Real-Time Multitarget Tracking by a Cooperative Distributed Vision System

TAKASHI MATSUYAMA AND NORIMICHI UKITA

Invited Paper

Target detection and tracking is one of the most important and fundamental technologies to develop real-world computer vision systems such as security and traffic monitoring systems. This paper first categorizes target tracking systems based on characteristics of scenes, tasks, and system architectures. Then we present a real-time cooperative multitarget 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. With this cooperative tracking capability, the system as a whole can track multiple moving objects persistently even under complicated dynamic environments in the real world. In this paper, we address the technologies employed in the system and demonstrate their effectiveness.

Keywords**—***Cooperative distributed vision, cooperative tracking, fixed-viewpoint camera, multi-camera sensing, multitarget tracking, real-time cooperation by multiple agents, real-time tracking.*

I. Introduction

Target detection and tracking is one of the most important and fundamental technologies to develop real-world computer vision systems, e.g., visual surveillance systems and intelligent transport systems (ITSs). (See [1] and [2] for modern visual/video surveillance methods and systems.)

To realize real-time flexible tracking in a wide-spread area, we proposed the idea of *cooperative distributed vision* (CDV) [3]. The goal of CDV is summarized as follows (Fig. 1).

Embed in the real world a group of *active vision agents* (AVA: a network-connected computer with an active camera), and realize:

1) wide-area dynamic scene understanding;
2) versatile scene visualization.

Applications of CDV include real-time wide-area surveillance and traffic monitoring, remote conference and lecturing, 3–D video [4] and intelligent TV studio, and navigation of mobile robots and disabled people.

While the idea of CDV shares much with those of distributed vehicle monitoring testbed (DVMT) [5] and the video surveillance and monitoring (VSAM) project by DARPA [6], our primary interest rests in how we can realize intelligent systems which work adaptively in the real world. Also, we put our focus upon *dynamic interactions among perception, action, and communication*. That is, we believe that intelligence does not dwell solely in the brain but emerges from active interactions with environments through perception, action, and communication.

With this scientific motivation in mind, we designed a real-time cooperative multitarget tracking system, where we developed the following.

Visual Sensor: a *fixed-viewpoint pan-tilt-zoom camera* [7] for wide-area active imaging.

Visual Perception: *active background subtraction* for target detection and tracking [3].

Dynamic Integration of Visual Perception and Camera Action: *dynamic memory architecture* [8] for real-time reactive tracking.

Network Communication for Cooperation: a three-layered dynamic interaction architecture for real-time communication among AVAs.

In this paper, following a categorization of target tracking systems, we address the key ideas of the above mentioned technologies and demonstrate their effectiveness in real-time multitarget tracking. As for technical details of the system, refer to [9].

II. Categorization of Target Tracking Systems

First of all, target tracking systems can be classified into online real-time and off-line batch systems. While the former focus on real-time observation and reactive sensor control, a

Manuscript received May 31, 2001; revised February 15, 2002.

The authors are with the Department of Intelligent Science and Technology, Graduate School of Informatics, Kyoto University, Sakyo, Kyoto, 606-8501 Japan (e-mail: tm@i.kyoto-u.ac.jp; souhaku@i.kyoto-u.ac.jp).

Publisher Item Identifier 10.1109/JPROC.2002.801442.

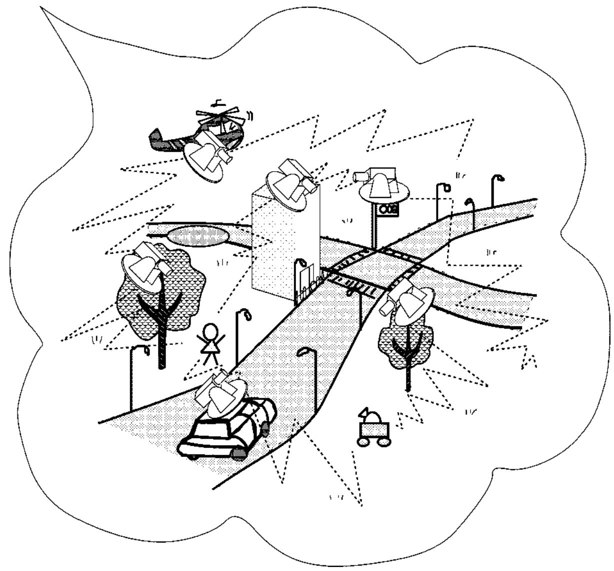


Fig. 1. Cooperative distributed vision.

major interest of the latter rests in computational algorithms to estimate optimal target motion trajectories from a set of *recorded* sensor data [10]–[12].

Real-time target tracking systems, which are the major topic of this paper, can be classified into several different types. For their categorization, the following four characteristics can be used. The first is an assumption about the scene, while the second specifies the task given to a tracking system. The latter two are concerned with system architectures and functions.

1) How Many Objects in the Scene?: If we can assume that just a single object appears in the scene, the task of the system becomes just to detect the object and track it. Even for this simple task, however, the object detection and tracking in real-world environments requires sophisticated image processing methods and object models to discriminate the foreground object from the background scene. In [13], for example, a multiclass statistical model of color and shape was employed to detect and track human heads and hands. In general, flexible object models are required to track articulated and/or deformable objects. The work in [14] used a geometric cardboard model to locate human body parts (head, torso, hands, legs, and feet). Active contour models [15] are very effective to track an object whose shape deforms dynamically (e.g., a beating heart).

When the scene includes multiple objects, the tracking system has to explicitly establish the object identification/discrimination over space and time, which increases the processing complexity of the system significantly. Moreover, the observation of an object is often interfered by others due to occlusion.

2) How Many Target Objects to be Tracked?: Multitarget tracking systems have to solve the above mentioned difficult problems, which leads to the introduction of novel tracking methods and/or the augmentation of system architectures. The former approach includes probabilistic tracking methods [16]–[18], where the object existence in the scene is represented by a probabilistic distribution being dynamically modified based on estimated object motions and observed images. We will discuss the latter approach below.

3) Fixed Camera or Active Camera?: By employing an active camera, the sensing capability of a system is greatly increased.

a) The system can observe a wider area by changing the gazing direction and the position of the camera.

b) The system can dynamically adjust the visual field and the resolution of images by zooming.

The complexity of the system, on the other hand, increases considerably.

a) We have to design real-time and reactive camera control methods taking into account object motion characteristics and mechanical camera dynamics. These topics have been studied in Active Vision [19] and Visual Servo [20], [21].

b) Since camera actions often incur significant changes in observed object appearances, the system has to reason about and/or compensate for such appearance changes to identify and track the object(s). [22], for example, employed a well-calibrated 3-D geometric model of a pan-tilt camera to rectify images taken with different pan-tilt parameters.

4) How Many Cameras?: Employing multiple cameras is one of the most effective methods to solve various problems in target tracking;

a) Continuous wide-area observation: by switching the cameras, the system can continuously track a focused target object even if it moves around the wide area [3].

b) Simultaneous multi-view observation: since multiple different views of the scene can be observed simultaneously, the system can discriminate multiple target objects even if they are occluded in a single view [23]–[25].

c) Reconstruction of 3-D information: when the 3-D positions of all cameras are calibrated, 3-D shape and location of a target object can be reconstructed from 2-D multi-view images. [26]–[28], for example, measured dynamic 3-D human body actions with multiple cameras.

To make full use of these advantages of the multi-camera systems, we have to solve the following camera coordination problems.

- To keep tracking a moving target without a break, a camera needs to request another camera to take over tracking of the target.
- To simultaneously track multiple objects, the system has to discriminate and identify objects detected from multi-view images.
- To robustly reconstruct 3-D information of an object, 2-D multi-view object appearances should be integrated by such computer vision algorithms as stereo matching and volume intersection [26]–[28].

Since each of the above four basic characteristics has two classes, we have $16\ (=2^4)$ classes of target tracking systems in total. A large number of studies have been done on single-target tracking and the current research focus is shifting toward the development of multitarget tracking systems. Among others, it is a good challenge to develop a multitarget tracking system with multiple active cameras, since it includes all properties of the other classes and is the most powerful way to cope with various tasks and complicated situations in the real world.

For such system development, however, we have to solve all problems discussed before. Moreover, we have to additionally solve the dynamic resource allocation problem. That is, since multiple objects move freely in the scene, the system has to adaptively determine which cameras should track which objects depending on dynamic object behaviors. This real-time dynamic resource allocation problem has rarely been studied in computer vision.

In what follows, we present a multitarget tracking system with multiple active cameras we developed, where each active camera is controlled by its corresponding PC and the dynamic resource allocation problem is solved by real-time cooperative network communications among PCs.

III. Fixed-Viewpoint Pan-Tilt-Zoom Camera for Wide-Area Active Imaging

To develop wide-area video surveillance systems, we first of all should study methods of expanding the visual field of a video camera:

1) omnidirectional cameras using fish-eye lenses or curved mirrors [29]–[31];
2) active cameras mounted on computer controlled camera heads [7]–[32].

In the former optical methods, while omnidirectional images can be acquired at video rate, their resolution is limited. In the latter mechanical methods, on the other hand, high-resolution image acquisition is attained at the cost of limited instantaneous visual field.

In our tracking system, we took the active camera method for the following reasons.

1) High-resolution images are of the first importance for object identification and scene visualization.
2) Dynamic visual field and image resolution control can be realized by active zooming.
3) The limited instantaneous visual field problem can be solved by incorporating a group of distributed cameras.

The next problem is how to design an active camera. Suppose we design a pan-tilt camera. This active camera system includes a pair of geometric singularities: 1) the projection center of the imaging system and 2) the pan and tilt rotation axes. In ordinary pan-tilt camera systems, no deliberate design about these singularities is incorporated, which introduces difficult problems in image analysis. That is, the discordance of the singularities causes photometric and geometric appearance variations during the camera rotation: varying highlights and motion parallax. To cope with these appearance variations, consequently, sophisticated image processing should be employed [22].

Our idea to solve this appearance variation problem is very simple but effective [7], [32]:

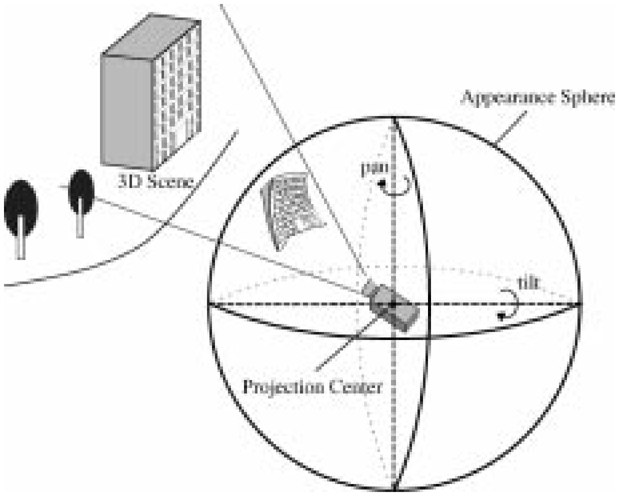


Fig. 2. Fixed-viewpoint pan-tilt camera.

1) Make pan and tilt axes intersect with each other.
2) Place the projection center at the intersecting point.

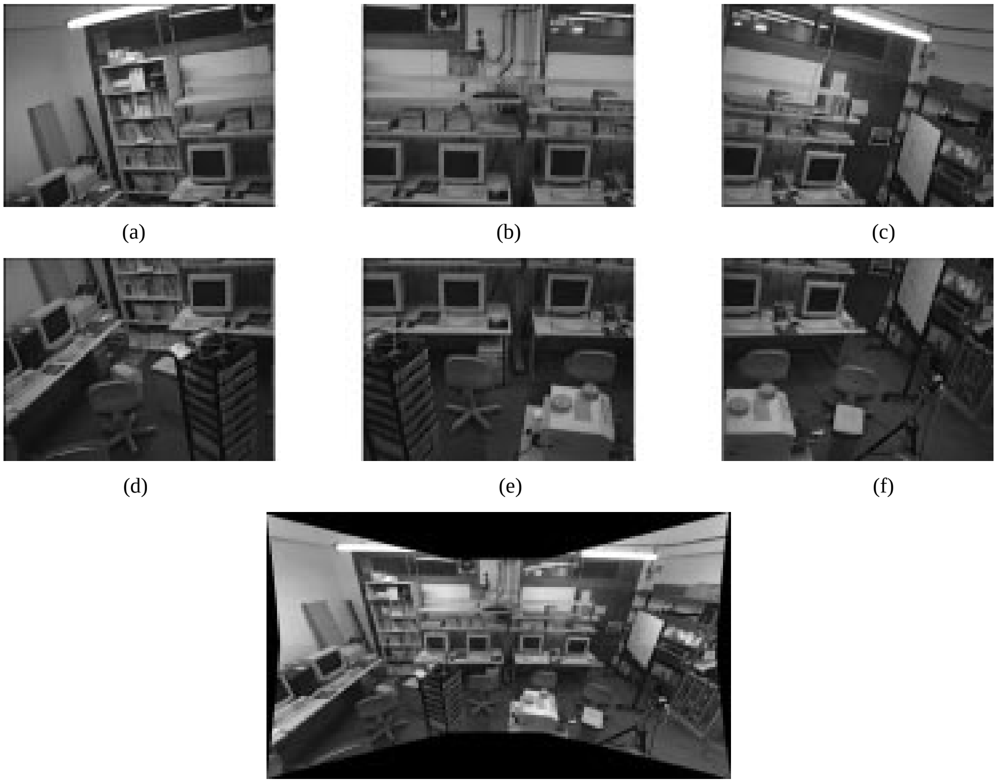
We call the above designed active camera the fixed viewpoint pan-tilt camera. With this camera, all images taken with different pan-tilt angles can be mapped seamlessly onto a common virtual screen (appearance sphere in Fig. 2) to generate a wide panoramic image. Note that, once the panoramic image is obtained, images taken with arbitrary combinations of pan-tilt parameters can be generated by back-projecting the panoramic image onto the corresponding image planes.

Usually, zooming can be modeled by the shift of the projection center along the optical axis [33]. Thus, to realize the fixed viewpoint pan-tilt-zoom (FV-PTZ) camera, either of the following additional mechanisms should be employed.

1) Design such a zoom lens system whose projection center is fixed irrespectively of zooming.
2) Introduce a slide stage to align the projection center depending on zooming.

The above mentioned omnidirectional image representation is equivalent to those proposed in [34] and [35] in computer graphics and virtual reality. Our objective, however, is not to synthesize panoramic images natural to human viewers but to develop an active camera system that facilitates the image analysis for wide-area surveillance. That is, the coordinate system used in the image projection should match accurately with physical camera positioning. To attain such accurate matching, we have to develop sophisticated camera calibration methods [7], [32].

We found that the SONY EVI G20, an off-the-shelf active video camera, is a good approximation of an FV-PTZ camera ($-30^\circ \leq \mathrm{pan} \leq 30^\circ, -15^\circ \leq \mathrm{tilt} \leq 15^\circ$ with zoom $15^\circ \leq \mathrm{horizontal\ view\ angle} \leq 44^\circ$). Then, we developed a sophisticated internal-camera-parameter calibration method for this camera, with which we can use the camera as an FV-PTZ camera [3]. Fig. 3(a) illustrates a set of observed images taken by changing pan-tilt angles with the smallest zooming factor. Fig. 3(b) shows the panoramic image generated from the observed images.



(a) (b) (c)

(d) (e) (f)

(g)

Fig. 3. Panoramic image taken by the developed FV-PTZ camera. (a)–(f) Observed images taken by changing (pan, tilt) angles. (a) $(-30^\circ, 10^\circ)$. (b) $(0^\circ, 10^\circ)$. (c) $(30^\circ, 10^\circ)$. (d) $(-30^\circ, -10^\circ)$. (e) $(0^\circ, -10^\circ)$. (f) $(30^\circ, -10^\circ)$. (g) Generated panoramic image.

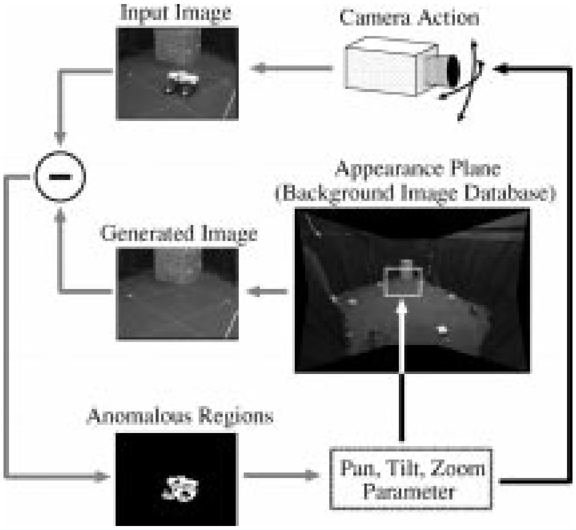


Fig. 4. Active background subtraction with an FV-PTZ camera.

IV. Active Background Subtraction for Target Detection and Tracking

With an FV-PTZ camera, we can easily realize an active target tracking system. Fig. 4 illustrates the basic scheme of the active background subtraction for target detection and tracking we developed [3].

Step 1) Generate the panoramic image of the scene without any objects: Appearance Plane in the figure.

Step 2) Extract a window image from the appearance plane according to the current pan-tilt-zoom parameters and regard it as the *current* background image.

Step 3) Compute difference between the generated background image and the observed image.

Step 4) If anomalous regions are detected in the difference image, select one and control the camera parameters to track the selected target.

Step 5) Go back to Step 2.

To cope with dynamically changing situations in the real world, we have to augment the above scheme at the following three points:

1) robust background subtraction which can work stably under nonstationary environments;
2) flexible system dynamics to control the camera reactively to unpredictable object behaviors;
3) multitarget tracking in cluttered environments.

We do not address the first problem here, since various robust background subtraction methods have been developed [36]–[38]. As for the system dynamics, we will present a novel real-time system architecture in the next section and then, propose a cooperative multitarget tracking system in Section VI.

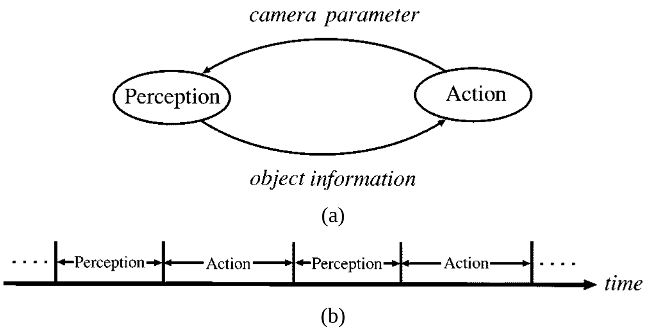


Fig. 5. Dynamic interaction between visual perception and camera action modules. (a) Information flow between visual perception and camera action modules. (b) Dynamics in a sequential target tracking system.

V. Dynamic Integration of Visual Perception and Camera Action for Real-Time Reactive Target Detection and Tracking

The active tracking system described in Fig. 4 can be decomposed into visual perception and camera action modules. The former includes image capturing, background image generation, image subtraction, and object region detection. The latter performs camera control and camera state (i.e., pan-tilt angles and zooming factor) monitoring.

Here we discuss the dynamics of this system. Fig. 5(a) illustrates the information flow between the perception and action modules: the former obtains the *current* camera parameters from the latter to generate the background image and the latter the *current* target location from the former to control the camera. Fig. 5(b) shows the dynamics of the system, where the two modules are activated sequentially.

While this system worked stably [3], the camera motion was not so smooth nor could follow abrupt changes of target motion for the following reasons.

1) The frequency of image observations is limited due to the sequential system dynamics. That is, the perception module should wait for the termination of the *slow* mechanical camera motion.
2) Due to delays involved in image processing, camera state monitoring, and mechanical camera motion, the perception and action modules cannot obtain accurate *current* camera state or target location respectively.

To solve these problems and realize real-time reactive target tracking, we proposed a novel dynamic system architecture named *dynamic memory architecture* [8], where the visual perception and camera action modules run in parallel and dynamically exchange information via a specialized shared memory named the *dynamic memory* (Fig. 6).

A. Access Methods for the Dynamic Memory

While the system architecture consisting of multiple parallel processes with a common shared memory looks similar to the "whiteboard architecture" [39] and the "smart buffer" [40], the critical difference rests in that each variable in the dynamic memory stores a discrete temporal sequence of values and is associated with the following temporal interpolation and prediction functions (Fig. 7).

The write and read operations to/from the dynamic memory are defined as follows.

1) Write Operation: When a process computes a value *val* of a variable $\mathbf{v}$ at a certain moment t, it writes (val, t) into the dynamic memory. Since such computation is done repeatedly according to the dynamics of the process, a discrete temporal sequence of values is recorded for each variable in the dynamic memory (a sequence of black dots in Fig. 7).

2) Read Operation:

a) Temporal interpolation: A reader process runs in parallel to the writer process and tries to read from the dynamic memory the value of the variable $\mathbf{v}$ at a certain moment, e.g., the value at T_1 in Fig. 7. When no value is recorded at the specified moment, the dynamic memory interpolates it from recorded data. With this function, the reader process can read a value at any temporal moment along the continuous temporal axis without making any synchronization with the writer process.

b) Future prediction: A reader process may run fast and require data which are not written yet by the writer process (for example, the value at T_3 in Fig. 7). In such a case, the dynamic memory predicts an expected value in the future based on those data recorded so far and returns it to the reader process.

3) Specification and Modification of Dynamics: Since a variable in the dynamic memory represents a state of some dynamic object (e.g., pan-tilt-zoom parameters of an active camera), the interpolation and prediction functions associated with the variable should be designed to model well the dynamics of the object. When a writer process declares a variable in the dynamic memory, the process specifies the dynamics of the variable in terms of interpolation and prediction functions, which may be acquired by off-line calibrations or updated online by adaptive modeling methods such as Kalman filtering. Dynamics (i.e., interpolation and prediction functions) of some variable may be changed during processing by the writer process. For example, when the camera action module issues a speed control command to a pan-tilt camera, dynamics of variables representing pan and tilt angles should be changed accordingly.

With the above described functions, each process can get any data along the temporal axis freely without waiting (i.e., wasting time) for synchronization with others. This no-wait asynchronous module interaction capability greatly facilitates the implementation of real-time reactive systems. As will be shown later in Section VI–B3, moreover, the dynamic memory supports the *virtual synchronization* between multiple network-connected systems (i.e., AVAs), which facilitates the real-time dynamic cooperation among the systems.

B. Effectiveness of the Dynamic Memory

To verify the effectiveness of the dynamic memory, we developed a real-time single-target tracking system and conducted experiments of tracking a radio-controlled car in a computer room. The system employed the parallel active

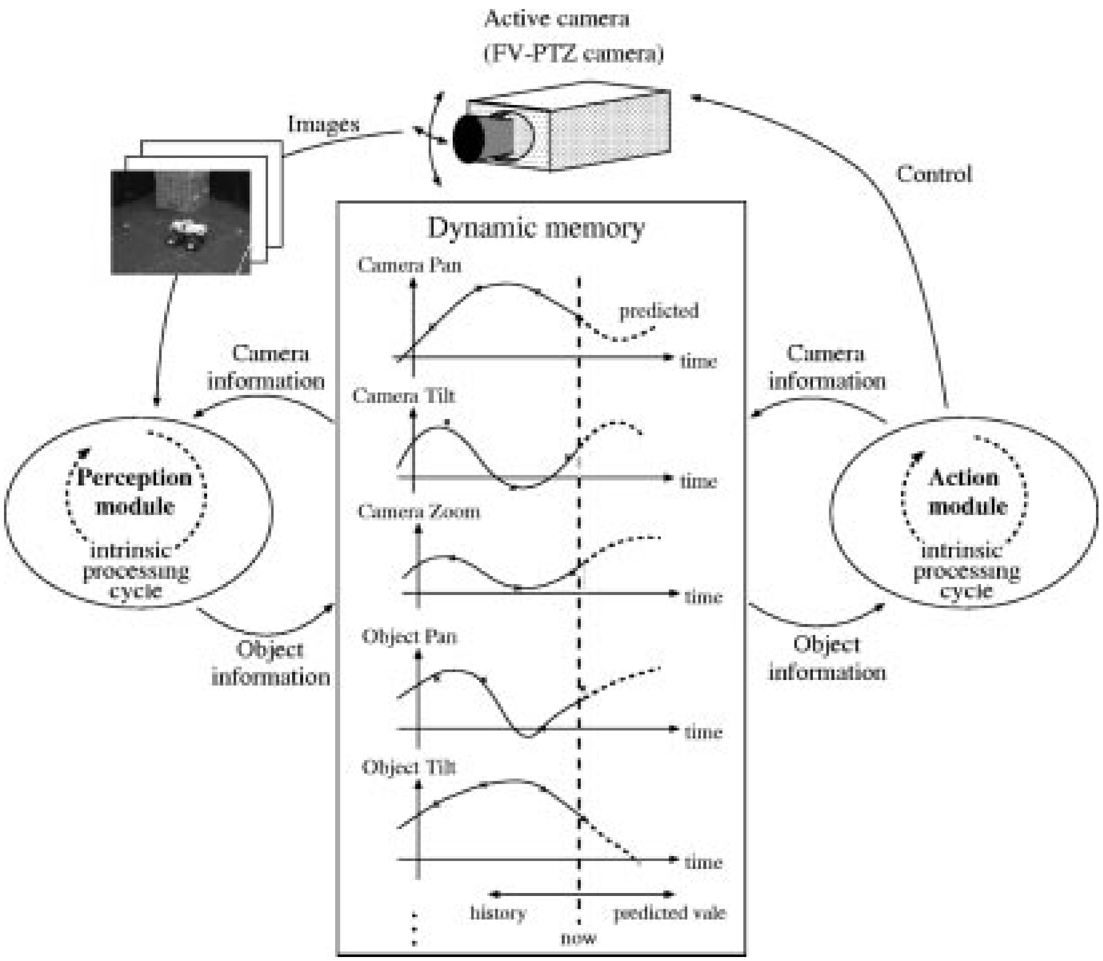


Fig. 6. Real-time reactive target tracking system with the dynamic memory.

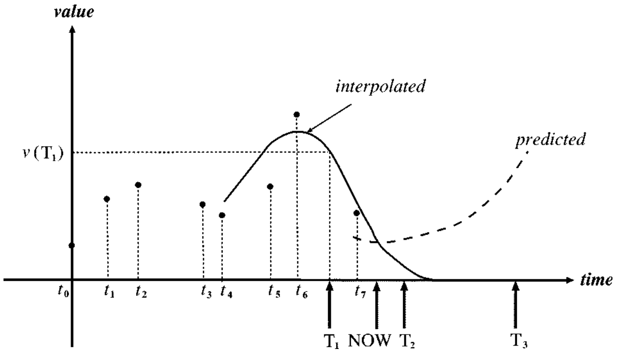


Fig. 7. Representation of a time-varying variable in the dynamic memory.

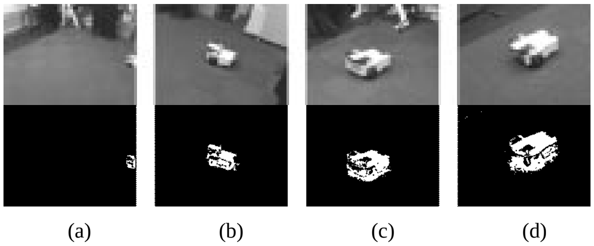
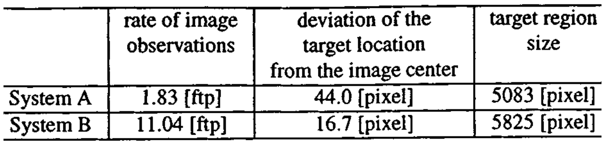

(a) (b) (c) (d)

Fig. 8. Observed image sequence taken by the system. Upper: input images. Lower: detected object regions. (a) Frame 0. (b) Frame 50. (c) Frame 100. (d) Frame 150.

background subtraction method with the FV-PTZ camera, where the perception and action modules were implemented as UNIX processes sharing the dynamic memory. Fig. 8 illustrates a partial sequence of observed images and detected object regions. Note that the accurate calibration of the FV-PTZ camera enabled the stable background subtraction even while changing pan, tilt, and zooming.

Table 1 compares the performance between System A (sequential dynamics) and System B (parallel dynamics with the dynamic memory). Both systems tracked a computer-controlled toy car under the same experimental settings and performance factors were averaged over about 30 s. The left column of the table shows that the dynamic memory greatly improved the rate of image observations owing to the no-wait asynchronous execution of the perception module. The other two columns verify the improvements in the camera control. That is, with the dynamic memory, the camera was directed toward the target more accurately (the middle column) and hence could observe the target in higher resolution (the right column). Note that our system controls pan-tilt angles to observe the target at the image center and adjusts the zooming factor depending on deviations of the former from the latter: smaller deviations lead to zooming in to capture higher resolution target images, while larger deviations to zooming out not to miss the target [3].

Table 1
Performance Evaluation



	rate of image observations	deviation of the target location from the image center	target region size
System A	1.83 [ftp]	44.0 [pixel]	5083 [pixel]
System B	11.04 [ftp]	16.7 [pixel]	5825 [pixel]

Fig. 9 illustrates target and camera motion trajectory data written into and read from the dynamic memory, where:

- graph 1 (upper right) represents pan-tilt camera positions measured from the camera by the action module;
- graph 2 (upper left) represents pan-tilt camera positions read from the dynamic memory by the perception module;
- graph 3 (lower left) represents target locations computed from observed images by the perception module;
- graph 4 (lower right) represents target locations read from the dynamic memory by the action module.

Each graph includes a pair of trajectories: the larger amplitude denotes the pan angle and the smaller the tilt. Note that the target locations as well as the camera positions are described in terms of (pan, tilt).

We make the following observations.

1) Comparing graph 1 with graph 2, the data density of the latter is higher than that of the former. This is because the perception module runs about two times faster and hence reads pan-tilt camera position data more frequently. This holds also for graphs 3 and 4.
2) The camera control is well synchronized with the target motion; by overlaying graphs 1 and 3, they match very well. That is, the camera follows the target motion without delay. This is because the action module employed precise prediction-based camera control to cope with various delays involved in image processing and camera motion.

VI. Cooperative Multi-Target Tracking

Now we address cooperative multitarget tracking by communicating active vision agents (AVAs), where an AVA denotes an augmented target tracking system described in the previous section. The augmentation means that an AVA consists of visual perception, camera action, and network communication modules, which run in parallel exchanging information via the dynamic memory.

A. Basic Scheme for Cooperative Tracking

Our multitarget tracking system consists of a group of AVAs embedded in the real world (Fig. 1). 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 a target that comes into its observable area. Such AVA that is searching for a target is called a *freelancer*.
2) If an AVA detects a target, it navigates the gazes of the other AVAs toward that target [Fig. 10(a)].
3) A group of AVAs which gaze at the same target form what we call an *Agency* and keep measuring the 3-D information of the target from multi-view images [Fig. 10(b)].
4) Depending on target locations in the scene, each AVA dynamically changes its target [Fig. 10(c)].

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

- **Multitarget identification**: to gaze at each target, the system has to distinguish multiple targets.
- **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 and 2) to track each moving target persistently, the system has to adaptively determine which AVAs should track which targets.

In what follows, we address how these problems can be solved by real-time cooperative communications among AVAs.

B. Three-Layered Dynamic Interactions for Cooperative Tracking

We designed and implemented the three-layered dynamic interaction architecture illustrated in Fig. 11 to realize real-time cooperative multitarget tracking.

1) Intra-AVA Layer: In the lowest layer in Fig. 11, perception, action, and communication modules that compose an AVA interact with each other via the dynamic memory.

An AVA is an augmented target tracking system described in Section 5, where the augmentation is threefold.

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

b) Gaze control based on the 3-D target position: When the FV-PTZ camera is ready to accept a control command, the action module reads the 3-D view line toward the target (i.e., $\hat{L}(\mathrm{now})$) 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 3-D position of the target (denoted by $\hat{P}(t)$) and sends it to all member AVAs, which then is written into

[1] The 3-D line determined by the projection center of the camera and an object region centroid.

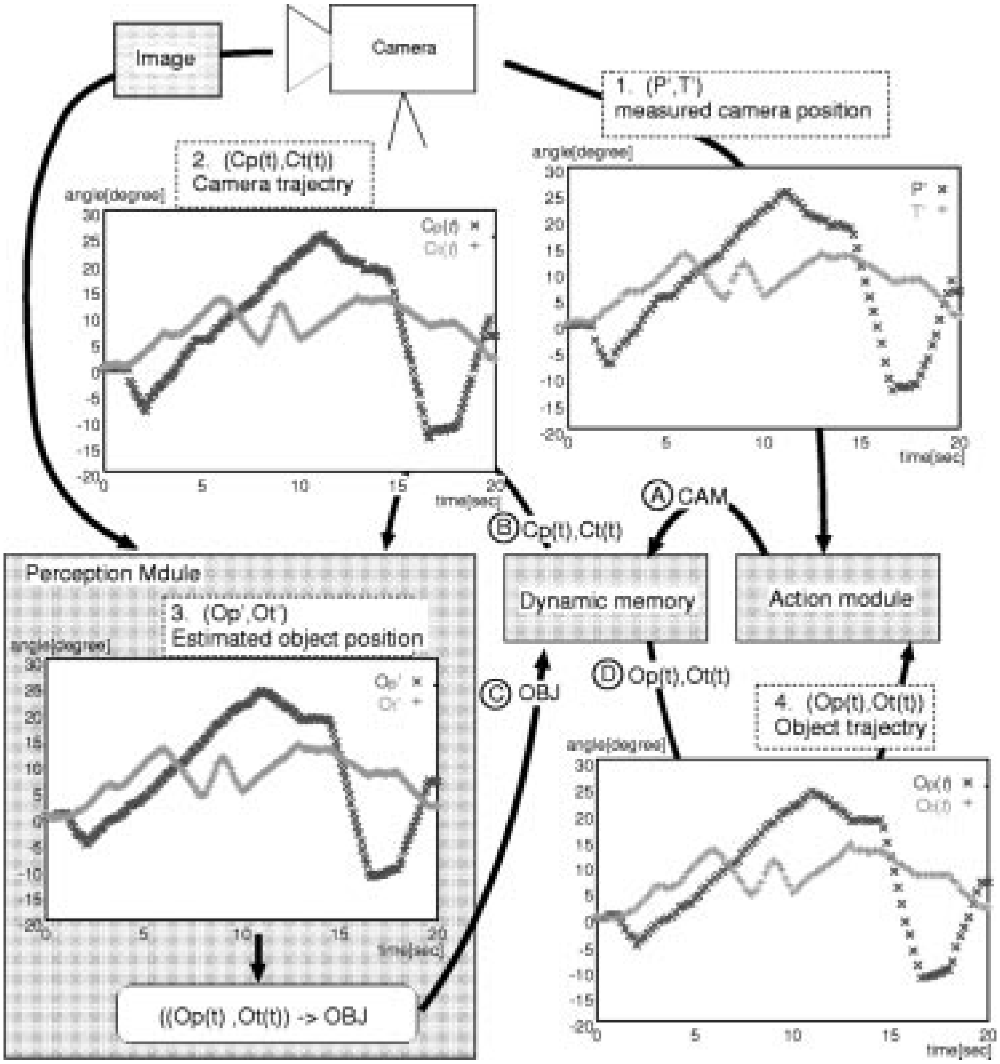
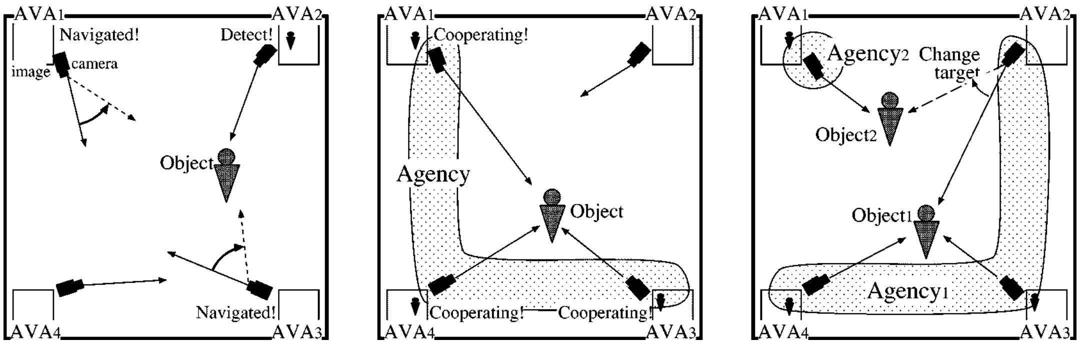


Fig. 9. Dynamic data exchanged between the perception and action modules.



(a) (b) (c)

Fig. 10. Basic scheme for cooperative tracking. (a) Gaze navigation. (b) Cooperative gazing. (c) Adaptive target switching.

the dynamic memory by the communication module. If such information is available, the action module controls the camera based on $\hat{P}(\mathrm{now})$ instead of $\hat{L}(\mathrm{now})$.

c) Incorporation of the communication module: Data exchanged by the communication module over the network can be classified into two types: detected object data and messages for cooperations among AVAs. The former include 3-D view lines toward detected objects: AVA $\rightarrow$ other AVAs and agencies, and 3-D target position: agency $\rightarrow$ member AVAs. The latter realize various communication protocols, which will be described later.

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. The middle layer in

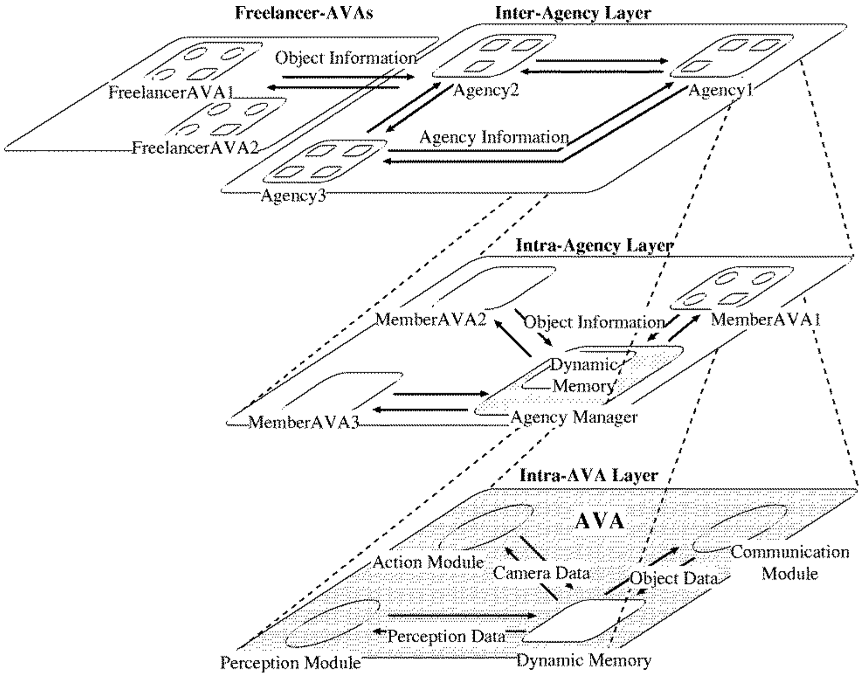


Fig. 11. Three-layered dynamic interaction architecture.

Fig. 11 specifies dynamic interactions between an agency manager and 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.

a) Spatial object identification: The agency manager has to establish the object identification between the groups of the 3-D view lines detected and transmitted by its member AVAs. The agency manager checks distances between those 3-D view lines detected by different member AVAs and computes the 3-D target position from a set of nearly intersecting 3-D view lines. The manager employs what we call the *virtual synchronization* to virtually adjust observation timings of the 3-D view lines (see Section VI–B3 for details). Note that the manager may find none or multiple sets of such nearly intersecting 3-D view lines. To cope with these situations, the manager conducts the following temporal object identification.

b) Temporal object identification: The manager records the 3-D trajectory of its target, with which the 3-D object position(s) computed by the spatial object identification is compared. That is, when multiple 3-D locations are obtained by the spatial object identification, the manager selects the one closest to the target trajectory. When the spatial object identification failed and no 3-D object location was obtained, on the other hand, the manager selects the 3-D view line that is closest to the latest recorded target 3-D position. Then the manager projects the target 3-D position onto the selected view line to estimate the new 3-D 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 2-D tracking by itself.

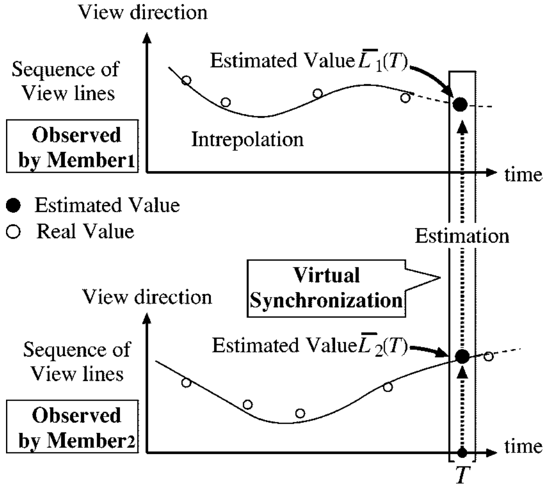


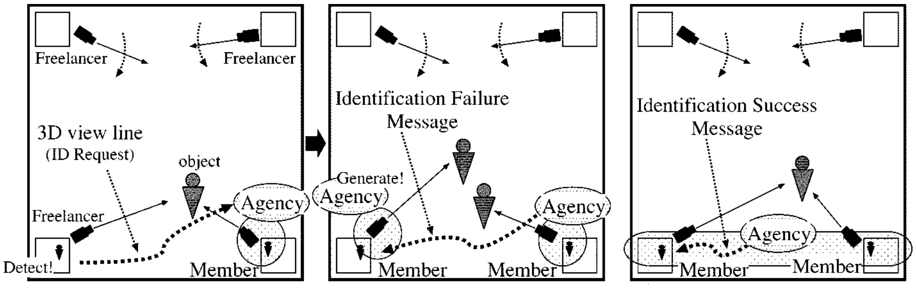
Fig. 12. Virtual synchronization for spatial object identification.

3) Virtual Synchronization: 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. These asynchronous activities can significantly damage the reliability of the spatial object identification.

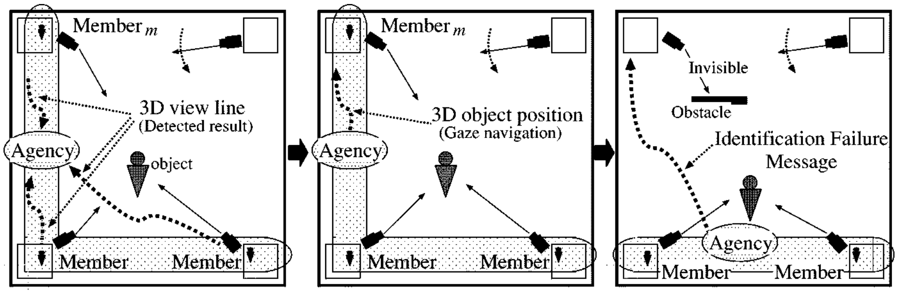
To solve this problem, we introduce the dynamic memory into an agency manager, which enables the manager to virtually synchronize any asynchronously observed/transmitted data. We call this function *virtual synchronization* by the dynamic memory.

Fig. 12 shows the mechanism of the virtual synchronization. All 3-D view lines computed by each member



(a) (b) (c)

Fig. 13. Agency formation (see text).



(a) (b) (c)

Fig. 14. Agency maintenance (see text).

AVA are transmitted to the agency manager, which then records them into its internal dynamic memory. Fig. 12, for example, shows a pair of temporal sequences of 3-D view line data transmitted from member AVA$_1$ and member AVA$_2$, respectively. When the manager wants to establish the spatial object identification at T, it can read the pair of the synchronized 3-D view line data at T from the dynamic memory (i.e., $\bar{L}_1(T)$ and $\bar{L}_2(T)$ in Fig. 12). That is, the values of the 3-D view lines used for the identification are completely synchronized with that identification timing even if their measurements are conducted asynchronously.

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

4) Communications at the Intra-Agency Layer: The above mentioned temporal object identification fails if the closest distance between the estimated and observed 3-D target locations exceeds a threshold. The following three communication protocols are activated depending on the success or failure of the object identification. They materialize dynamic interactions at the intra-agency layer.

a) Agency formation protocol: This protocol defines: 1) the new agency generation procedure by a freelancer AVA and 2) the 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 the identification between the detected object and the target object of each agency [Fig. 13(a)]. Depending on the result of this object identification, the freelancer AVA works as follows.

No agency established the object identification: The freelancer AVA generates a new agency manager to track the newly detected object and joins into that agency as its member AVA [Fig. 13(b)].

An agency established the object identification: The freelancer-AVA joins into the agency that has made successful object identification, if requested [Fig. 13(c)].

b) Agency maintenance protocol: This protocol defines procedures for the continuous maintenance of an agency and the elimination of an agency.

After an agency is generated, the agency manager repeats the spatial and temporal object identifications for cooperative tracking [Fig. 14(a)]. Following the spatial object identification, the manager transmits the newest 3-D target location to each member AVA [Fig. 14(b)], which then is recorded into the dynamic memory of the member AVA.

Suppose a member AVA$_m$ cannot detect the target object due to an obstacle or processing errors [Fig. 14(c)]. Even in this case, the manager informs AVA$_m$ the 3-D position of the target observed by the other member AVAs. This information navigates the gaze of AVA$_m$ toward the (invisible) target. However, if such misdetection continues for a long time, the agency manager forces AVA$_m$ out of the agency to be a freelancer.

If all member AVAs cannot observe the target being tracked so far, the agency manager destroys the agency and makes all its member AVAs become freelancers.

c) Agency spawning protocol: This protocol defines a new agency generation procedure from an existing agency.

After the spatial and temporal object identifications, the agency manager may find such a 3-D view line(s) that does

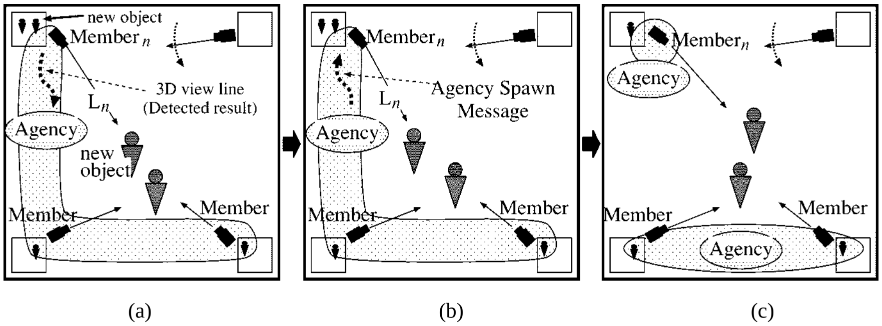


Fig. 15. Agency spawning (see text).

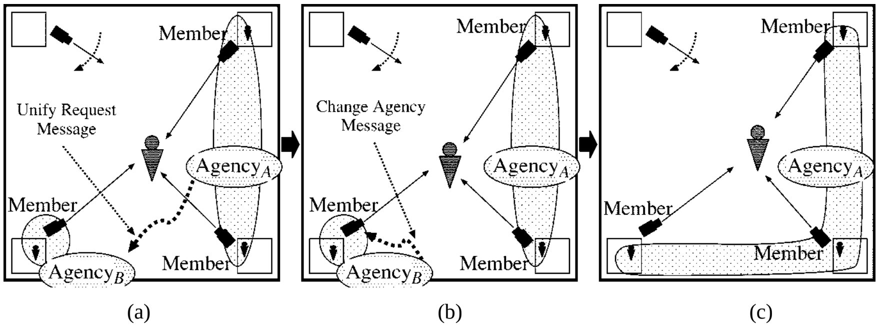


Fig. 16. Agency unification (see text).

not correspond to the target. This means the detection of a new object by its member AVA. Let L_n denote such 3-D view line detected by AVA_n [Fig. 15(a)]. Then, the manager broadcasts L_n to other agency managers to examine the identification between L_n and their tracking targets.

If none of the identification is successful, the agency manager makes AVA_n quit from the current agency and generate a new agency [Fig. 15(b)]. AVA_n then joins into the new agency [Fig. 15(c)].

5) Inter-Agency Layer: In multitarget 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 (the top layer in Fig. 11).

The dynamic interactions between agency managers are triggered based on the object identification across agencies. That is, when a new target 3-D location is obtained, agency manager AM_i broadcasts it to the others. Agency manager AM_j, which receives this information, compares it with the 3-D position of its own target to check the object identification. Note that here also the virtual synchronization between a pair of 3-D target locations is employed to increase the reliability of the 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.

In principle, the system should keep the one-to-one correspondence between agencies and target objects. However, this correspondence sometimes is violated due to failures of object identification and discrimination:

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

Fig. 16 shows an example. When agency manager AM_A of agency_A establishes the identification between its own target and the one tracked by AM_B, AM_A asks AM_B to be merged into AM_A [Fig. 16(a)]. Then, AM_B asks its member AVAs to join into agency_A [Fig. 16(b)]. After copying the target information recorded in the dynamic memory into the object trajectory database, AM_B eliminates itself [Fig. 16(c)].

As noted above, agencies corresponding to multiple different targets may be unified if they are very close. However, this heterogeneously unified agency can be separated back by the agency spawning protocol when the distance between the targets get larger. In such case, characteristics of the newly detected target are compared with those recorded in the object trajectory database to check if the new target corresponds to a target that had been tracked before. If so, the corresponding target trajectory data is moved from the database into the dynamic memory of the newly generated agency.

b) Agency restructuring protocol: When the inter-agency object identification fails, agency manager AM_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.

Fig. 17 illustrates an example. Agency manager AM_C of agency_C sends its target information to AM_D, which fails in the object identification. Then, AM_D asks AM_C to trade its member AVA into AM_D [Fig. 17(a)]. When requested, AM_C selects its member AVA and asks it to move to agency_D [Fig. 17(b) and (c)].

6) Communication With Freelancer AVAs: An agency manager communicates with freelancer AVAs as well as with

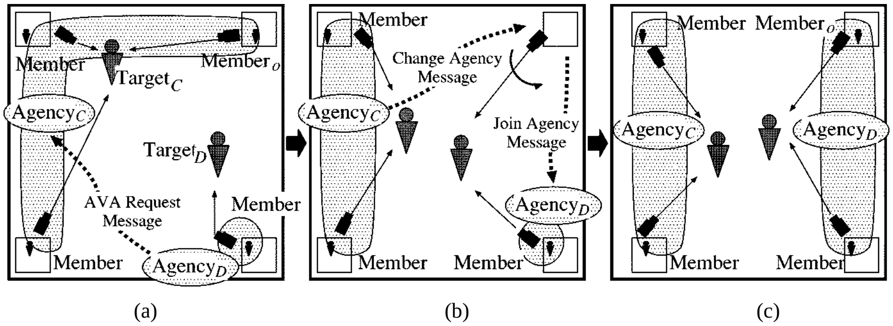


Fig. 17. Agency restructuring (see text).

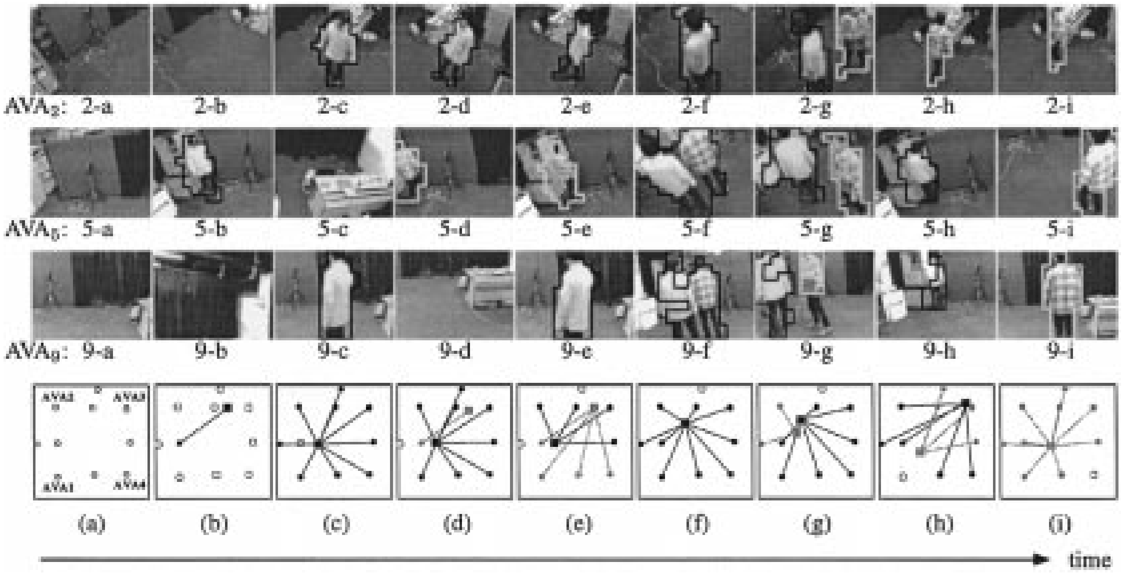

AVA$_2$: 2-a 2-b 2-c 2-d 2-e 2-f 2-g 2-h 2-i
AVA$_5$: 5-a 5-b 5-c 5-d 5-e 5-f 5-g 5-h 5-i
AVA$_9$: 9-a 9-b 9-c 9-d 9-e 9-f 9-g 9-h 9-i
AVA2 AVA3 AVA1 AVA4
(a) (b) (c) (d) (e) (f) (g) (h) (i)
time

Fig. 18. Experimental results.

other managers (the top row of Fig. 11). As described in the agency formation protocol in Section VI–B4, a freelancer activates the communication with agency managers when it detects an object. An agency manager, on the other hand, sends to freelancers its target position when the new data are obtained. Then, each freelancer decides whether it continues to be a freelancer or joins into the agency depending on the target position and the current number of freelancers in the system. Note that in our system a user can specify the number of freelancers to be preserved while tracking targets.

VII. Experiments

To verify the effectiveness of the proposed system, we conducted experiments on multiple human tracking in a room (about 5 m × 5 m). The system consists of ten AVAs. Each AVA is implemented on a network-connected PC (Pentium III 600-MHz × 2) with an FV-PTZ camera (SONY EVI-G20), where the perception, action, and communication modules as well as agency managers are realized as UNIX processes. Fig. 19(a) illustrates the camera layout: camera_9 and camera_{10} are on the walls, while the others on the ceiling. The external camera parameters are calibrated. Note that the internal clocks of all the PCs are synchronized by the Network Time Protocol to realize the virtual synchronization. With this architecture, the perception module of each AVA can capture images (320 × 240 8-b black-and-white images) and detect objects at about 10 frames per second on average.

In the experiment, the system tracked two people. Target_1 first came into the scene, and after a while target_2 came into the scene. Both targets then moved freely. The upper part of Fig. 18 shows the partial image sequences observed by AVA_2, AVA_5, and AVA_9. The images on the same row were taken by the same AVA. 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. Note that the image sequences in Fig. 18 are not recorded ones but are captured real-time according to target motion.

Each figure in the bottom of Fig. 18 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.

1) Initially, each AVA searched for an object independently.
2) AVA_5 first detected target_1, and agency_1 was formed.

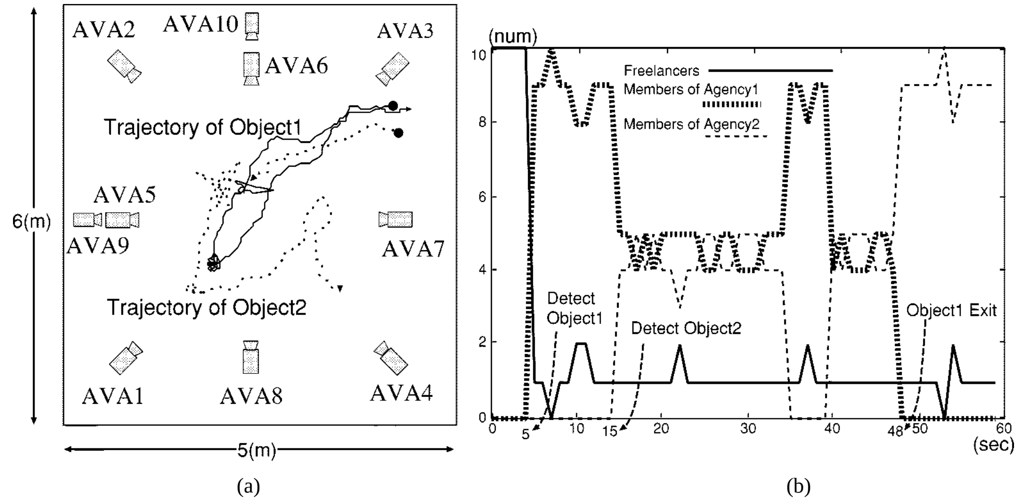


Fig. 19. Experimental results. (a) Trajectories of the targets. (b) The number of AVAs that performed each role.

3) All AVAs except for AVA_5 were tracking target_1, while AVA_5 was searching for a new object as a freelancer.
4) Then, AVA_5 detected target_2 and generated agency_2.
5) The agency restructuring protocol balanced the numbers of member AVAs in agency_1 and agency_2. Note that AVA_9 and AVA_{10} were working as freelancers.
6) Since two targets came very close to each other and no AVA could distinguish them, the agency unification protocol merged agency_2 into agency_1.
7) When the targets moved apart, agency_1 detected a “new” target. Then, it activated the agency spawning protocol to generate agency_2 again for target_2.
8) Target_1 was going out of the scene.
9) After agency_1 was eliminated, all the AVAs except AVA_4 tracked target_2.

Fig. 19(a) shows the trajectories of the targets computed by the agency managers. Fig. 19(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 very well and enabled the system to persistently track multiple targets. In [9], we discussed the soundness (i.e., deadlock-free interaction) and the completeness (i.e., the number of trackable objects) of the system.

VIII. Concluding Remarks

This paper presented a real-time active multitarget tracking system, which is the most powerful and flexible but difficult to realize among various types of target tracking systems.

To implement the system, we developed: 1) a fixed-viewpoint pan-tilt-zoom camera for wide area active imaging; 2) active background subtraction for target detection and tracking; 3) dynamic memory architecture for real-time reactive tracking; and 4) a three-layered dynamic interaction architecture for real-time communication among active vision agents.

While many visual/video surveillance systems have been developed [1], [2], most of them put their focus onto object detection and tracking methods and little studies have been done on how visual perception and camera action modules are integrated dynamically. In this sense, we believe the dynamic memory architecture is one of distinguishing characteristics of our system. In fact, with this architecture, our system can track targets reactively even if they are very close to a camera (i.e., even if apparent object motion on the image is large).

The most distinguishing characteristics of our system rests in the introduction of real-time network communications among multiple distributed AVAs. While some systems incorporated distributed sensors for surveillance [23]–[25], [41], [42], their communication protocols are not so sophisticated nor integrated real-time with perception or camera action processes. In our system, the dynamic memory architecture enables distributed AVAs to *virtually synchronize* their activities over the network. That is, in our system, all parallel processes (i.e., 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 multitarget tracking system. We believe that this cooperative distributed processing greatly increases the flexibility and adaptability of the system, which has been verified by experiments of multiple human tracking.

One of the most frequently asked questions about our system is how many targets the system with N AVAs can track. The answer is at most N but for stable tracking at most $N/2$. This limitation comes from the communication protocol employed in the current system. That is, while each AVA can detect multiple objects simultaneously, only one of them is identified as the target by the agency manager to which it belongs. In other words, the current system employs a rather strict constraint to coordinate a group of AVAs: each AVA can belong to only one agency, which has one-to-one correspondence to a target. This constraint limits the maximum number of trackable targets. For stable target tracking, moreover, the system needs 3-D location of each target, for which at least two AVAs should track a target. To increase the number of trackable targets, we should modify the communication protocol so that each AVA can belong to multiple agencies at the same time. While this modification introduces some complications into the protocol, it is not hard to implement.

As shown in Fig. 19(a), the system is installed in a rather small room. This means the system can observe very high-resolution multiview images of a target. To make full use of this characteristics, we developed a real-time dynamic 3-D shape reconstruction system by adding an ultra high speed network for parallel processing [26]. This system tracks a moving (e.g., dancing) human and reconstructs its dynamic 3-D shape from multiview video data. While the implemented system can reconstruct about 12 human volumes per second in 2 cm × 2 cm × 2 cm spatial resolution with fixed cameras, the processing speed slows down to about 1 volume per second with active cameras. Currently we are studying how we can integrate the cooperative multitarget tracking capability described in this paper with parallel processing for real-time 3-D shape reconstruction. With this new integrated system, we will be able to capture full 3-D video of various dancing and sports activities played by multiple people in rather wide areas.

References

[1] R. T. Collins, A. J. Lipton, and T. Kanade, Eds., *IEEE Trans. Pattern Anal. Machine Intell. (Special Section on Video Surveillance)*, Aug. 2000, vol. 22.

[2] C. S. Regazzoni, V. Ramesh, and G. L. Foresti, Eds., *Proc. IEEE (Special Issue on Third Generation Surveillance Systems)*, Oct. 2001, vol. 89.

[3] T. Matsuyama, "Cooperative distributed vision—Dynamic integration of visual perception, action and communication," in *Proc. Image Understanding Workshop*, 1998, pp. 365–384.

[4] S. Moezzi, L. Tai, and P. Gerard, "Virtual view generation for 3D digital video," *IEEE Multimedia*, pp. 18–26, 1997.

[5] V. R. Lesser and D. D. Corkill, "The distributed vehicle monitoring testbed: A tool for investigating distributed problem solving networks," *AI Mag.*, vol. 4, no. 3, pp. 15–33, 1983.

[6] G. E. Lukes, Ed., "Video surveillance and monitoring," in *Proc. Image Understanding Workshop*, 1998, vol. 1, pp. 3–400.

[7] T. Wada and T. Matsuyama, "Appearance sphere: Background model for pan-tilt-zoom camera," in *Proc. ICPR*, vol. A, 1996, pp. 718–722.

[8] T. Matsuyama *et al.*, "Dynamic memory: Architecture for real time integration of visual perception, camera action, and network communication," in *Proc. CVPR*, 2000, pp. 728–735.

[9] N. Ukita, "Real-time cooperative multi-target tracking by communicating active vision agents," Ph.D. dissertation, Kyoto University, Japan, 2001.

[10] K. R. Pattipati, S. Deb, Y. Bar-Shalom, and R. B. Washbarn, "A new relaxation algorithm and passive sensor data association," *IEEE Trans. Automat. Contr.*, vol. 37, no. 2, pp. 198–213, 1992.

[11] K. W. Lo and C. K. Li, "An improved multiple target angle tracking algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 28, no. 3, pp. 797–805, 1992.

[12] V. Kettnaker and R. Zabih, "Bayesian multi-camera surveillance," in *Proc. CVPR*, vol. 2, 1999, pp. 253–259.

[13] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 780–785, July 1997.

[14] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: Who? When? Where? What? A real-time system for detecting and tracking people," *Proc. 3rd IEEE Int. Conf. Automatic Face and Gesture Recognition*, pp. 222–227, 1998.

[15] A. Blake and M. Isard, *Active Contours*. Berlin, Germany: Springer-Verlag, 1998.

[16] M. Isard and A. Blake, "Condensation—Conditional density propagation for visual tracking," *Int. J. Comput. Vis.*, vol. 29, no. 1, pp. 5–28, 1998.

[17] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of nonrigid objects using mean shift," in *Proc. CVPR*, vol. 2, 2000, pp. 142–149.

[18] M. Isard and J. MacCormick, "BraMBle: A bayesian multiple-blob tracker," in *Proc. 8th ICCV*, vol. 2, 2001, pp. 34–41.

[19] Y. Aloimonos, Ed., *Active Perception*. Hillsdale, NJ: Lawrence Erlbaum Associates, 1993.

[20] L. E. Weiss, A. C. Sanderson, and C. P. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Trans. Robot. Automat.*, vol. RA-3, pp. 404–417, Oct. 1987.

[21] C. M. Brown, "Gaze control with interactions and delays," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 518–527, 1990.

[22] D. Murray and A. Basu, "Motion tracking with an active camera," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 449–459, May 1994.

[23] A. Utsumi, H. Mori, J. Ohya, and M. Yachida, "Multiple-view-based tracking of multiple humans," in *Proc. 14th ICPR*, vol. 1, 1998, pp. 597–601.

[24] A. Nakazawa, H. Kato, and S. Inokuchi, "Human tracking using distributed vision systems," in *Proc. 14th ICPR*, 1998, pp. 593–596.

[25] T. Sogo, H. Ishiguro, and M. M. Trivedi, "Real-time target localization and tracking by N-ocular stereo," in *IEEE Workshop on Omnidirectional Vision (OMNIVIS'00)*, 2000, pp. 153–160.

[26] T. Wada, X. Wu, S. Tokai, and T. Matsuyama, "Homography based parallel volume intersection: Toward real-time volume reconstruction using active cameras," in *Proc. Computer Architectures for Machine Perception*, 2000, pp. 331–339.

[27] E. Borovikov and L. Davis, "A distributed system for real-time volume reconstruction," in *Proc. Computer Architectures for Machine Perception*, 2000, pp. 183–189.

[28] G. Cheung and T. Kanade, "A real time system for robust 3D voxel reconstruction of human motions," in *Proc. CVPR*, 2000, pp. 714–720.

[29] Y. Yagi and M. Yachida, "Real-time generation of environmental map and obstacle avoidance using omnidirectional image sensor with conic mirror," in *Prof. CVPR*, 1991, pp. 160–165.

[30] K. Yamazawa, Y. Yagi, and M. Yachida, "Obstacle detection with omnidirectional image sensor HyperOmni vision," in *Proc. ICRA*, 1995, pp. 1062–1067.

[31] V. N. Peri and S. K. Nayar, "Generation of perspective and panoramic video from omnidirectional video," in *Proc. IUW*, 1997, pp. 243–245.

[32] S. Coorg and S. Teller, "Spherical mosaics with quaternions and dense correlation," *Int. J. Comput. Vis.*, vol. 37, no. 3, pp. 259–273, 2000.

[33] J. M. Lavest, C. Delherm, B. Peuchot, and N. Daucher, "Implicit reconstruction by zooming," *Comput. Vis. Image Understanding*, vol. 66, no. 3, pp. 301–315, 1997.

[34] N. Greene, "Environment mapping and other applications of world projections," *Comput. Graph. Applicat.*, vol. 6, no. 11, pp. 21–29, 1986.

[35] S. E. Chen, "QuickTime VR—An image-based approach to virtual environment navigation," in *Proc. SIGGRAPH'95*, 1995, pp. 29–38.

[36] K. Toyama *et al.*, "Wallflower: Principles and practice of background maintenance," in *Proc. ICCV*, 1999, pp. 255–261.

[37] T. Matsuyama, T. Ohya, and H. Habe, "Background subtraction for nonstationary scenes," in *Proc. 4th Asian Conf. Computer Vision*, 2000, pp. 662–667.

[38] E. Durucan and T. Ebrahimi, "Change detection and background extraction by linear algebra," *Proc. IEEE*, vol. 89, pp. 1368–11 381, Oct. 2001.

[39] C. Thorpe, M. H. Herbert, T. Kanade, and S. A. Shafer, "Vision and navigation for the Carnegie-Mellon Navlab," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 10, pp. 362–373, Mar. 1988.

[40] J. J. Little and J. Kam, "A smart buffer for tracking using motion data," in *Proc. Computer Architecture for Machine Perception*, 1993, pp. 257–266.

[41] L. Marcenaro, F. Oberti, and G. L. Foresti, "Distributed architecture and logical-task decomposition in multimedia surveillance," *Proc. IEEE*, vol. 89, pp. 1419–1440, Oct. 2001.

[42] R. T. Collins, A. J. Lipton, H. Fujiyoshi, and T. Kanade, "Algorithms for cooperative multisensor surveillance," *Proc. IEEE*, vol. 89, pp. 1456–1477, Oct. 2001.

Takashi Matsuyama received the B. Eng., M. Eng., and D. Eng. degrees in electrical engineering from Kyoto University, Japan, in 1974, 1976, and 1980, respectively.

He is currently a Professor in the Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University. His research interests include knowledge-based image understanding, computer vision, image media processing, and artificial intelligence. He has written approximately 100 papers and books including two research monographs, *A Structural Analysis of Complex Aerial Photographs* (New York: Plenum, 1980) and *SIGMA: A Knowledge-Based Aerial Image Understanding System* (New York: Plenum, 1990). He was the leader of the five years research project on Cooperative Distributed Vision. The project was started from October 1996 under the support of the Research for the Future Program, the Japan Society for the Promotion of Science. The project members include Japanese leading computer vision researchers. He is on the editorial boards of *Computer Vision and Image Understanding* and *Pattern Recognition*.

Prof. Matsuyama is a fellow of the International Association for Pattern Recognition and a member of the Institute of Electronics, Information and Communication Engineers of Japan, the Information Processing Society of Japan, the Japanese Society for Artificial Intelligence, and the IEEE Computer Society. He won six best paper awards from Japanese and international academic societies including the Marr Prize at ICCV'95.

Norimichi Ukita received the B.Eng. and M.Eng. degrees from Okayama University, Japan, in 1996 and 1998, respectively, and the Doctor of Informatics degree from Kyoto University, Kyoto, Japan, in 2001.

He is currently a Research Assistant in the Graduate School of Information Science, Nara Institute of Science and Technology, Nara Prefecture, Japan. His current research interests include computer vision and cooperative distributed processing.