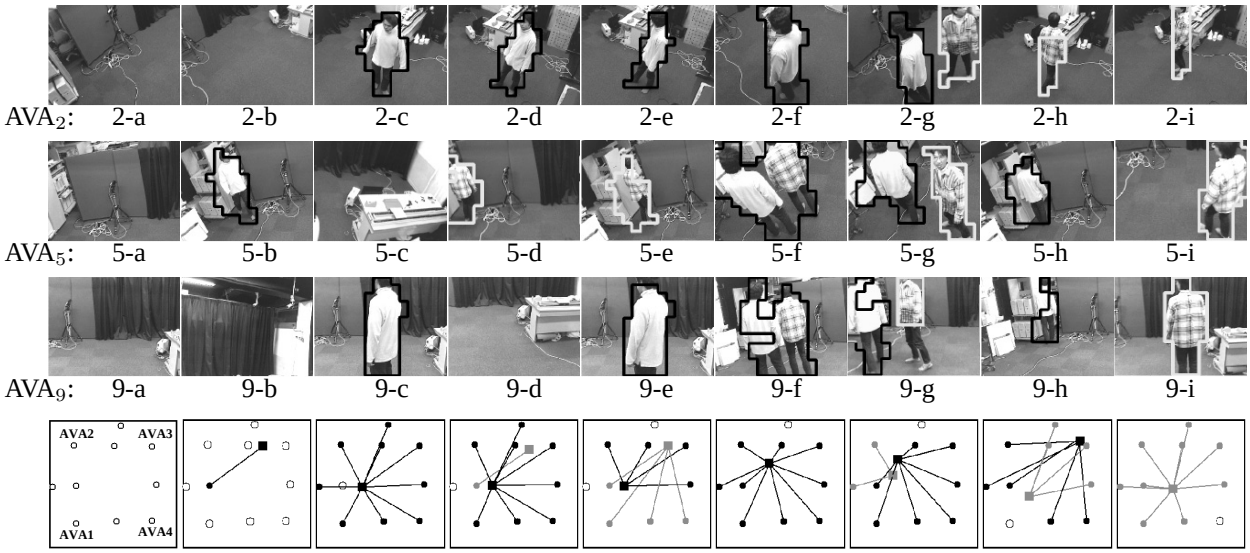
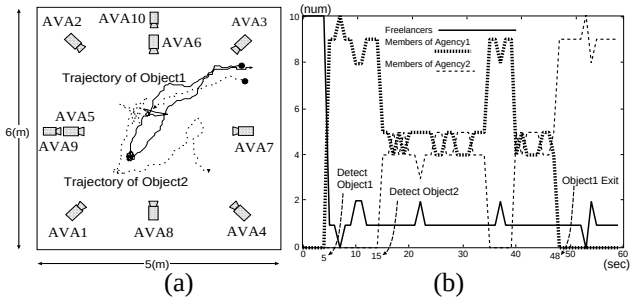


Figure 8. Upper: Partial image sequences, Lower: The role of each AVA and the agency organization.



(a) (b)

Figure 9. (a) Trajectories of the targets, (b) The number of AVAs that performed each role.

h: Target$_1$ was going out of the scene.
i: After agency$_1$ was eliminated, all the AVAs except AVA$_4$ tracked target$_2$.

Fig.9 (a) shows the trajectories of the targets computed by the agency managers. Fig.9 (b) shows the dynamic population changes of freelancer-AVAs, AVAs tracking target$_1$ and those tracking target$_2$.

As we can see, the dynamic cooperations among AVAs and agency managers worked well and enabled the system to keep tracking multiple targets.

6. Concluding Remarks

This paper presented a real-time cooperative multi-target tracking system with multiple active cameras. In our system, parallel processes (i.e., agency managers, AVAs and its constituent perception, action, and communication modules) cooperatively work interacting with each other. As a result, the system as a whole works as a very flexible real-time reactive multi-target tracking system.

References

[1] T. Matsuyama, "Cooperative Distributed Vision - Dynamic Integration of Visual Perception, Action and Communication -", *Proc. of Image Understanding Workshop*, pp.365–384, 1998.

[2] T. Matsuyama, S. Hiura, T. Wada, K. Murase and A. Yoshioka, "Dynamic Memory: Architecture for Real Time Integration of Visual Perception, Camera Action, and Network Communication", *Proc. of CVPR*, pp.728–735, 2000.

[3] A. Nakazawa, H. Kato, S. Hiura and S. Inokuchi, "Tracking multiple people using distributed vision systems", *Proc. of ICRA*, 2002.

[4] K. Toyama, J. Krumm, B. Brumitt and B. Meyers, "WallFlower: Principles and Practice of Background Maintenance", *Proc. of ICCV*, pp.255–261, 1999.

[5] H. Ishiguro, "Distributed Vision System: A Perceptual Information Infrastructure for Robot Navigation", *Proc. of IJCAI-97*, Vol.1, pp.36–41, 1997.

[6] G. P. Stein, "Tracking from Multiple View Points: Self-calibration of Space and Time", *Proc. of CVPR*, Vol. I, pp.521–527, 1999.