Multiple Camera Coordination in a Surveillance System

Nam T. Nguyen　Svetha Venkatesh　Geoff West　Hung H. Bui

(*Department of Computing, Curtin University of Technology, Perth, Australia*)

(E-mail: {nguyentn, svetha, geoff, buihh}@cs. curtin. edu. au})

Abstract　We present a distributed surveillance system that uses multiple cheap static cameras to track multiple people in indoor environments. The system has a set of Camera Processing Modules and a Central Module to coordinate the tracking tasks among the cameras. Since each object in the scene can be tracked by a number of cameras, the problem is how to choose the most appropriate camera for each object. This is important given the need to deal with limited resources (CPU, power etc.). We propose a novel algorithm to allocate objects to cameras using the object-to-camera distance while taking into account occlusion. The algorithm attempts to assign objects in the overlapping field of views to the nearest camera, which can see the object without occlusion. Experimental results show that the system can coordinate cameras to track people and can deal well with occlusion.

Key words　Human tracking, distributed surveillance system, occlusion, camera allocation

1　Introduction

Rapid advances in technology have made cheap sensors and especially cameras available. This has changed the goals of surveillance from building surveillance systems using only a single, powerful camera to building surveillance systems deploying many cheap cameras. Moreover, many types of monitoring require large spaces to be monitored from several viewpoints, and such problems have found no robust solutions with earlier techniques. The scenario we consider is many static cameras monitoring a large area, dividing up the area and objects among themselves. The problem is of tremendous current interest, and other teams such as VSAM[1] are working toward this goal. The ability to analyze data from such a "forest of sensors"[1] should improve existing surveillance and monitoring methods.

The problem is complex given the limited capabilities of camera on-board processing and the difficulties in coordinating many cameras. Assuming that a number of cameras can solve the same tracking tasks, the issue is how to choose the most appropriate camera for each task. This can be regarded as a coordination problem in that the goal is to maximize the reliability of the tracking system given that there is limited processing capability available for each camera. The key question we seek to answer is: given overlapping field of views (FOV) of the cameras and multiple people moving around, can we find a good assignment algorithm to track the people reliably? The algorithm should deal well with situations where a person moves from the FOV of one camera to the FOV of another camera, or when a person is occluded by others.

This paper presents a distributed surveillance system. It operates in indoor environments and uses multiple cheap static cameras to track multiple people. Unlike other tracking systems, in this system, a novel algorithm is introduced to allocate people to the cameras. With the help of this algorithm, the system can track people reliably even with the limitation of camera on-board processing.

The paper is structured as follows. The next section reviews work in the field of multiple object tracking using multiple cameras. We then introduce our distributed surveillance system and describe the object assignment algorithm in detail. Finally, the ability of the algorithm to maximize performance in the presence of occlusion is demonstrated in experimental surveillance scenarios.

2 Related background

The problem of using cameras to track people has been investigated for a long time. There have been numerous surveillance systems for detecting and monitoring people activities in indoor or outdoor environments. These systems can be classified as single camera tracking systems and multiple camera tracking systems.

Several robust single camera tracking systems have been presented in [1~5]. Intille *et al.* present a real-time system[1] that uses a single camera to track multiple non-rigid objects, which interact with each other often. They propose a nine-step procedure to match blobs with current objects in the scene. However, the accuracy of this matching procedure is not very high. Haritaoglu *et al.* described a system[4] to monitor people activities in outdoor environments. The system is able to track multiple people even with occlusion by employing a combination of shape analysis and tracking. Wren *et al.* propose a system[2] that uses a multi-class statistical model of color and shape to detect parts of a person such as head, hands and feet. The system is improved to work with stereo cameras to track 3-D movement of the human body[6]. Single camera surveillance systems are not very useful for many applications because they work in compact areas, while in many situations it is required to monitor people activities in large, complex environments.

Recently, work on tracking people with multiple cameras has emerged[7~13]. In multiple camera tracking, many problems arise. On the one hand, we have to solve the problems in single camera tracking such as object segmentation, object occlusion, and so on. On the other hand, we have to deal with the new issues that arise when there are multiple cameras in the system. These include how to identify a person when he moves between the FOV of the cameras and how to coordinate the cameras to track people.

There are two approaches used in building tracking systems that comprise multiple cameras: a centralized approach or a decentralized approach. With the centralized approach, the surveillance system needs to fuse the data at a central server[14]. This leads to the computational and communication bottlenecks at the central server when the number of cameras increases. Thus, the decentralized approach is becoming of more interest. Rao *et al.* describe a distributed tracking system[15], in which a completely decentralized algorithm for Kalman filtering is run on each node simultaneously. However, they do not consider the problem of dividing the tracking tasks among the cameras to make the system work more efficiently. Regazzoni and Marcenaro[16] present a design of general distributed multi-sensor surveillance systems. They also describe six major issues relating to distributed surveillance systems: the system architecture, the information processing, the communication, the data fusion at different levels and the performance evaluation.

Some robust multiple camera tracking systems have been developed. The VSAM project at CMU[7] developed a system using multiple pan/tilt/zoom cameras to monitor activities over a large area. New targets detected in the scene are classified using neural networks. Then the system uses multiple cameras cooperatively to track targets through the scene. Stauffer and Grimson[8] present a monitoring system, which uses multiple static cameras and works in indoor as well as outdoor environments. The system is able to classify different kinds of objects and learn variety of object activities. Orwell *et al.* present a

tracking system[13] that uses several cameras placed far from each other. The system models the color distributions of objects, then uses this information to recognize them when they appear in the FOV of other cameras.

3 Overview of the distributed tracking system

Our aim is to build a cheap surveillance system to track people moving in indoor environments. The system consists of multiple cameras, each connected to a computer on a local area network. We use static cameras in the system because of their low price. The camera FOVs overlap one another. Therefore an object may be viewed from several cameras at a time. Due to noise, objects need to be tracked by Kalman filters[17]. There are two approaches to tracking an object using Kalman filters: 1) it is tracked by all cameras that can see the object, and 2) it is tracked by only one appropriate camera. The object is robustly tracked with the first approach, but more computational resources would be required for the system because it needs to maintain more Kalman filters. Moreover, the increase of the tracking reliability sometimes is not worth the additional computational costs. Because of the limitation of camera on-board processing, we choose the second approach for our system. With this approach, the system still can track objects reliably with the help of Kalman filters, but it requires an algorithm to assign each object to a suitable camera.

The distributed tracking system is shown in Figure 1. With each camera, we have a corresponding Camera Processing Module (CPM) running on the computer that the camera is connected to. The system also has a Central Module (CM) that maintains a set of current objects in the entire scene and coordinates the tracking tasks between the CPMs. Each CPM then needs to process the video stream to track the objects that are assigned to the CPM by the CM.
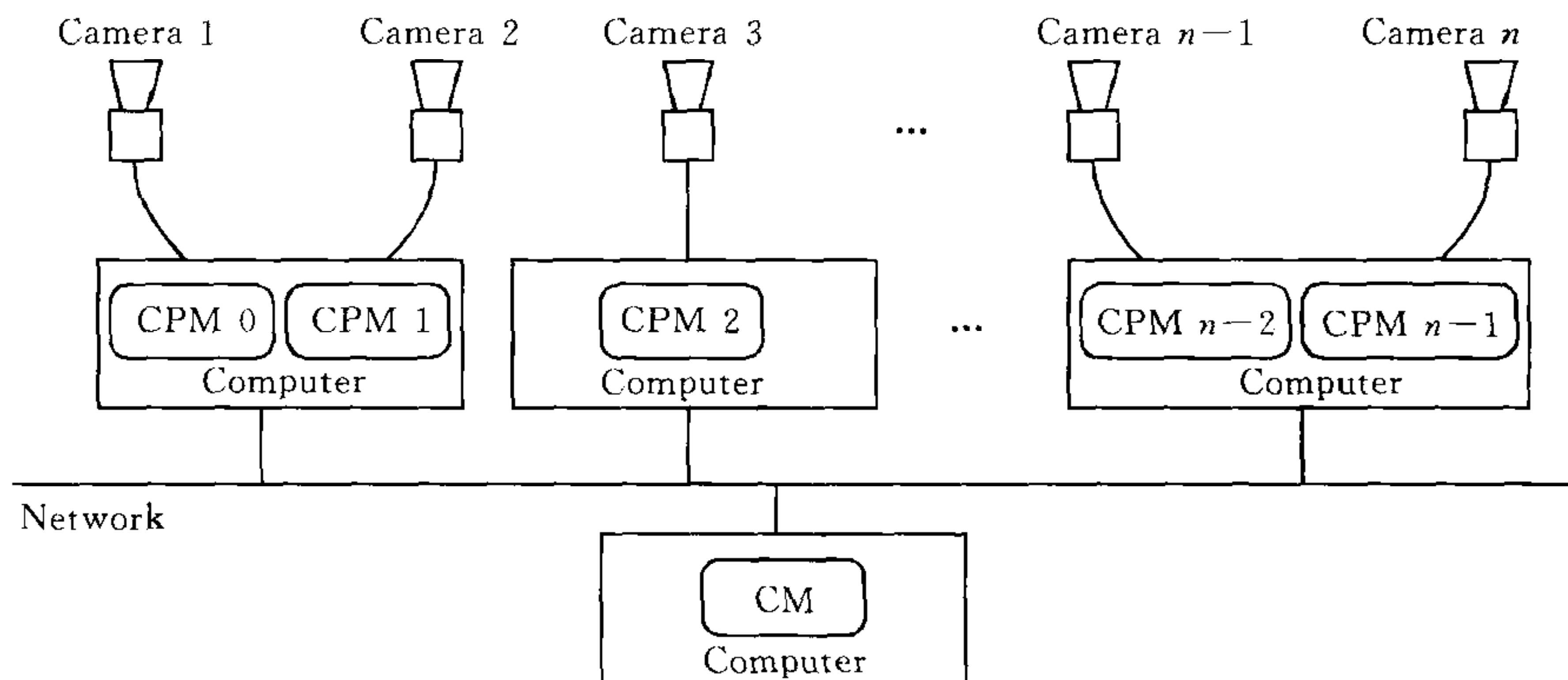

Fig. 1   A system of distributed network of cameras

Figure 2 shows the data processing and communication that takes place in a CPM and the CM. The left block shows the tasks performed at the CPM. The image captured by the camera is processed by the blob segmentation step to extract the motion blobs (the step A in Figure 2). The CPM also initializes a set of Kalman filters to track objects that are assigned to the camera (step B in Figure 2). These Kalman filters are updated using the blob information. The right block shows the tasks performed at the CM. The CM maintains a set of current objects in the scene and assigns each of them to a suitable camera.
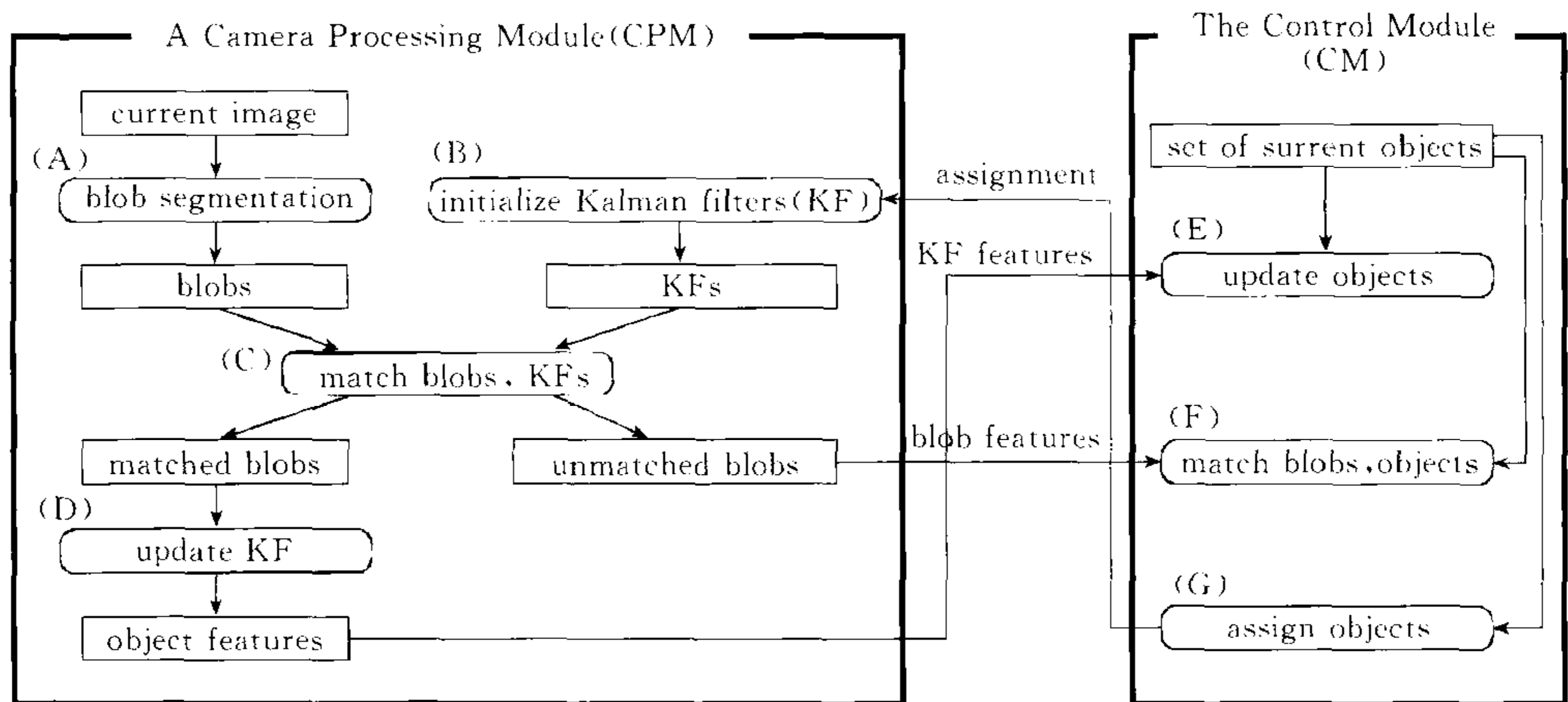
A Camera Processing Module(CPM)

current image
(A)
blob segmentation

(B)
initialize Kalman filters(KF)

blobs

KFs

(C) match blobs . KFs

matched blobs

unmatched blobs

(D)
update KF

object features

The Control Module (CM)

set of current objects

assignment

KF features

(E)
update objects

(F)
match blobs ,objects

blob features

(G)
assign objects

Fig. 2   Processing at a CPM and at the CM

4   The camera processing module

4. 1   Blob segmentation

The blob segmentation step (step A in Figure 2) extracts motion blobs from the image sequences using background subtraction. To do this, we need a background model. There are many methods to build background models[2,9,18,8,19,20,4]. Some methods model the intensity of each pixel of the background with a normal distribution[2,9]. Others use a mixture of Gaussian distributions to model pixel values[18,8]. However, these methods are computationally expensive. Our system runs in indoor environments, so for simplicity and fast computation, the background image is first computed as the average of several images, and then updated periodically to adapt to the changes in the scene.

The foreground pixels are found by comparing the current image with the background. We detect the boundaries of these foreground pixels by a chain-code algorithm[21]. The blobs of motion are computed based on these boundaries. Due to the camera noise and the similarity of the object and background color, an object may be broken into several blobs. Therefore, two blobs that are near each other are merged into a single blob. We do this by merging recursively until all resulting blobs are well separated. After merging, blobs that are too small are considered as noise and discarded. Figure 3 shows an example of the results of the blob segmentation step at a CPM.
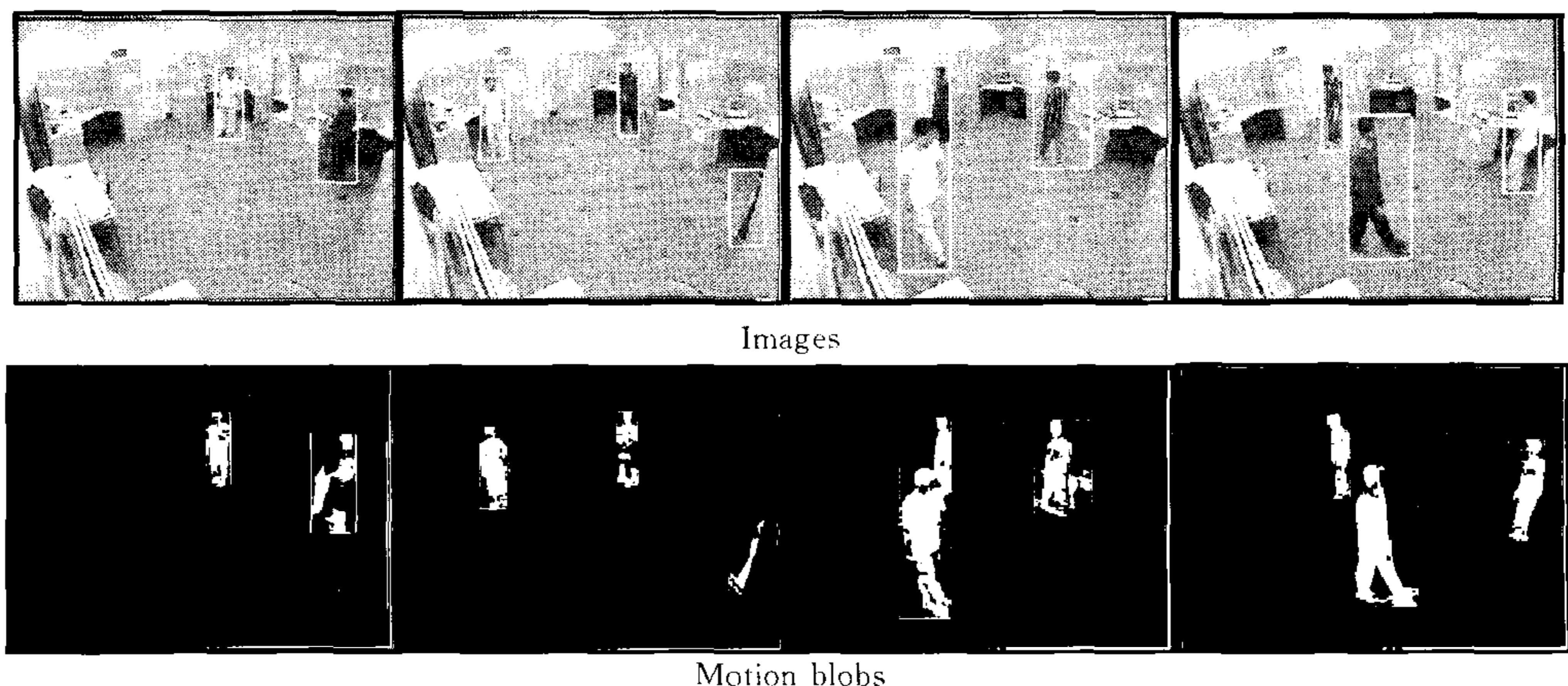
Images

Motion blobs

Fig. 3   Examples of blob segmentation at a CPM

4.2 The Kalman filters

A Kalman filter[17] is initialized every time the CPM is assigned to track a specific object. The state vector of a Kalman filter K is:

$$[k_x, k_y, k_{vx}, k_{vy}, k_{tr}, k_{tg}, k_{tb}, k_{br}, k_{bg}, k_{bb}]^T \qquad (1)$$

where (k_x, k_y) is the estimate of the object's feet on the image, taken as the center of the bottom edge of the object bounding box, (k_{vx}, k_{vy}) is the velocity of the object's feet, (k_{tr}, k_{tg}, k_{tb}) and (k_{br}, k_{bg}, k_{bb}) are the estimates of the average red, green and blue color components of the top half and bottom half of the object. Normally, the average of the top half color of an object characterizes the color of the shirt of the person, and the average of the bottom half color characterizes the color of the trousers of the person. All these variables are assumed to have Gaussian distributions.

4.3 Matching blobs to the Kalman filters

Blobs extracted from the blob segmentation step are the observations of the Kalman filters. We need to find which blob corresponds to the observation of which Kalman filter. This task is performed by the blob-to-Kalman-filter matching step (step C in Figure 2).

We use the position and average color properties to find the match. The probabilistic distance between a blob B and a Kalman filter state K is defined as:

$$d_1(B,K) = P(b_x, b_y \mid k_x, k_y) \qquad (2)$$

where (b_x, b_y) is the center of the bottom edge of blob B. The blobs that are too big or too small are not considered in the matching step. We also enforce three hard constraints to exclude invalid matches:

1) $d_1(B,K) \leqslant thres_{dist}$

2) $P(b_{tr}, b_{tg}, b_{tb} \mid k_{tr}, k_{tg}, k_{tb}) \leqslant thres_{color}$

3) $P(b_{br}, b_{bg}, b_{bb} \mid k_{br}, k_{bg}, k_{bb}) \leqslant thres_{color}$

where $thres_{dist}$ and $thres_{color}$ are the distance threshold and color threshold respectively, (b_{tr}, b_{tg}, b_{tb}) and (b_{br}, b_{bg}, b_{bb}) are the color properties of the blob B similar to the color properties that have been defined for Kalman filter K.

We find a set of matched (blob, Kalman filter state) pairs, so that the total match probabilistic distance is minimal. This is an instance of the bipartite matching problem[22]. In our system, we use the non-iterative greedy algorithm[23] to solve this problem. It works as follows

1) Choose a valid pair (B, K) for which the probabilistic distance $d_1(K,B)$ is the minimum. Output the match (B, K).

2) Remove B from the list of blobs, remove K from the list of Kalman filter states. Return to Step 1 if there are still valid (blob, Kalman filter state) pairs.

Because of occlusion, a Kalman filter may have no observation. In that case, it continues to estimate the object properties for some time period. After this period, if the Kalman filter still has not received the observation, the corresponding object is considered having the status of a "lost" object. The Central Module (CM) will try to switch the task of tracking this object to another camera.

5 The central module

At the CM, we maintain a database of current objects in the scene. For each camera-object pair, the database keeps information on whether the camera is currently tracking the object with the Kalman filter, or is not tracking it but can see the object in its FOV. Additionally, the camera might not see the object clearly if it is occluded or not in the camera's FOV. Figure 4 shows an example of the object database at the CM at a particular time.

	Camera 1	Camera 2	Camera 3
Object 1	tracked	not seen	seen,but not tracked
Object 2	seen,but not tracked	tracked	not seen
...	...	...	...

Fig. 4   The status of the database at a particular time

In order to construct this database, the CM performs the following steps. If an object is being tracked by a camera (decided by step G in Figure 2), it is assigned the label of "tracked". The properties of this object are updated by getting information from the corresponding Kalman filter (step E in Figure 2). For any remaining blobs detected but not matched in step C, they are matched with other existing objects in the database using real-world position and color properties (step F in Figure 2). The purpose of this matching is to let the system know which objects each camera can see clearly. The distance between a blob B and an object O is defined as:

$$d_2(B,O) = \sqrt{(b_{rx} - o_{rx})^2 + (b_{ry} - o_{ry})^2} \qquad (3)$$

where (b_{rx}, b_{ry}) and (o_{rx}, o_{ry}) are the real-world positions of blob B and object O respectively. They are calculated from the image positions of blob B and object O by the calibration step. A match between blob B and object O must satisfy three constraints:

1) $d_2(B,O) \leqslant thres_{wdist}$

2) $\max\{|b_{tr} - o_{tr}|, |b_{tg} - o_{tg}|, |b_{tb} - o_{tb}|\} \leqslant thres_{wcolor}$

3) $\max\{|b_{br} - o_{br}|, |b_{bg} - o_{bg}|, |b_{bb} - o_{bb}|\} \leqslant thres_{wcolor}$

where (o_{tr}, o_{tg}, o_{tb}) and (o_{br}, o_{bg}, o_{bb}) are the average colors of the top half and bottom half of object O, $thres_{wdist}$ and $thres_{wcolor}$ are the distance and color thresholds. We have the same bipartite matching problem as in Section 4. 3. The same algorithm is used to find the blob-to-object correspondences. When a matched (blob, object) pair is found, the object is considered be "seen, but not tracked" by the camera that detects the blobs.

6   Assigning objects to the cameras

In this part, we propose an algorithm to assign objects to the cameras using the object-to-camera distance while taking into account the object occlusion. We also introduce a function to measure the performance of the tracking system. This function is then used to evaluate our assignment algorithm.

6. 1   Quality of service of a tracking system

In many cases, we need to compare the operation of a tracking system working with the different parameters and algorithms. To do that, we need a function representing quantitatively the performance of tracking systems. We term this function the Quality of Service (QOS) function. The QOS function is defined based on the sizes of the objects in the image and the object occlusion status as follows:

$$Q_{all} = QOS(C_0, \cdots, C_{n-1}, O_0, \cdots, O_{m-1}) = \sum_{i=0}^{n-1} \sum_{O \in \Omega_i} QOS(C_i, O) \qquad (4)$$

where Q_{all} is the QOS of the whole system, $QOS(C_i, O)$ is the QOS which camera C_i gives to object O, Ω_i is the set of objects currently tracked by camera C_i.

The QOS that a camera gives to an object equals the estimated size of the object obtained from the Kalman filter. Thus, the closer the object is to the camera the higher the QOS becomes. Should the object be occluded, the Kalman filtering estimate is then used for several frames, after that the object is considered as lost and the QOS drops to zero.

6. 2   Object assignment algorithm

At each new time step, objects are re-assigned to the cameras. For a reliable tracking system, we need an object assignment algorithm to obtain the highest QOS for the sys-

tem. In Equation (4), the QOS of the system is maximized when each object is tracked by the camera that can view it with the largest size. Usually, in the case of no occlusion, the nearest camera will give the largest view of the object. If there is occlusion, an object needs to be tracked by one of the cameras that can see it clearly. Therefore, we have the assignment algorithm as follows: with each object, among the cameras that see this object clearly, choose the nearest one to track the object. In the case that no camera sees the object clearly, the object continues to be assigned to the current camera.

The system gets the set of cameras that can see an object clearly from the database at the CM (see Section 5). The distance between a camera and an object can be calculated using their positions in real-world space. Therefore, the assignment algorithm described above can be easily implemented in the system.

To see how the algorithm works, assume that an object O is being tracked by a camera C. When object O moves out of the FOV of camera C, camera C can no longer see object O clearly. Then, the algorithm will switch the tracking of object O to one of the cameras that see it clearly. Thus, the system can avoid losing the object. In the case that object O is occluded, the system will attempt to switch O to the camera that views it without occlusion. By this switching step, the system can take the advantage of multiple cameras to deal with occlusion.

7   Experimental results

We have implemented the tracking system in a real environment. The goal of the system is to find the trajectories of people in the scene. We demonstrate the capability of the system to track people reliably in a large environment.

7.1   The environment

The environment consists of a corridor, the Staff room and Vision lab. We have mounted six static cameras in the environment, of which three (cameras 0, 1 and 5) are in the Vision lab, two (cameras 2 and 3) are in the corridor and the last one (camera 4) is in the Staff room. Figure 5 shows the positions of these cameras. The camera FOVs overlap
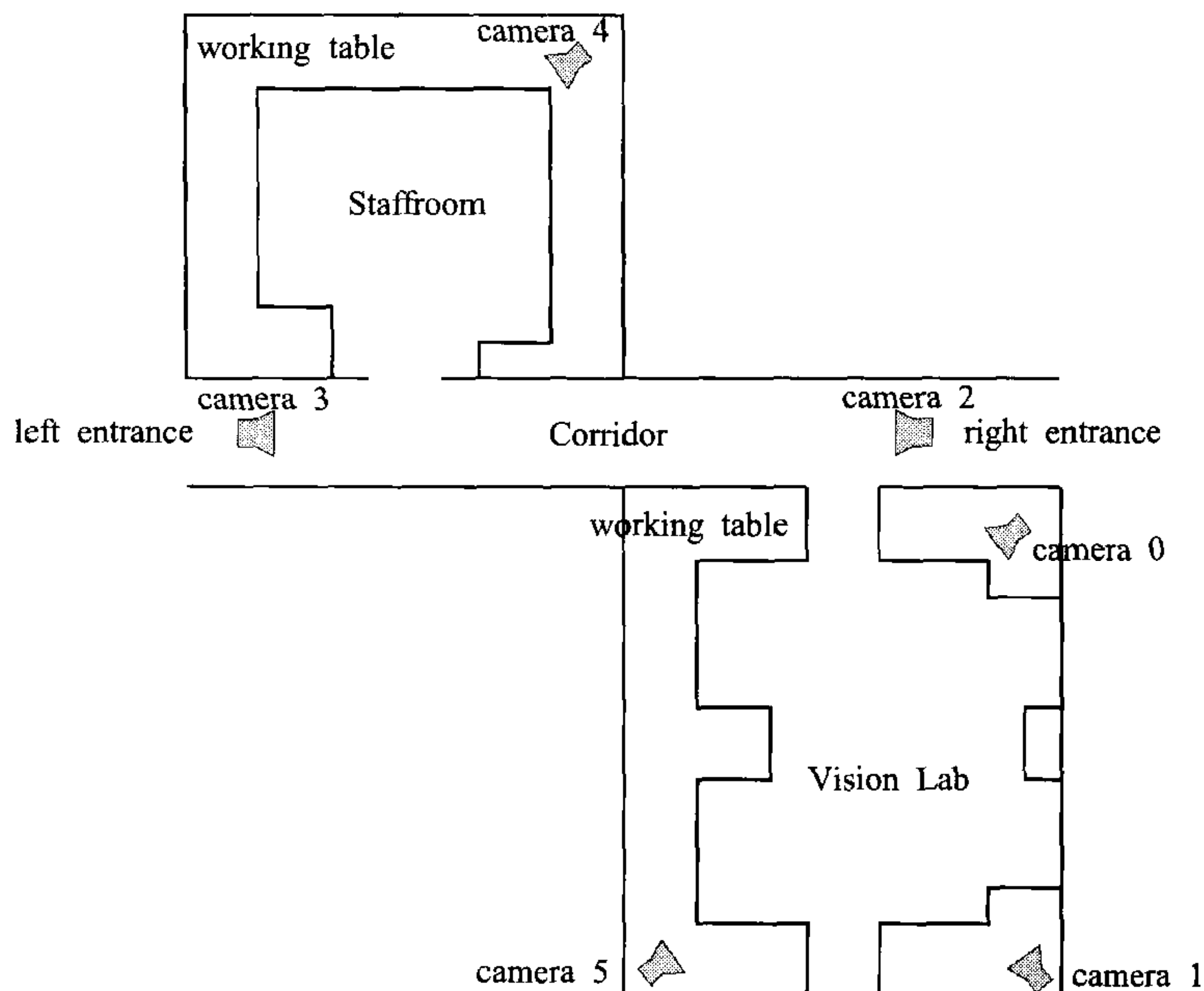
Fig. 5   The positions of cameras $C_0 \sim C_5$

each other and covers most of the ground plane in the scene. Figures 6(a)~(f) show the scene viewed from the six cameras. All cameras are calibrated to get the correspondence between points on the floor (ground plane) and points in the image. People enter to or exit from the scene via the left/right entrances of the corridor.

We consider several scenarios having some people walking in the environment. Each scenario is recorded in six video files. Each video file contains the sequence of frames of each camera view. The frame rate of the recording is 15 frame/sec. These files are taken as the input for the tracking system. We can re-run the experiments to compare the tracking reliability of the system for different switching algorithms.

(a) camera 0   (b) camera 1   (c) camera 2

(d) camera 3   (e) camera 4   (f) camera 5
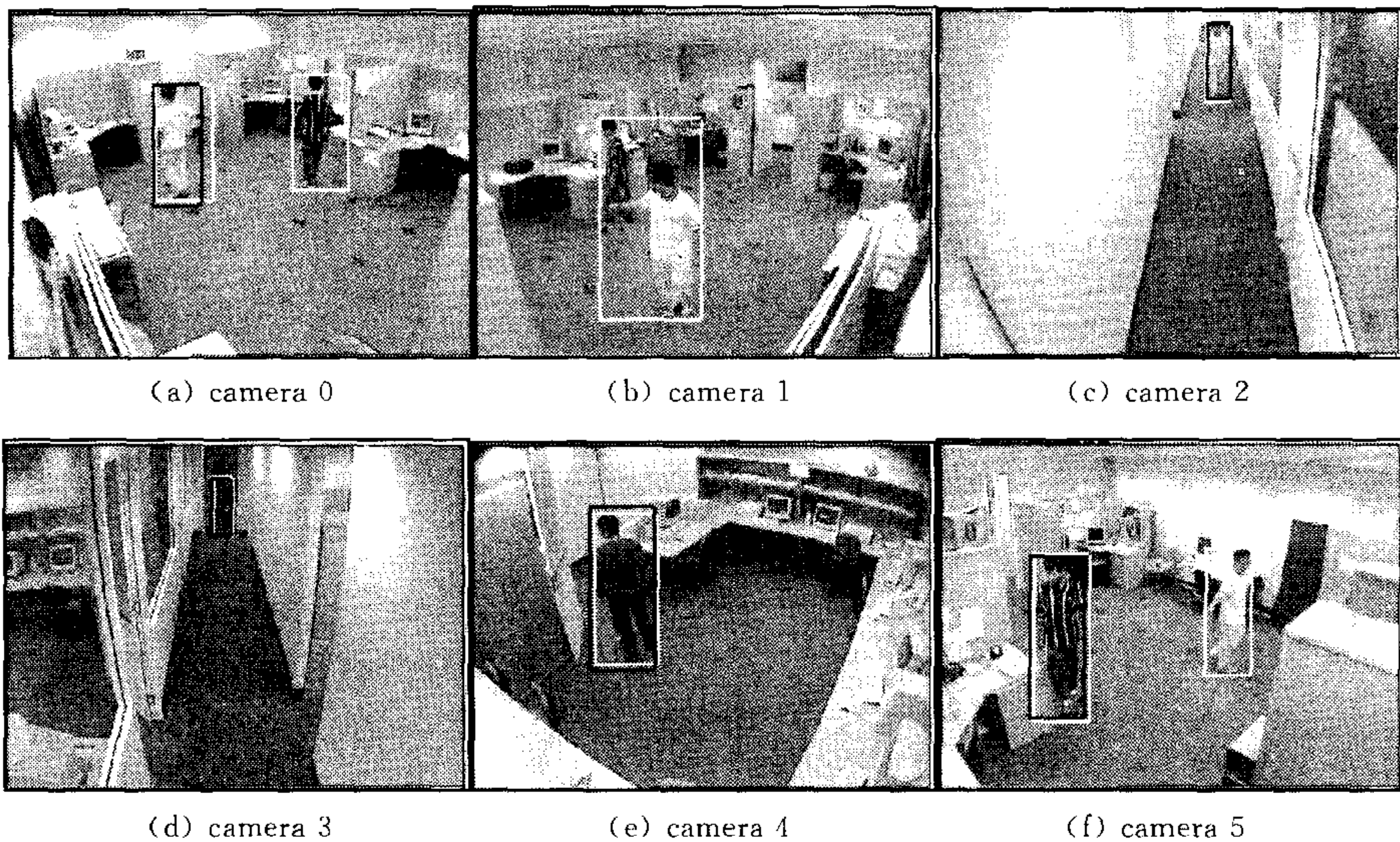
Fig. 6   The scene viewed from the six cameras

7. 2   Extracting the people trajectories

We get the people trajectories from the recorded scenarios. Figures 7(a)~(c) show the people trajectories returned from the system in scenario 1. In this scenario, the system has detected the appearance of three people and monitored their trajectories. Person 1 enters the scene via the right entrance, walks round the Vision lab and takes the right entrance to exit (see Figure 7(a)). Person 2 enters the scene via the right entrance and takes the left entrance to exit (see Figure 7(b)). Person 3 enters the scene via the left entrance and takes the right entrance to exit (Figure 7(c)). Figures 8(a)~(c) show the people trajectories returned from the tracking system in another scenario. In this scenario, the system has also detected the appearance of three people and extracted their trajectories accordingly. There is reasonably seamless integration across cameras to form one trajectory.

We examine how the system uses multiple cameras to track person 2 in scenario 1 (see the trajectory in Figure 7(b)). At each time, person 2 is tracked by only one camera with a Kalman filter. Figure 9 shows which camera tracks person 2 (in dark shirt) with Kalman filter at which period of time. In each image in the figure, the white boxes are the blob bounding boxes, the black boxes are the estimate bounding boxes by the Kalman filters. It can be seen that person 2 is tracked by the camera that has the best view of him all the time.
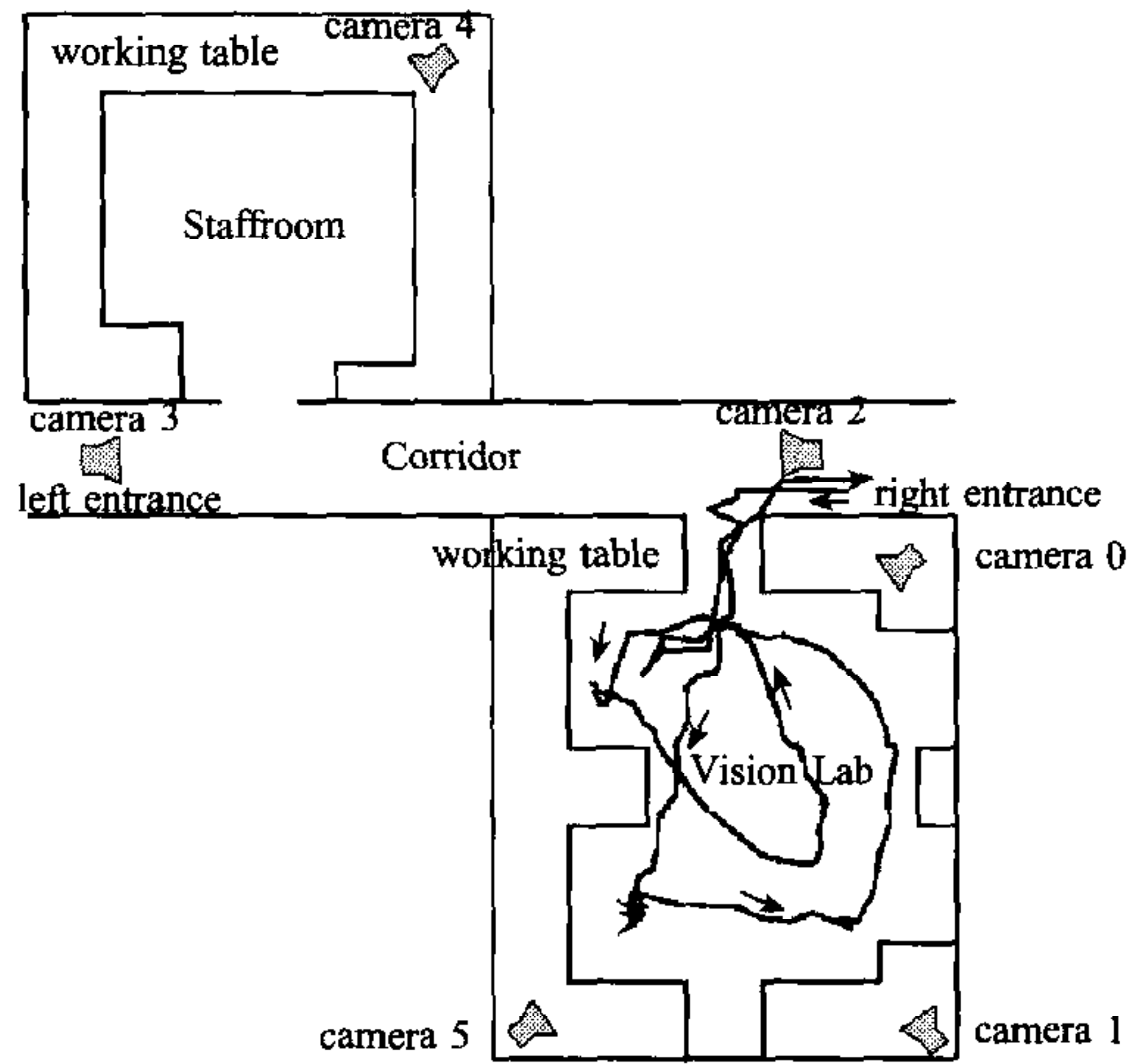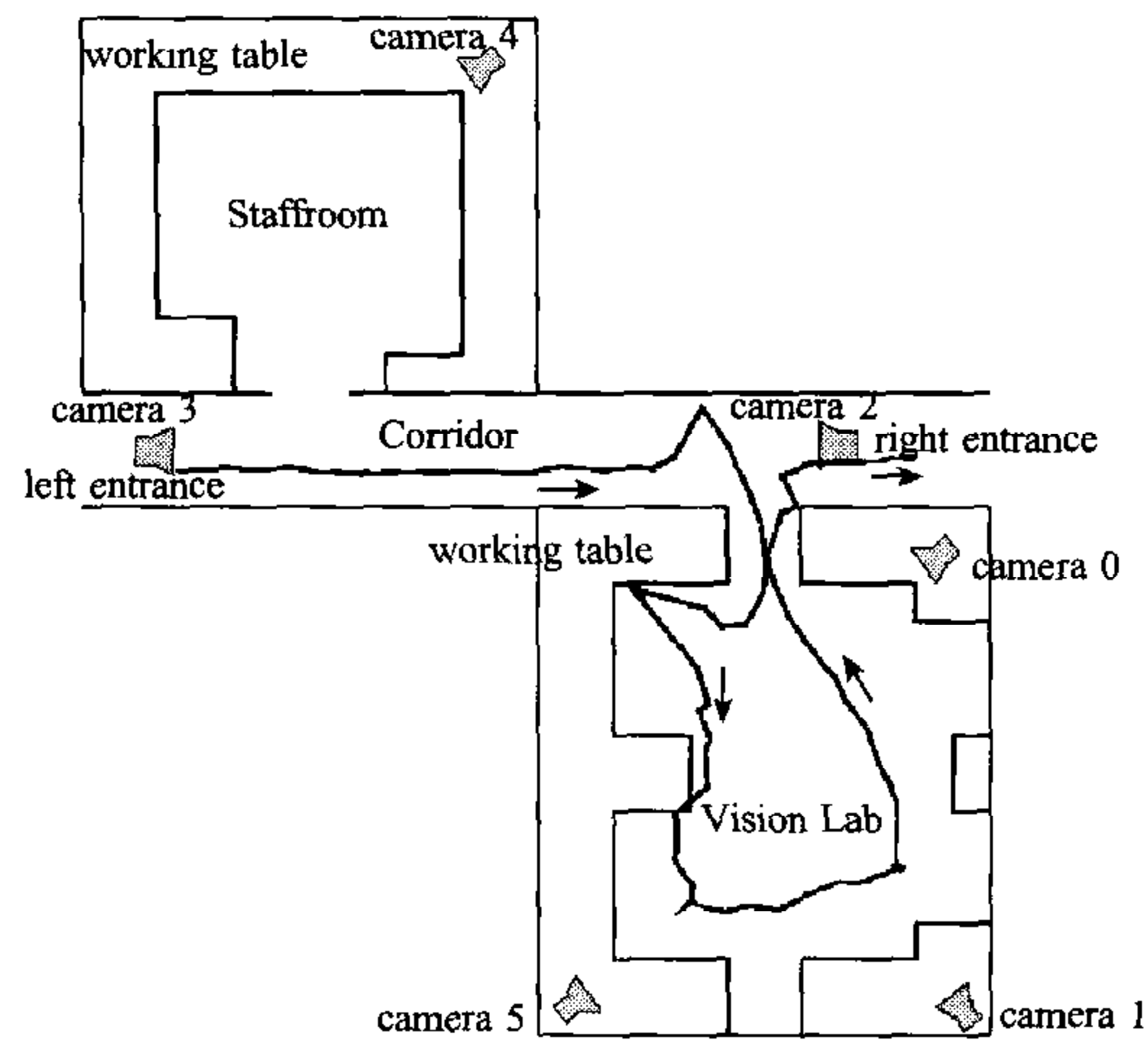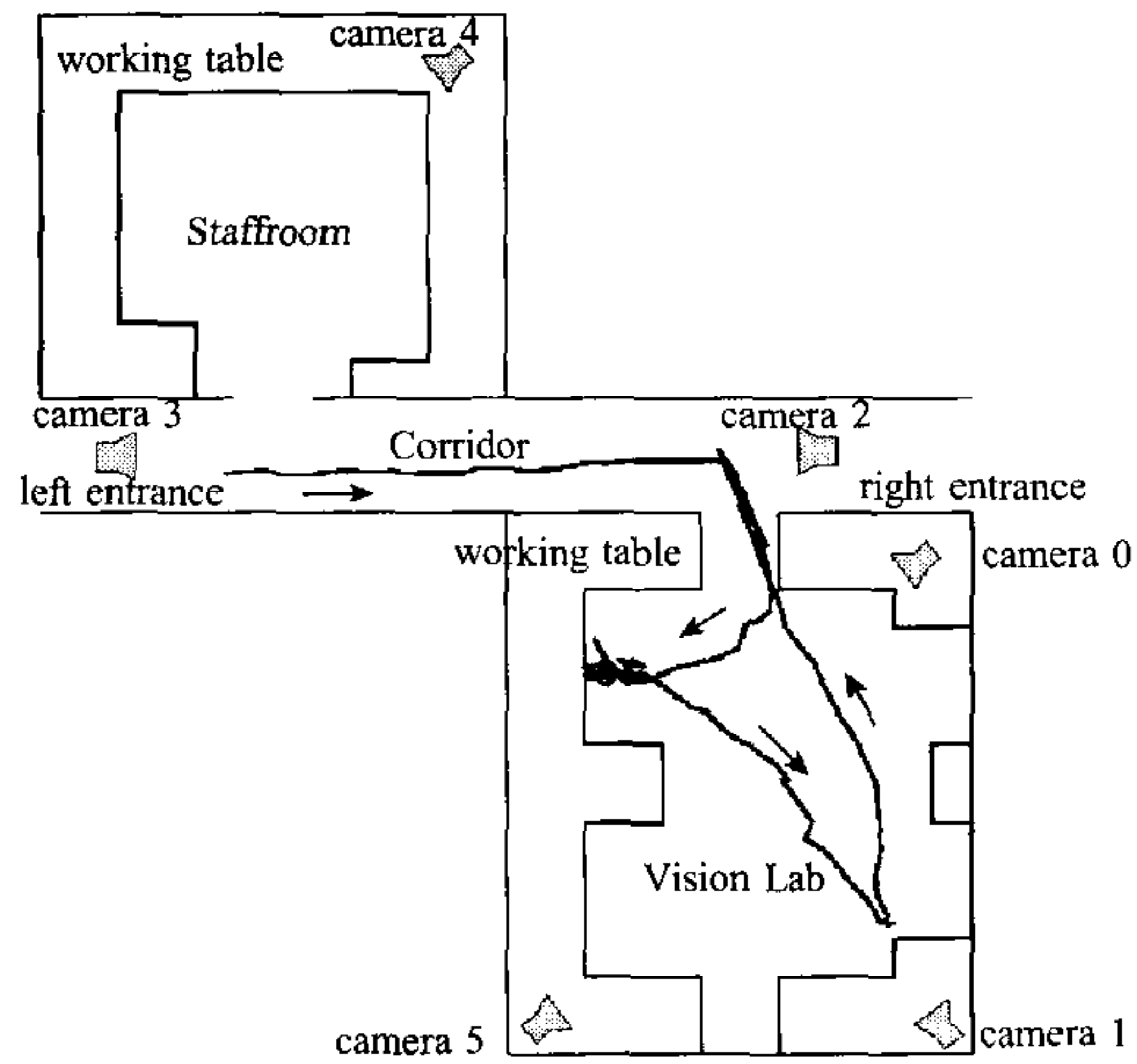
(a) person 1
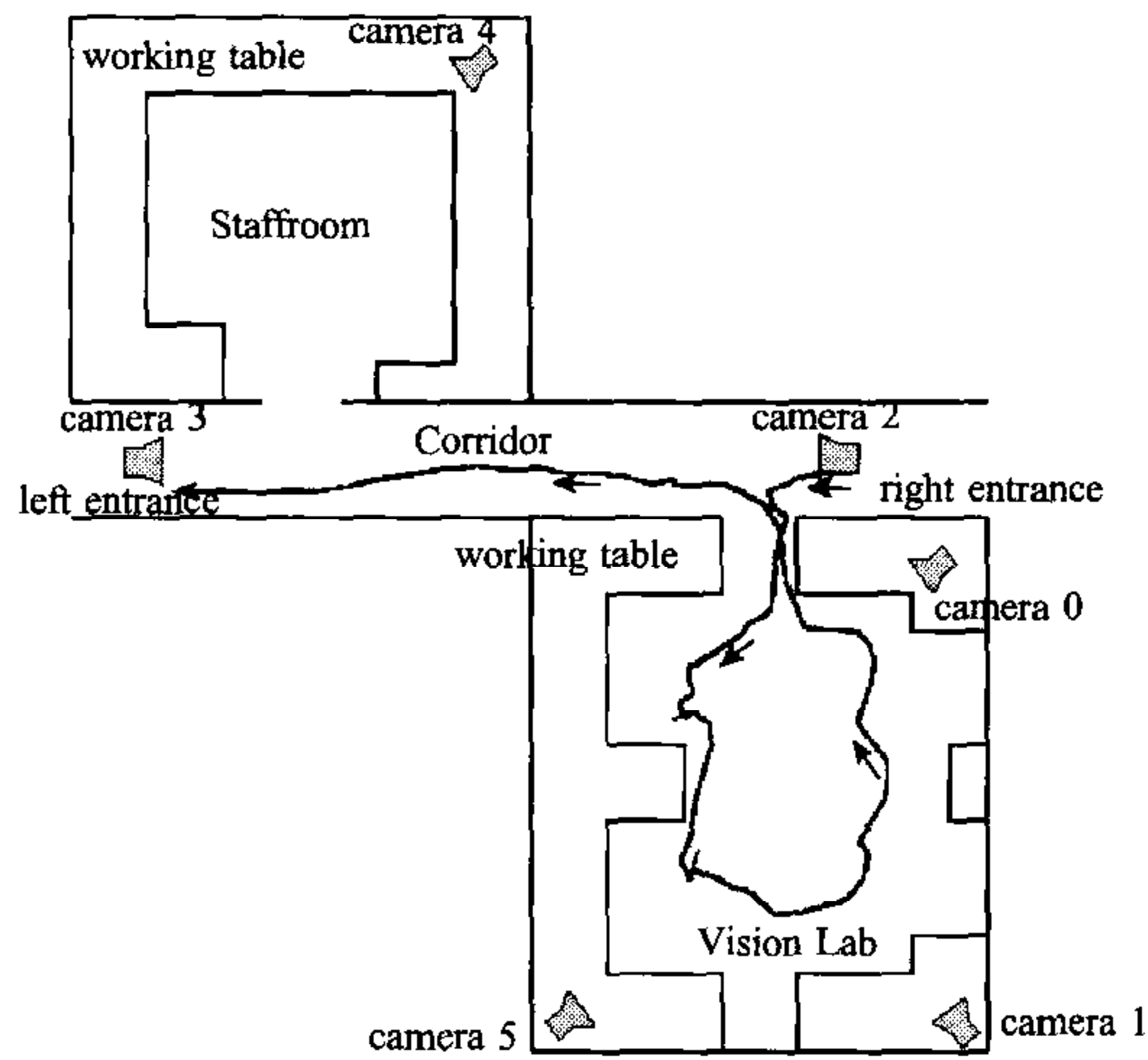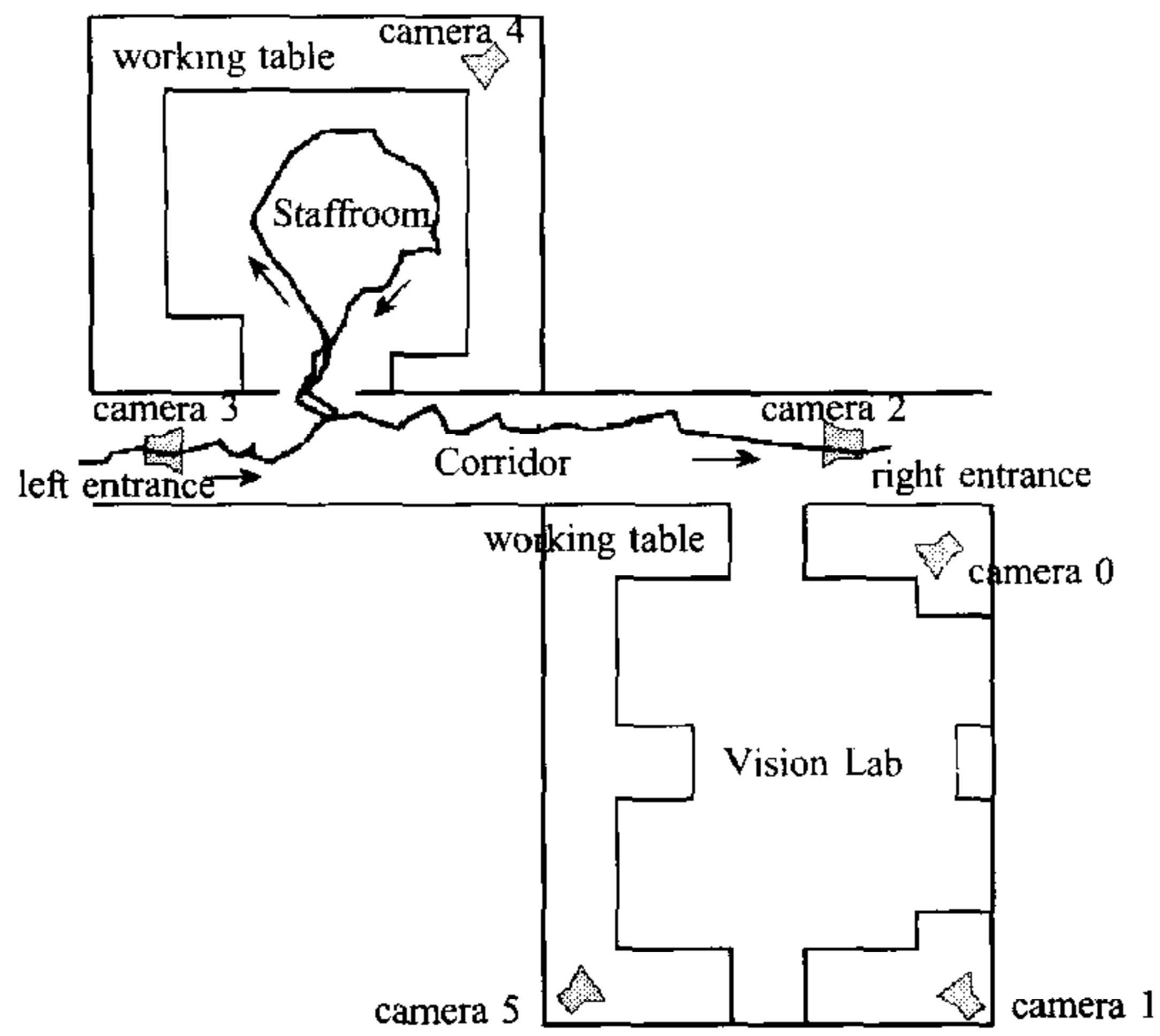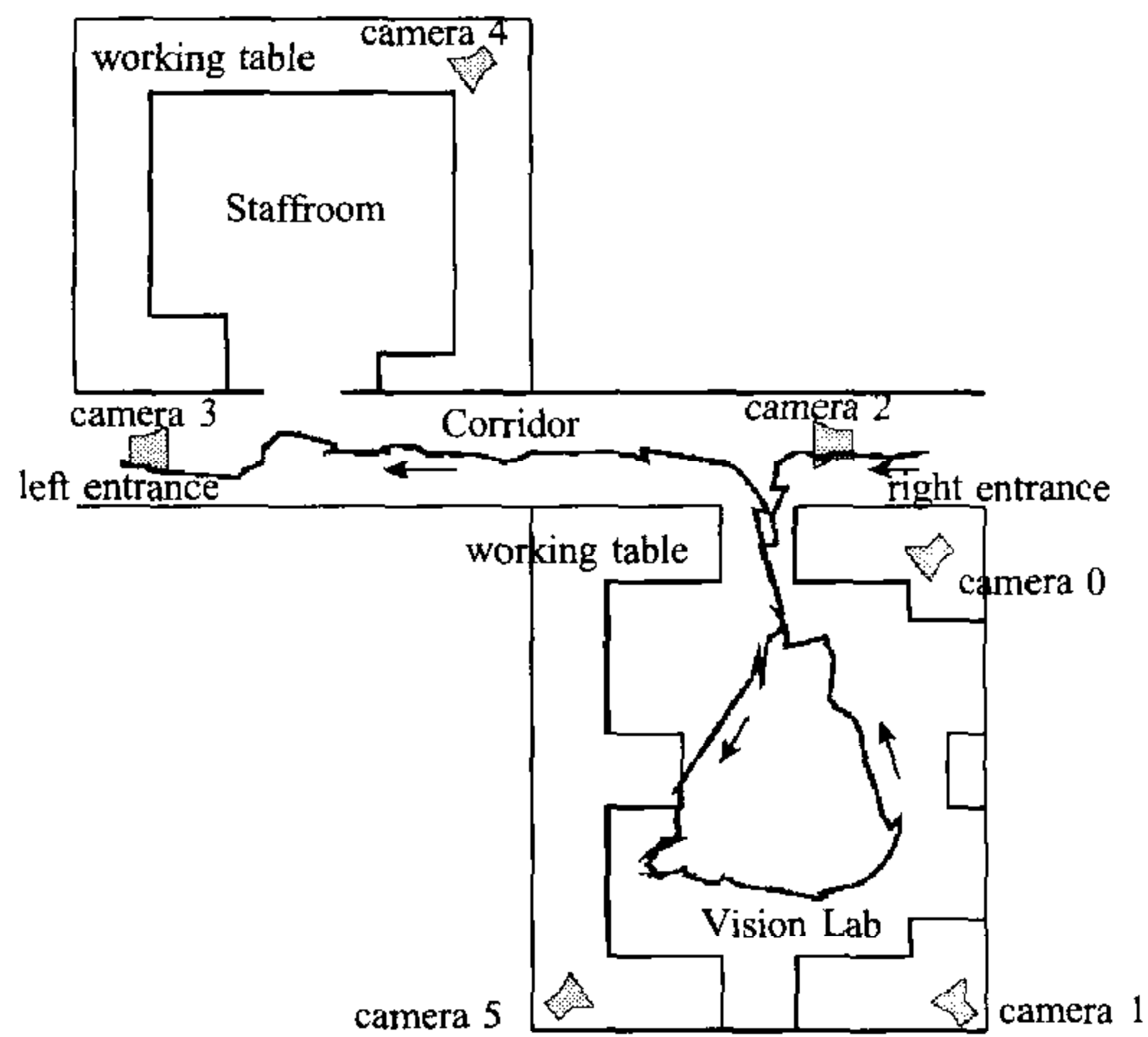

(b) person 2

(c) person 3

Fig. 7   The trajectories of three people in scenario 1

(a) person 1

(b) person 2

(c) person 3

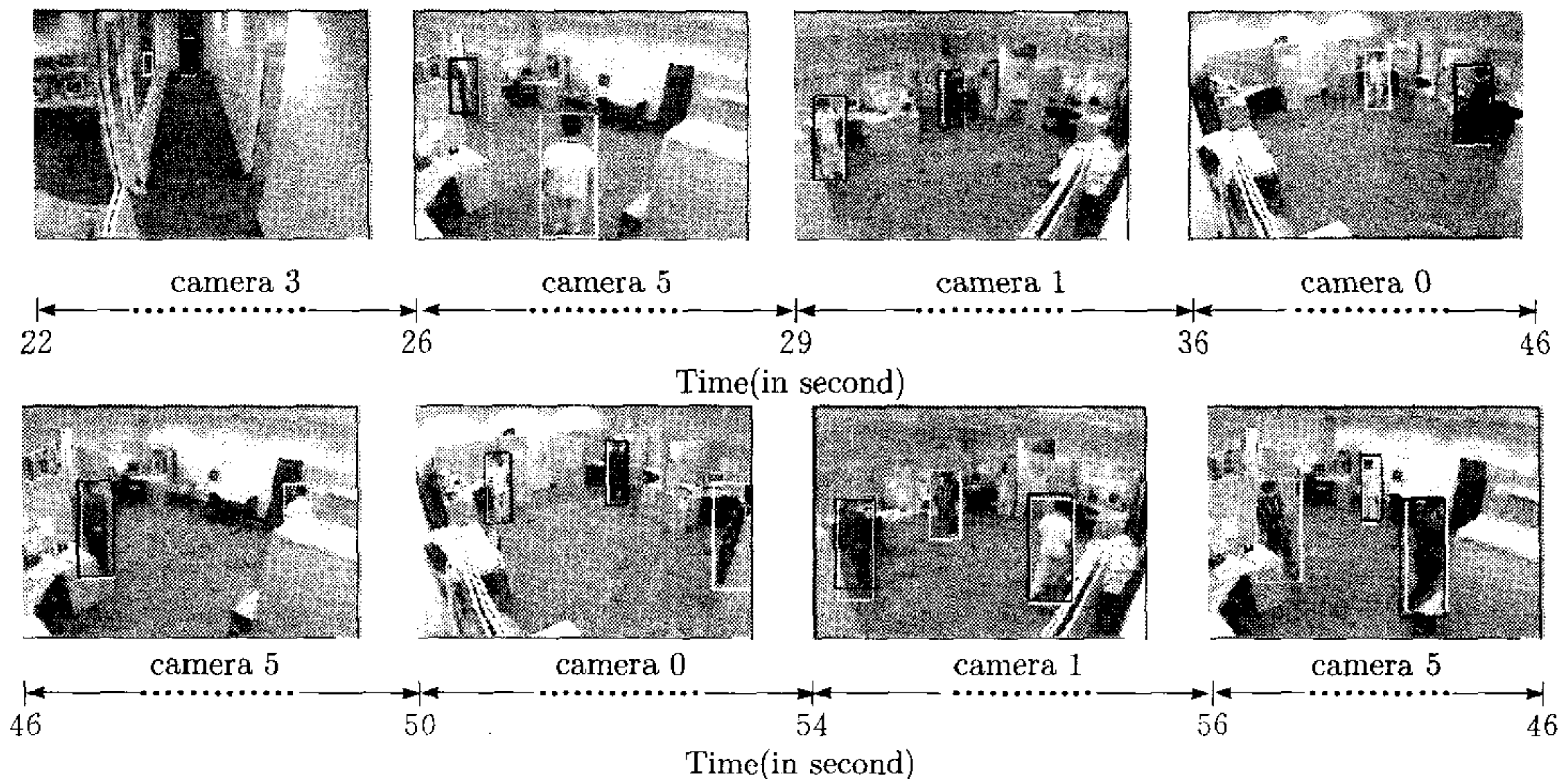Fig. 8   The trajectories of three people in scenario 2

Fig. 9   The periods of time that each camera tracks person 2 in scenario 1

7.3   Dealing with people's occlusion

The system uses the advantage of multiple cameras to deal with occlusion. We examine what happens in the Vision lab, where we have mounted three cameras (cameras 0, 1 and 5). We consider scenario 1 at 55, 57 and 60 seconds, when there are 3 people in the lab.

Figures 10(a), (b) and (c) show the views from cameras 0, 1 and 5 respectively at 55 seconds. The black boxes in Figure 10(b) show that person 2 (in dark shirt) is currently tracked by camera 1. At 57 seconds, the person 2 overlaps person 3 (in striped shirt) from the view of camera 1 (see Figure 11(b)), therefore camera 1 cannot track person 2 properly. The system switches the tracking task of person 2 to camera 5 to avoid occlusion (see Figure 11(c)).

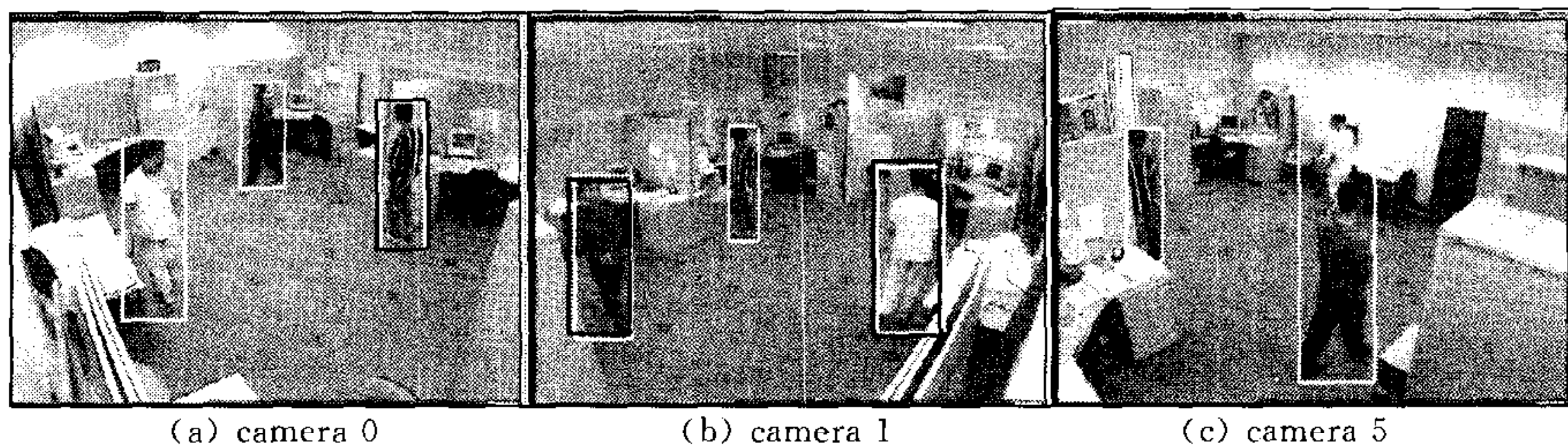(a) camera 0   (b) camera 1   (c) camera 5

Fig. 10   Views from cameras 0, 1 and 5 at 55 seconds in scenario 1

Let us see another example of switching to deal with occlusion. At 57 seconds, person 3 is tracked by camera 5 (see Figure 11(c)). Then, occlusion happens between person 3 and person 1 (in white shirt) in the FOV of camera 5 (see Figure 12(c)). Because of this, the task of tracking person 3 is switched to camera 1 which can view him clearly (see Figure 12(b)).

(a) camera 0   (b) camera 1   (c) camera 5
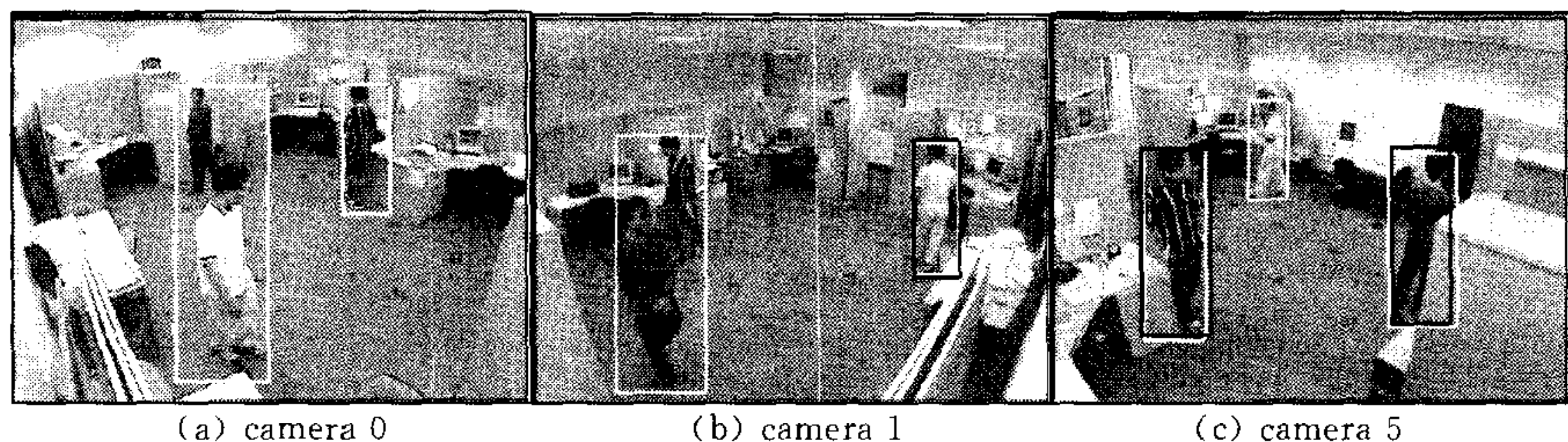
Fig. 11   Views from cameras 0, 1 and 5 at 57 seconds in scenario 1

(a) camera 0   (b) camera 1   (c) camera 5

Fig. 12   Views from cameras 0, 1 and 5 at 60 seconds in scenario 1

7. 4   Examining the quality of service

To illustrate how the object switching algorithm described in Section 6 helps the system track people reliably, we introduce another switching algorithm and make a comparison using the QOS function. The former algorithm assigns a person to a camera based on the person-to-camera distance and the occlusion status. We call this algorithm Distance and Occlusion Switching Algorithm (DOSA). The latter switching algorithm is described as follows. If a camera is tracking a person, it continues to tracks this person unless the person is occluded or out of the FOV of the camera. In the case of occlusion, the system will assign this person to another camera. We call this algorithm Occlusion Switching Algorithm (OSA).

We replace DOSA, which is currently implemented in the tracking system, with OSA and re-run the system with scenarios 1 and 2. Figure 13 shows the comparison of the QOS assigned to all people in DOSA and OSA in scenarios 1 and 2. As expected, most of the time, DOSA maintains a higher QOS than OSA. To explain this, we see how the system tracks person 1 in scenario 2 at 17 and 18. 5 seconds. At 17 seconds, person 1 is tracked by
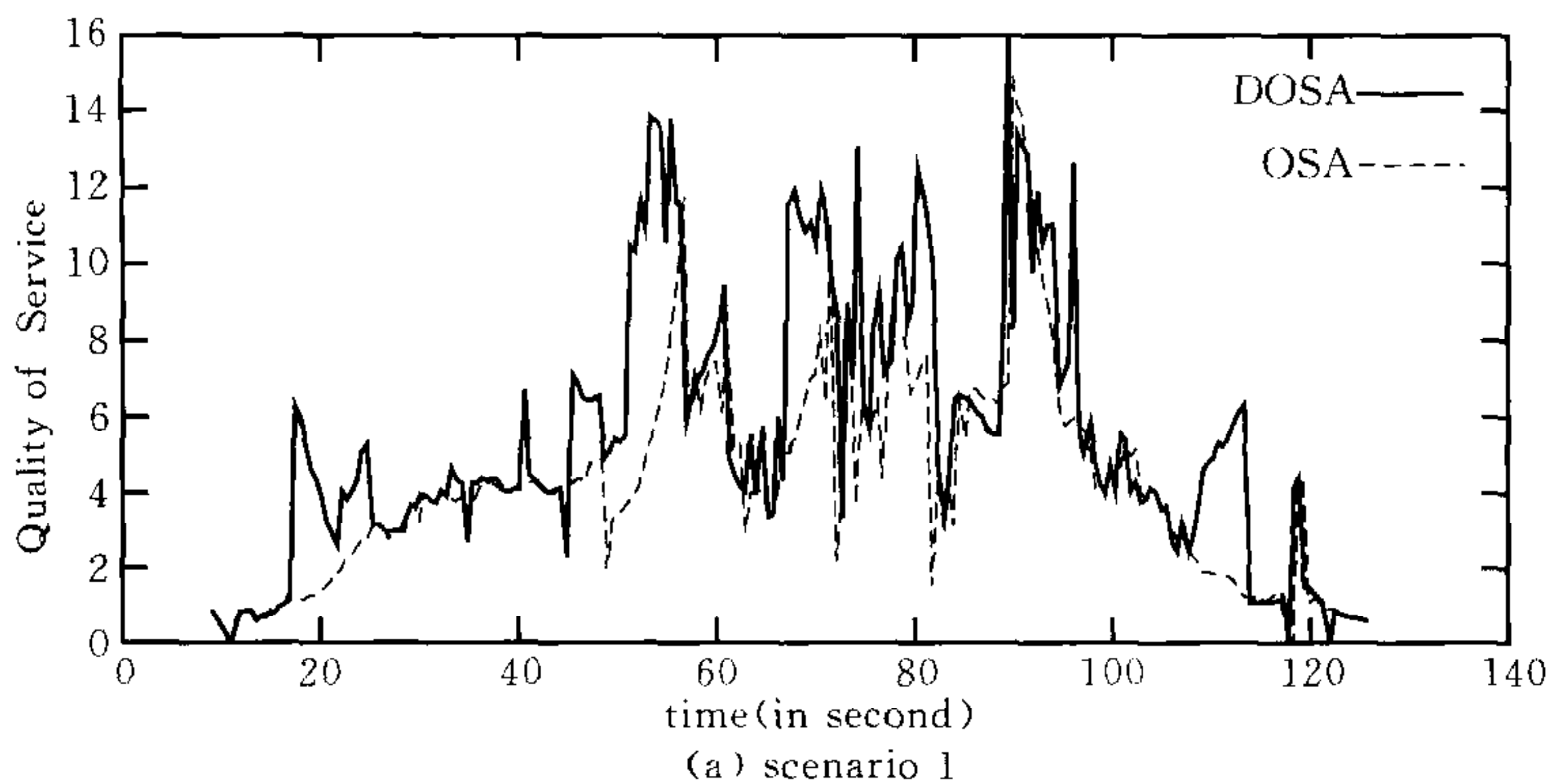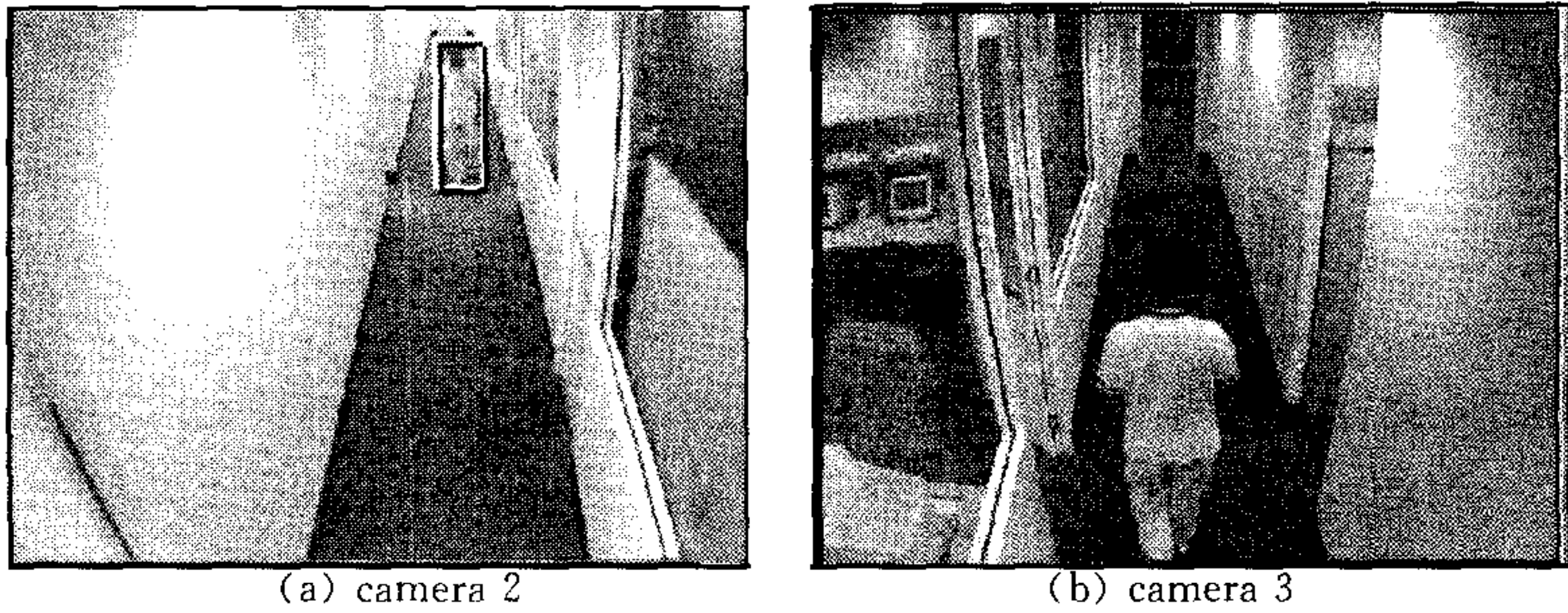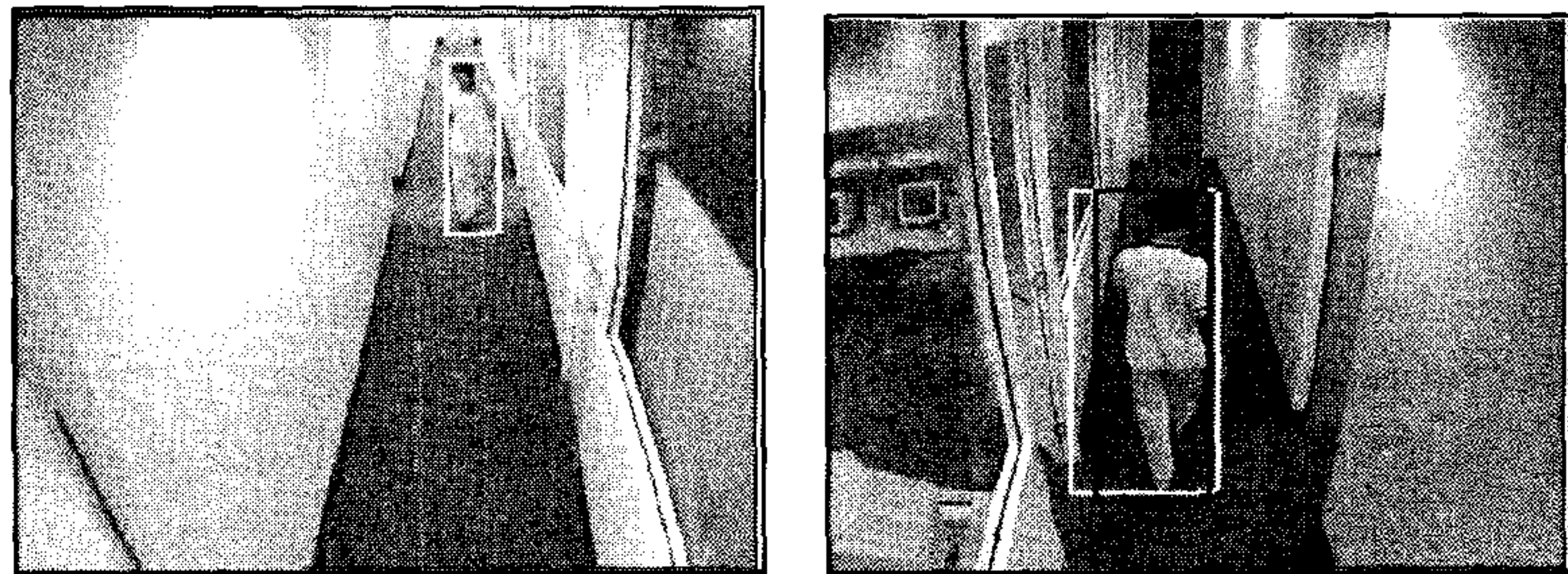

(a) scenario 1
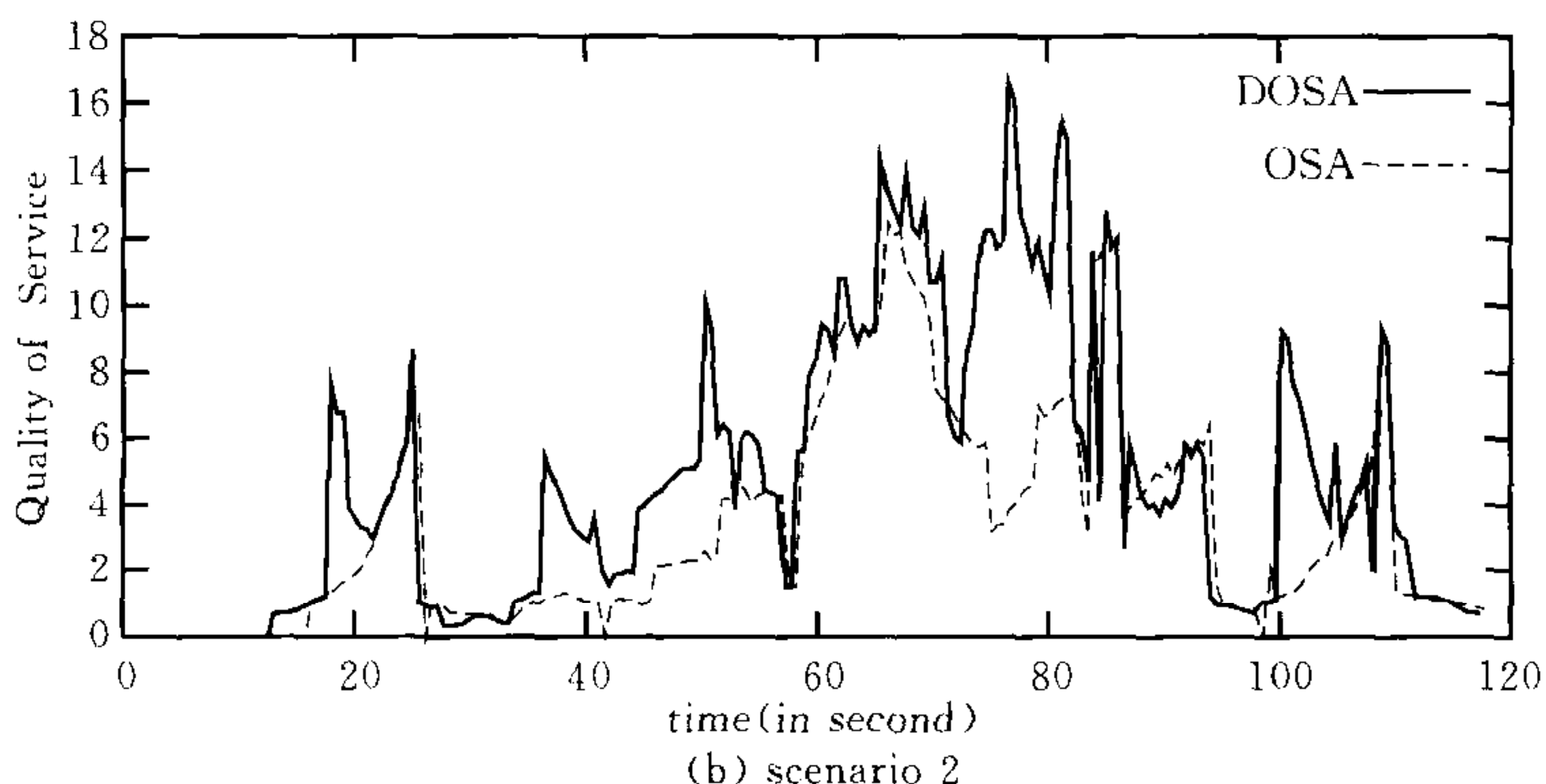
(b) scenario 2

Fig. 13   Comparing the QOS assigned to all people in DOSA and OSA in scenarios 1 and 2

camera 2 in both algorithms (see Figure 14(a)). At 18.5 seconds, in DOSA, the task of tracking person 1 is switched to camera 3 because he is nearer camera 3 than camera 2 (see Figures 15(a) and (b)). But in OSA, person 1 is still tracked by camera 2 while camera 3 can offer a better view (see Figures 16(a) and (b)). This leads to the QOS assigned to person 1 in DOSA being higher than in OSA at 18.5 seconds (see Figure 17).
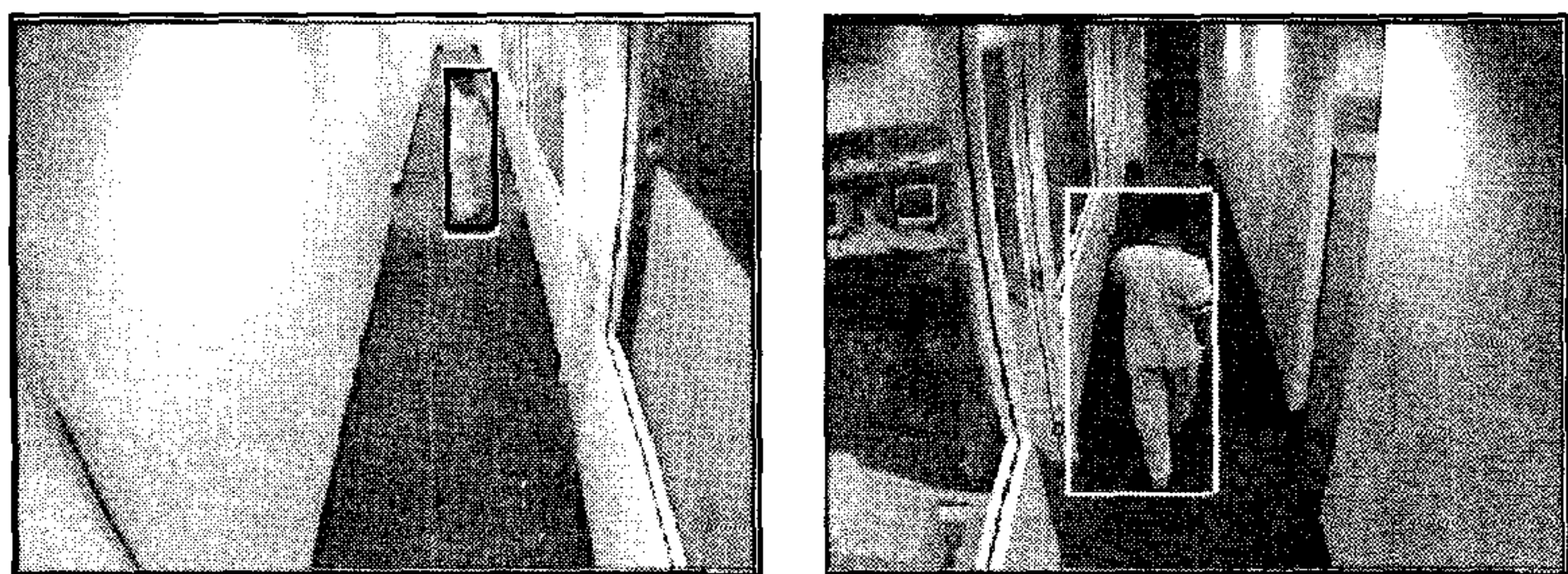
The above results show that switching based on object-to-camera distance and occlusion status helps the system track people better.

(a) camera 2   (b) camera 3

Fig. 14   Person 1 at 17 seconds in scenario 2

(a) camera 2   (b) camera 3

Figure 15: Person 1 is switched to camera 3 at 18.5 seconds in DOSA

(a) camera 2   (b) camera 3

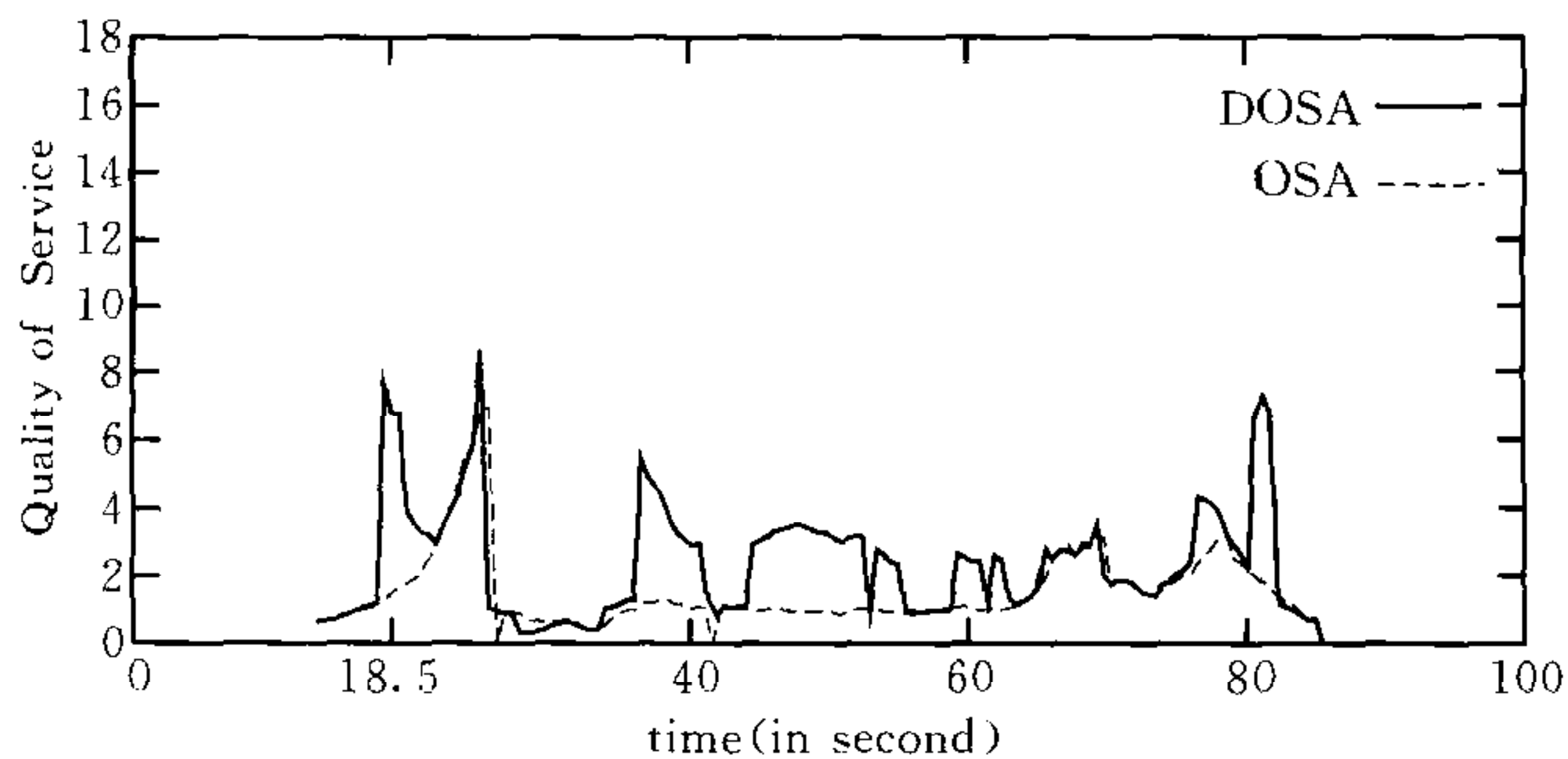Figure 16: Person 1 is still tracked by camera 2 at 18.5 seconds in OSA

Fig. 17   Comparing QOS assigned to person 1 in DOSA and OSA in scenario 2

8 Conclusion

In this paper, we have presented a distributed surveillance system to track multiple people using multiple cameras. In the system, we introduced an algorithm to coordinate the cameras to track people more reliably. The algorithm allocates objects to cameras using the object-to-camera distances while taking into account occlusion. We also have proposed a QOS function to measure the performance of the algorithm. Our experimental results show the robustness of our algorithm in dealing with the occlusion and its performance to obtain high QOS at all times.

In future research, we will consider the issue of resource allocation when dealing with a large number of objects. When many cameras and objects are involved in the system, the processing at all cameras should be balanced to avoid camera computational overloading. To measure the tracking reliability of the system, the QOS function needs to take into account other parameters such as the camera frame rate, the number of objects currently tracked by each camera, etc. This should allow the system to scale up to work in more complex spatial environments such as a complete building.

References

1   Intille S S, Davis J W, Bobick A F. Real-time closed-world tracking. In: IEEE Conference on Computer Vision and Pattern Recognition, San Juan: IEEE Press, 1997. 697~703

2   Wren C, Azarbayejani A, Darrell T, Pentland A. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1997,19(7):780~7875

3   Boult T. Frame-rate multibody tracking for surveillance. In: Proceedings of DARPA Image Understanding Workshop, DARPA/ISO, Morgan Kaufmann Publishers, Inc. , 1998. 305~313

4   Haritaoglu I, Harwood D, Davis L S. w^4 : Real time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000,22(8):809~830

5   Rehg J, Loughlin M, Waters K. Vision for a smart Kiosk. In: IEEE Conference on Computer Vision and Pattern Recognition, San Juan: IEEE Press, 1997. 690~696

6   Azabrayjani A, Wren C, Pentland A. Real-time 3d tracking of the human body. In: Proceedings of IMAGE'COM, Bordeaux,1996

7   Collins R, Lipton A, Fujiyoshi H, Kanade T. Algorithms for cooperative multi-sensor surveillance. *Proceedings of the IEEE*, 2001,89(10): 1456~1477

8   Stauffer C, Grimson E. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 2000,22(8):747~757

9   Olson T, Brill F. Moving object detection and event recognition algorithms for smart cameras. In: Proceedings of DARPA Image Understanding Workshop, 1997. 159~175

10   Shafer A, Krumm J, Brumitt B, Meyers B, Czerwinski M, Robbins D. The new EasyLiving project at Microsoft. In: Proceedings of DARPA/NIST Smart Spaces Workshop, Gailhersburg, 1998

11   Cai Q, Aggrwal J K. Tracking human motion using multiple cameras. In: Proceedings of the 5th International Conference on Pattern Recognition, Vienna, 1996. 68~72

12   Stillman S, Tanawongsuwan R, Essa I. A system for tracking and recognizing multiple people with multiple cameras. In: Proceedings of the 2nd International Conference on Audio and Video based biometric person authentication, Washington, 1999

13   Orwell J, Remagnino P, Jones G A. Multi-camera colour tracking. In: IEEE International Workshop on Visual Surveillance, 1999. 14~21

14   Bar-Shalom Y, Fortmann T E. Tracking and Data Association. New York: Academic Press, 1988

15   Rao B, Durrant-Whyte H, Sheen J. A fully decentralized multi-sensor system for tracking and surveillance. *International Journal of Robotics Research*, 1993,12(1):20~44

16   Regazzoni C S, Marcenaro L. Object detection and tracking in distributed surveillance systems using multiple cameras. In:Proceedings of NATO Advanced Studies Institute on Multisensor Data Fusion, Perthshire, 2000

17   Maybeck P S. Stochastic Models, Estimation and Control. New Work: Academic Press, 1979

18   Friedman N, Russell S. Image segmentation in video sequence: A probabilistic approach. In: Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence, 1997. 175~181

19   Koller D, Weber J, Huang T, Malik J, Ogasawara G, Rao B, Russell S. Towards robust automatic traffic scene analysis in real-time. In: Proceedings of the International Conference on Pattern Recognition, Israel, 1994. 126~131

20   Ridder C, Munkelt O, Kirchner H. Adaptive background estimation and foreground detection using Kalman-filtering. In: Proceedings of International Conference on Recent Advances in Mechatronics, Istanbul,1995. 193~199

21   Freeman H. Computer processing of line-drawing images. *Computing Surveys*, 1974, 6(1): 57~97

22   Papadimitriou C H, Steiglitz H. Combinatorial Optimazation: Algorithms and Complexity. New Jersey: Prentice Hall, 1982

23 　Rangarajan, Shah M. Establishing motion correspondence. *Computer Vision*, *Graphics and Image Processing*: *Image Understanding*, 1991,24(6):56~73

Nam T. Nguyen Doctor candidate in computer science at the Department of Computing, Curtin University of Technology. Received his bachelor degree in computer science, Department of Mathematics, Mechanics and Informatics, Vietnam National University in 1994, master degree in computer science, Department of Mathematics, Mechanics and Informatics, Vietnam National University, Hanoi, Vietnam, in 1997. His research interests include machine vision with emphasis on people tracking and human behaviour understanding.

Svetha Venkatesh Professor at the School of Computing at Crutin University of Technology, Perth, Western Australia. She is currently the co-director for the Center of Excellence in Intelligent Operations Management. Her research interests include large scale pattern recognition, image understanding and applications of computer vision to image and video indexing and retrieval.

Geoff West Received his bachelor degree in 1978 and his P. D. degree in 1982 from the City University, London. From 1982 to 1988 he was employed as a research fellow and from 1988 as a lecturer at the City University in the School of Engineering investigating techniques for automatic inspection. Since 1990 he has been on the staff of the School of Computing at Curtin University of Technology in Perth, Western Australia where he is currently Professor of Computer Science. Professor West is a senior member of the IEEE and a member of the IEE(UK) and is a Chartered Engineer. He is on the editorial board of the journal Image and Vision Computing. His current research interests include 3-D object recognition, feature extraction, surveillance, real-time systems, automatic visual inspection, data mining, telemedicine and related applications.

Hung H. Bui Received the Ph. D. degree in 1998 from the Curtin University of Technology, Perth, Australia. The research interests include artificial intelligence, probabilistic plan recognition, uncertainty, graphical models,distributed AI, multi-agent systems, negotiation,machine learning.

监控系统中的多摄像机协同

Nam T. Nguyen 　 Svetha Venkatesh 　 Geoff West 　 Hung H. Bui

(*School of Computing*, *Curtin University of Technology*, *Perth*, 澳大利亚)

(E-mail:{nguyentn, svetha, geoff, buihh}@cs. curtin. edu. au})

摘　要 描述了一个用于室内场合对多个目标进行跟踪的分布式监控系统. 该系统由多个廉价的固定镜头的摄像机构成,具有多个摄像机处理模块和一个中央模块用于协调摄像机间的跟踪任务. 由于每个运动目标有可能被多个摄像机同时跟踪,因此如何选择最合适的摄像机对某一目标跟踪,特别是在系统资源紧张时,成为一个问题. 提出的新算法能根据目标与摄像机之间的距离并考虑到遮挡的情况,把目标分配给相应的摄像机,因此在遮挡出现时,系统能把遮挡的目标分配给能看见目标并距离最近的那个摄像机. 实验表明该系统能协调好多个摄像机进行目标跟踪,并处理好遮挡问题.

关键词 人体跟踪,分布式监控系统,遮挡,摄像机分配

中图分类号 TP391. 41