Multimedia Systems (2006) 12:269–283
DOI 10.1007/s00530-006-0059-4

REGULAR PAPER

Surveillance camera scheduling: a virtual vision approach

Faisal Z. Qureshi · Demetri Terzopoulos

Published online: 8 November 2006

Abstract We present a surveillance system, comprising wide field-of-view (FOV) passive cameras and pan/tilt/zoom (PTZ) active cameras, which automatically captures high-resolution videos of pedestrians as they move through a designated area. A wide-FOV static camera can track multiple pedestrians, while any PTZ active camera can capture high-quality videos of one pedestrian at a time. We formulate the multi-camera control strategy as an online scheduling problem and propose a solution that combines the information gathered by the wide-FOV cameras with weighted round-robin scheduling to guide the available PTZ cameras, such that each pedestrian is observed by at least one PTZ camera while in the designated area. A centerpiece of our work is the development and testing of experimental surveillance systems within a visually and behaviorally realistic virtual environment simulator. The simulator is valuable as our research would be more or less infeasible in the real world given the impediments to deploying and experimenting with appropriately complex camera sensor networks in large public spaces. In particular, we demonstrate our surveillance system in a virtual train station environment populated by autonomous, lifelike virtual pedestrians, wherein easily reconfigurable virtual cameras generate synthetic video feeds. The video streams emulate those generated by real surveillance cameras monitoring richly populated public spaces.

A preliminary version of this paper appeared as [1].

F. Z. Qureshi (✉) · D. Terzopoulos
Department of Computer Science, University of Toronto, Toronto, ON M5S 3G4, Canada
e-mail: faisal@cs.toronto.edu

D. Terzopoulos
Computer Science Department, University of California, Los Angeles, CA 90095, USA
e-mail: dt@cs.toronto.edu

1 Introduction

We regard the design of an autonomous visual sensor network as a problem in resource allocation and scheduling, where the sensors are treated as resources necessary to complete the required sensing tasks. Imagine a situation where the camera network must acquire high-resolution videos of every pedestrian that passes through a region of interest. The captured video should be amenable to further biometric analysis; e.g., by gait, gesture, or facial recognition routines. Passive cameras alone cannot satisfy this task. Additional, active cameras, also known as pan/tilt/zoom (PTZ) cameras, are needed to capture high-quality videos of pedestrians. As there will often be more pedestrians in the scene than the number of available cameras, the PTZ cameras must intelligently allocate their time among the different pedestrians. A resource management strategy can enable the cameras to decide autonomously how best to allocate their time to observing the various pedestrians in the scene. The dynamic nature of the sensing task further complicates the decision making process; e.g., the amount of time a pedestrian spends in the designated area can vary dramatically among different pedestrians, or an attempted video recording by a PTZ camera might fail due to occlusion.

1.1 The virtual vision paradigm

Beyond the legal obstacles to monitoring people in public spaces for experimental purposes, the cost of deploying and repeatedly reconfiguring a large-scale camera

network in the real world can easily be prohibitive for computer vision researchers. As was argued in [2], however, rapidly evolving computer graphics and virtual reality technologies present viable alternatives to the real world for developing computer vision systems. Legal impediments and cost considerations aside, the use of a virtual environment can also offer greater flexibility during the system design and evaluation process. Terzopoulos [3] proposed a virtual vision approach to designing surveillance systems using a virtual train station environment populated by fully autonomous, lifelike pedestrians that perform various activities (Fig. 1) [4]. Within this environment, virtual cameras generate synthetic video feeds. The video streams emulate those generated by real surveillance cameras, and low-level image processing mimics the performance characteristics of a state-of-the-art surveillance video system. Recently, we have further developed the virtual vision approach to surveillance in sensor networks [5].

1.2 The virtual sensor network

Within the virtual vision paradigm, we propose a sensor network comprising wide field-of-view (FOV) passive cameras and PTZ active cameras to automatically capture and label high-quality video for every pedestrian that passes through a designated region. The network described herein, a special instance of the sensor network architecture proposed in [5], is capable of performing common visual surveillance tasks using local decision making at each camera node, as well as internode communication, without relying on camera calibration, a detailed world model, or a central controller.

Unlike our earlier work [5], we assume here that the wide-FOV static cameras are calibrated,[1] which enables the network to estimate the 3D locations of pedestrians through triangulation. However, we do not require the PTZ active cameras to be calibrated. Rather, during a learning phase, the PTZ cameras learn a coarse map between the 3D locations and the gaze-direction by observing a single pedestrian in the scene. A precise map is unnecessary since each PTZ camera is an autonomous agent that can invoke a search behavior to find a pedestrian using only coarse hints about the pedestrian's 3D position. The network uses a weighted round-robin strategy to assign PTZ cameras to surveil the various pedestrians. A new observation request is inserted into the task queue for every pedestrian that is sensed. Initially, each observation request is assigned the same priority; however, the decision making process uses domain-specific heuristics, such as the distance of the pedestrian from a camera or the heading of the pedestrian, to continuously evaluate the priorities of the observation requests. The PTZ cameras handle each task in priority sequence. The surveillance system issues a warning when an observation request cannot be met.

[1] This assumption is justifiable given the existence of several static camera calibration schemes [6,7].

1.3 The virtual world simulator

Our visual sensor network is deployed and tested within the virtual train station simulator that was developed in [4]. The simulator incorporates a large-scale environmental model (of the original Pennsylvania Station in New York City) with a sophisticated pedestrian animation system that combines behavioral, perceptual, and cognitive human simulation algorithms. The simulator can efficiently synthesize well over 1,000 self-animating pedestrians performing a rich variety of activities in the large-scale indoor urban environment. Like real humans, the synthetic pedestrians are fully autonomous. They perceive the virtual environment around them, analyze environmental situations, make decisions and behave naturally within the virtual train station. They can enter the station, avoiding collisions when proceeding though portals and congested areas, queue in lines as necessary, purchase train tickets at the ticket booths in the main waiting room, sit on benches when they are tired, purchase food/drinks from vending machines when they are hungry/thirsty, etc., and eventually proceed downstairs in the concourse area to the train tracks. Standard computer graphics techniques enable a photorealistic rendering of the busy urban scene with considerable geometric and photometric detail (Fig. 1).

1.4 Contributions and overview

In this paper, we introduce a sensor management scheme that appears well suited to the challenges of designing camera networks for surveillance applications capable of fully automatic operation. We also develop new gaze-direction controllers for active PTZ cameras. Our work demonstrates the conveniences of the virtual vision paradigm for implementing, experimenting with, and evaluating our surveillance system.

The remainder of the paper is organized as follows: Sect. 2 covers relevant prior work. Section 3 overviews our system. We explain the low-level vision emulation in Sect. 4. Section 5 describes the PTZ active camera controllers and proposes a scheme for learning the map between 3D locations and gaze directions. Section 6 introduces our scheduling strategy. We present our results in Sect. 7 and our conclusions and future research directions in Sect. 8.

Fig. 1 Virtual vision. Synthetic video feeds from multiple virtual surveillance cameras situated in a large-scale virtual train station populated by self-animating pedestrians. **a** Waiting room, **b** Concourses and platforms, **c** Arcade

2 Related work

Previous work on multi-camera systems has dealt with issues related to low and medium-level computer vision, namely identification, recognition, and tracking of moving objects [8–12]. The emphasis has been on tracking and on model transference from one camera to another, which is required for object identification across multiple cameras [13]. Numerous researchers have proposed camera network calibration to achieve robust object identification and classification from multiple viewpoints, and automatic camera network calibration strategies have been proposed for both static and actively controlled camera nodes [6,7].

Little attention has been paid, however, to the problem of controlling or scheduling active cameras when there are more objects to be monitored in the scene than there are available cameras.[2] Some researchers employ a static wide-FOV camera to control an active tilt-zoom camera [18,19]. The cameras are assumed to be calibrated and the total coverage of the cameras is restricted to the FOV of the static camera. Zhou et al. [19] track a single person using an active camera. When multiple people are present in the scene, the person closest to the last tracked person is chosen. The work of Hampapur et al. [20] is perhaps closest to ours in that it deals with the issues of deciding how cameras should be assigned to various people present in the scene. Costello et al. [21] evaluate various strategies for scheduling a single active camera to acquire biometric imagery of the people present in the scene.

In concordance with the virtual vision paradigm [2,3], Santuari et al. [22,23] advocate the development and evaluation of pedestrian segmentation and tracking algorithms using synthetic video generated within a virtual museum simulator containing scripted characters. They focus on low-level computer vision, whereas our work is concerned with high-level computer vision issues, especially multi-camera control in large-scale camera networks.

[2] The problem of online scheduling has been studied extensively in the context of scheduling jobs on a multitasking computer [14,15] as well as for packet routing in networks [16,17].

3 System overview

Our surveillance system is realized within the virtual Penn Station. The network comprises calibrated wide-FOV static cameras and uncalibrated active PTZ cameras. A central server acquires environmental information, such as the 3D positions of the pedestrians, through the static cameras and employs this information to schedule the active cameras in order to visually examine every pedestrian in the scene (Fig. 2).

The virtual cameras acquire synthetic video and employ machine vision routines to identify, recognize, label, and track pedestrians. At present, we assume that pedestrians can be reliably identified across multiple cameras. The static cameras track the 3D locations of pedestrians through *triangulation*. An offline machine learning scheme learns the map between the gaze direction parameters (i.e., pan-tilt angles) of the PTZ cameras and target 3D world locations. Each PTZ camera is modeled as an autonomous agent capable of recording the detailed video of the pedestrian of interest without relying on continuous feedback from the central controller. The PTZ camera uses image-based fixation and zooming routines to follow a pedestrian reliably (Sect. 5).

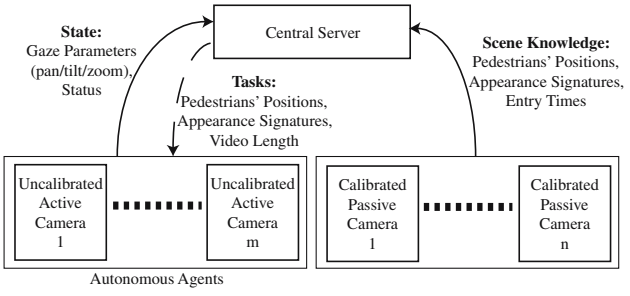

Fig. 2 A central controller uses the scene information collected by calibrated, static cameras to schedule active PTZ cameras for recording close-up videos of the pedestrians in the region of interest. The PTZ cameras are autonomous agents that attempt to achieve the tasks assigned to them by the central controller. The dashed arc indicates the intermittent flow of instructions from the central controller to the PTZ cameras

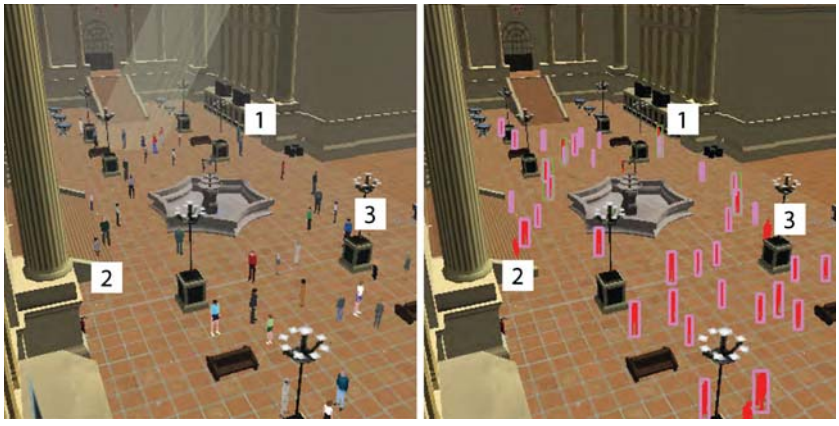

Fig. 3 Pedestrian segmentation and tracking. The labels indicate occasional problems: *1* Multiple pedestrians are grouped together due to poor segmentation; *2* noisy pedestrian segmentation results in a tracking failure; *3* pedestrian segmentation and tracking failure due to occlusion

4 Local vision routines

As we described in [5], each camera has its own suite of Local Vision Routines (LVRs) that support pedestrian recognition, identification, and tracking. The LVRs are computer vision algorithms that directly operate upon the synthetic video generated by the virtual cameras and the information available from the 3D virtual world. The virtual world affords us the benefit of fine tuning the performance of the recognition and tracking modules by taking into consideration the readily available ground truth. Our imaging model emulates camera jitter and imperfect color response; however, it does not yet account for such imaging artifacts as depth-of-field, motion blur, image vignetting, and interlacing. More sophisticated rendering schemes would address this limitation.

We employ appearance-based models to track pedestrians. Pedestrians are segmented to construct personalized color-based pedestrian signatures, which are then robustly matched across the subsequent frames. Pedestrian segmentation is carried out using 3D geometric information as well as background modeling and subtraction. The quality of the segmentation depends upon the amount of noise introduced into the process, and the noise is drawn from Gaussian distributions with appropriate means and variances. Color-based signatures, in particular, have found widespread use in tracking applications [24]. Unfortunately, color-based signatures can be sensitive to illumination changes. We mitigate the shortcoming by operating in HSV space instead of RGB space.

The tracking module mimics the capabilities and limitations of a state-of-the-art tracking system (Figs. 3,4). For example, it can lose track due to occlusions, poor segmentation, or bad lighting. Tracking sometimes locks onto the wrong pedestrian, especially if the scene contains multiple pedestrians with similar visual appearance; i.e., wearing similar clothes. Tracking also fails in group settings when the pedestrian cannot be segmented properly. Our surveillance system is designed to operate robustly in the presence of occasional low-level failures.

For the purposes of this paper, we assume that the scene is monitored by more than one calibrated wide-FOV passive camera plus at least one PTZ active camera. Multiple calibrated static cameras allow the system to use triangulation to compute the location of a pedestrian in 3D, when the pedestrian is simultaneously visible in two or more cameras. For PTZ cameras, zooming can drastically change the appearance of a pedestrian, thereby confounding conventional appearance-based schemes, such as color histogram signatures. We tackle this problem by maintaining HSV color histograms for several camera zoom settings for each pedestrian. Thus, a distinctive characteristic of our pedestrian tracking routine is its ability to operate over a range of camera zoom settings.

We employ histogram intersection to compute the match score between two histograms [25] (Fig. 5). For two normalized histograms $H_1(n)$ and $H_2(n)$, where n is the number of bins, and H_1^i and H_2^i are the number of samples in the ith bin, the match score is

$$H_1 \cap H_2 = \sum_i \min\left(H_1^i, H_2^i\right). \tag{1}$$

Here, the key issue is the selection of the color space and the number of bins or the quantization level. Currently, we use 2D hue-saturation histograms with 32 bins in each dimension. The match score of a histogram H_I against a stored *multi-zoom* color signature H_S is then

$$d(H_I, H_S) = \frac{1}{N_\theta} \sum_\theta \left(H_I \cap H_{S,\theta}\right), \tag{2}$$

where N_θ is the total number of histograms stored across multiple zoom settings and θ represents the field-of-view setting corresponding to the stored histogram $H_{S,\theta}$.

Fig. 4 Tracking pedestrians 1 and 3. Pedestrian 3 is tracked successfully; however, **a** track is lost of pedestrian 1 who blends in with the background. **b** The tracking routine loses pedestrian 3 when she is occluded by pedestrian 2, but it regains track of pedestrian 3 when pedestrian 2 moves out of the way **c**

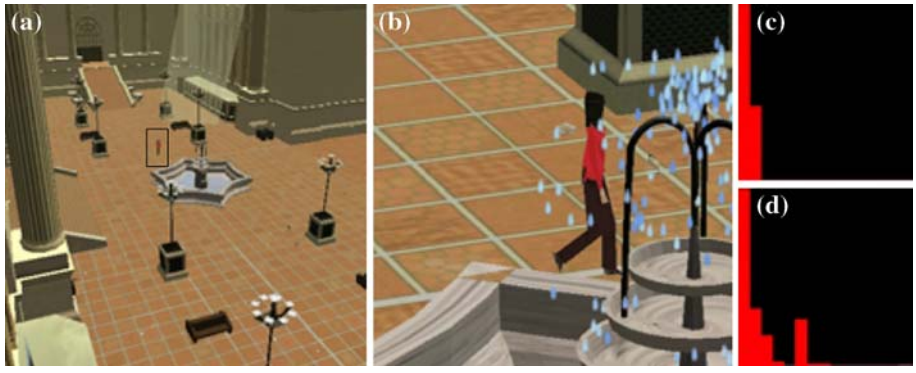

Fig. 5 Color (hue) image pixel histograms of a tracked pedestrian change drastically with zooming. **c** The histogram of the (boxed) subject **a** when the FOV of the camera is set to 45°. **d** The histogram of the same subject when the camera has zoomed in on the subject. To address this problem, a list of histograms is maintained over different zoom settings

5 PTZ active camera controller

We implement each PTZ active camera as a behavior-based autonomous agent [5]. The overall behavior of the camera is determined by the LVR and the current task. The camera controller is modeled as an augmented finite state machine. At the highest level, the camera can be in one of the following states: *free*, *tracking*, *searching*, and *lost* (Fig. 6a). When a camera is free, it selects the next observation request in the task pipeline. The observation requests are of the form, "observe pedestrian i currently at location (x, y, z) for t seconds." When performing the new observation request, the camera selects its widest FOV setting and chooses an appropriate gaze direction using the estimated 3D location of the pedestrian. Upon the successful identification of the pedestrian in question within the FOV, the camera uses image-driven fixation and zooming algorithms to follow the subject.

Each camera can fixate on and zoom in on an object of interest. The fixation and zooming routines are image driven and do not require any 3D information such as camera calibration or a global frame of reference. We discovered that traditional proportional derivative (PD) controllers generate unsteady control signals resulting in jittery camera motion. The noisy nature of tracking induces the PD controller to attempt to minimize the error metric continually without success, so the camera keeps servoing. Hence, we model the fixation and zooming routines as dual-state controllers. The states are used to activate/deactivate the PD controllers. In the *act* state the PD controller tries to minimize the error signal, whereas in the *maintain* state the PD controller ignores the error signal and does nothing (Fig. 6(b)).

The fixate routine brings the region of interest (e.g., the bounding box of a pedestrian) into the center of the image by tilting the camera about its local X and Y axes (Fig. 7, top row). The zoom routine controls the FOV of the camera such that the region of interest occupies the desired percentage of the image. This is useful in situations where, for example, the operator desires a closer look at a pedestrian of interest (Fig. 7, middle row). See [5] for the details of the *fixate* and *zoom* routines.

5.1 Learning the gaze direction computation

Computing an appropriate gaze direction in order to bring the desired pedestrian within the FOV of a camera requires a map between 3D locations in the scene and the associated internal gaze-direction parameters (i.e., the pan-tilt settings) of the camera. This map is learned automatically by observing a pedestrian directed to move around in the scene during an initial learning phase. While the active PTZ cameras are tracking and following the pedestrian, the 3D location of the pedestrian is estimated continuously through triangulation using the calibrated, passive FOV cameras. A lookup table is computed for each PTZ camera that associates the estimated 3D scene location of the target pedestrian with the corresponding gaze angles of the camera. Specifically, for PTZ camera i, this yields tuples of the form $(x^j, y^j, z^j, \alpha^j, \beta^j)$, where j indexes over the scene locations (x, y, z) and corresponding camera pan and tilt angles

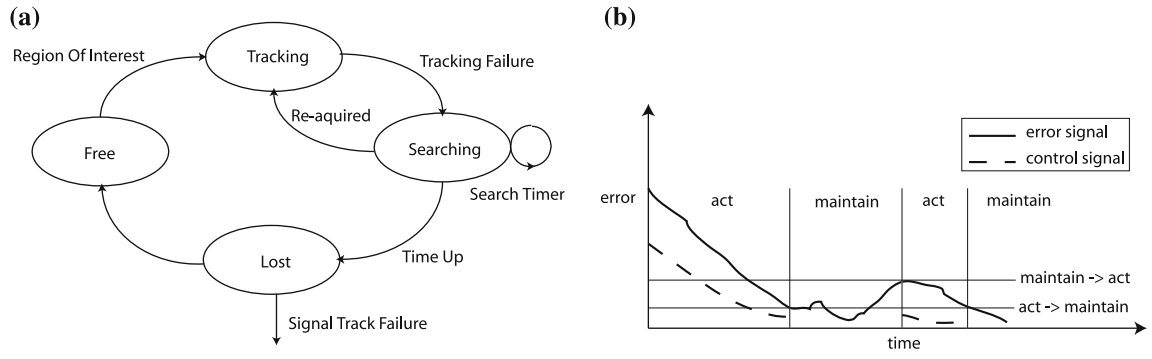

Fig. 6 **a** Top-level camera controller. **b** Dual-state controller for fixation and zooming

Fig. 7 *Top row*: A fixate sequence. *Middle*: A zoom sequence. *Bottom*: Camera returns to its default settings upon losing the pedestrian; it is now ready for another task

(α, β). The lookup table constitutes the training dataset used for learning a continuous map $\mathcal{M} : \Re^3 \rightarrow \Re^2$ between the locations and associated angles. We compute $\mathcal{M}$ using a nearest neighbor approximation. Given any 3D input point $\mathbf{p} = (x, y, z)$, the system estimates the values for α_i and β_i of any camera i that can observe $\mathbf{p}$ as follows: let S_k be the set of k nearest neighbors to $\mathbf{p}$ in $\{\mathbf{p}^j = (x^j, y^j, z^j)\}$, where proximity is computed using the L_2 norm $\|\mathbf{p} - \mathbf{p}^j\|$. Then

$$\alpha_i = \frac{1}{k} \sum_{j:\, \mathbf{p}^j \in S_k} \alpha^j; \quad \beta_i = \frac{1}{k} \sum_{j:\, \mathbf{p}^j \in S_k} \beta^j. \tag{3}$$

This provides only a coarse map between the 3D scene points and the camera pan-tilt angles, but the map is accurate enough in practice to bring the pedestrian within the field of view of the camera.

Figure 8 illustrates an example map learned using the above scheme. Each of the plots shows a plan view of a 3D space, covering the XY-plane ($Z = 0$). The red dots denote 3D points used in learning the map between the 3D world and the pan-tilt settings of the active, PTZ camera. The optical center of this camera is indicated by the green dot at the upper right of each plot. The red points indicate the triangulation-estimated locations of the visually tracked pedestrian, where the active PTZ

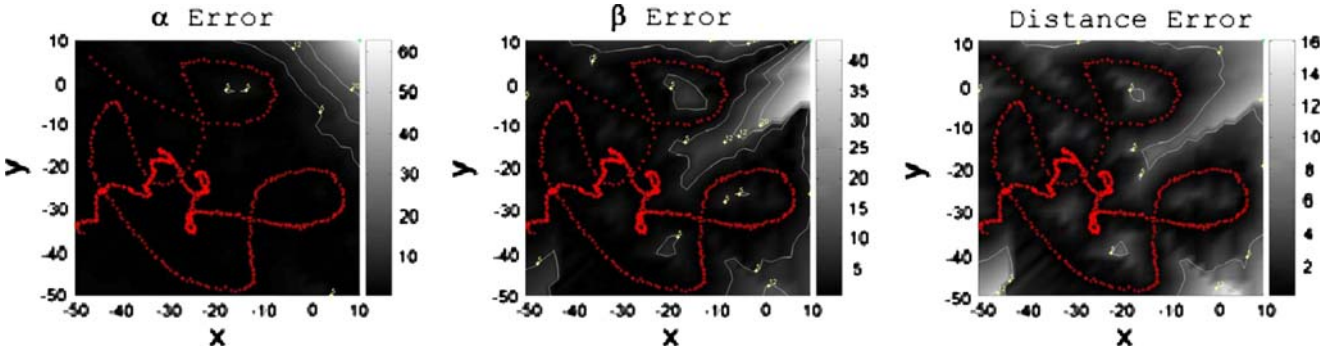

Fig. 8 Map from 3D world locations to PTZ camera gaze direction. Plots of the squared error between the true and estimated **a** pan α and **b** tilt β settings. **c** Euclidean distance between the true 3D location and the point where the XY-plane ($Z = 0$) intersects the optical axis of the PTZ camera when its pan-tilt angles are set to the estimated α and β

camera fixating on the pedestrian has stored an associated pan-tilt setting. Note that the training uses no information about the camera's 3D location.

Each plot shows the squared error between the true and estimated quantities, where brightness is proportional to the magnitude of the error. To generate the plots, we regularly sampled points on the XY-plane and used the position of the camera to compute the true pan-tilt settings for each of the sampled points (true α and β), which constitutes the ground truth. Next, we used the learned map to estimate the camera's pan-tilt settings for every sampled point (estimated α and β). Figure 8a, b plot the squared error in degrees between the true and estimated pan-tilt settings. Figure 8c plots the Euclidean distance between the true 3D location and the point where the XY-plane intersects the optical axis of the PTZ camera when the camera's pan-tilt angles are set to the estimated values. As expected, we observed that the estimate exhibits smaller error in the vicinity of the training data used to learn the map and that it improves over a larger area when the PTZ camera has had the chance to observe longer, more varied, pedestrian tracks like the one shown in the figure.

Upon receiving a new observation request from the scheduler, the active camera estimates the initial gaze direction (α, β) using the learned map. The camera then orients itself in this direction and invokes a visual search behavior (histogram matching) to acquire the pedestrian in question. The distance of $\mathbf{p}$ from the nearest $\mathbf{p}^j$ in the lookup table is a good indicator of the accuracy of the estimated gaze direction and it enables the tuning of the visual search behavior. If the distance is large, the PTZ camera chooses a larger cutoff time for the visual search behavior and selects a wider FOV angle during the search. To minimize the reliance on the initial learning phase, the lookup table is continuously updated when the PTZ camera is following a pedestrian whose 3D location is known.

6 Camera scheduling

The sensor network maintains an internal world model that reflects the current knowledge about the world. The model stores information about the pedestrians present in the scene, including their arrival times and the most current estimates of their positions and headings. The world model is available to the scheduling routine that assigns cameras to the various pedestrians in the scene. Using the 3D information stored in the world model, the cameras choose an appropriate gaze direction when viewing a particular pedestrian. The scheduling algorithm must find a compromise between two competing objectives: (1) to capture high-quality video for as many as possible, preferably all, of the pedestrians in the scene and (2) to observe each pedestrian for as long or as many times as possible, since the chances of identifying a pedestrian improve with the amount of data collected about that pedestrian. At one extreme, the camera follows a pedestrian for their entire stay in the scene, essentially ignoring all other pedestrians, whereas at the other extreme, the camera briefly observes every pedestrian in turn and repeatedly, thus spending most of the time transitioning between different pan, tilt, and zoom settings.

Following the reasoning presented in [1,21], the camera scheduling problem shares many characteristics with the network packet routing problem. Network packet routing is an online scheduling problem where the arrival times of packets are not known a priori and where each packet must be served for a finite duration before a deadline, when it is dropped by the router. Similarly, in our case, the arrival times of pedestrians entering the scene is not known a priori and a pedestrian must be observed for some minimal amount of time by one of the PTZ cameras before (s)he leaves the scene. That time serves as the deadline.

However, our problem differs from the packet routing problem in several significant ways. First, continuing with network terminology, we have multiple "routers" (one for every PTZ camera) instead of just one. This aspect of our problem is better modeled using scheduling policies for assigning jobs to different processors. Second, we typically must deal with additional sources of uncertainty: (1) it is difficult to estimate when a pedestrian might leave the scene and (2) the time period during which a PTZ camera should track and follow a pedestrian to record high-quality video suitable for further biometric analysis can vary depending on multiple factors; e.g., a pedestrian suddenly turning away from the camera, a tracking failure, an occlusion, etc.

Consequently, camera scheduling is an *online scheduling* problem (in scheduling jargon, a PTZ camera is a "processor" or "machine" and a pedestrian is a "job") for which (1) jobs are released over time, (2) have deadlines, (3) have different processing requirements, (4) can be assigned to one of many processors, and (5) the scheduler must schedule them without any knowledge of the future. Our approach is loosely related to the Multi Level Feedback Algorithm used for process scheduling in the Unix and Windows NT operating systems [15].

6.1 Online scheduling paradigms

An online scheduling algorithm does not have access to the entire input instance as it makes its decisions [26]. Thus, at each time t, the scheduler must decide which job(s) to run at t. Here, jobs typically have release times, and the scheduler is not aware of the existence of a job until its release time.

An online scheduler is referred to as "clairvoyant" when the processing times of the jobs are available to the scheduler upon their arrival. Such schedulers are modeled within an *online-time* paradigm. Alternatively, the lack of job processing time information is called "non-clairvoyance" and such schedulers are modeled using the *online-time-nclv* paradigm. Since the exit times of pedestrians are difficult to predict, our scheduler is non-clairvoyant.

Another issue that arises in scheduling algorithms is that of *preemption*; i.e., interrupting a running job to process another job. A common scheme for implementing preemption is to pause an active job and later resume it on the same processor. Another possibility, which only exists in the online setting and is meaningless for offline scheduling algorithms, is to stop an existing job and restart it from the beginning on the same or a different processor. It is well known that if preemption is not allowed for problems in either the *online-time* or *online-time-nclv* model, and jobs can have arbitrary processing times, then the schedules produced by any online scheduler will be far from optimal [15]. This is why most online schedulers generally allow preemption, unless all jobs have similar processing times. Preemption incurs an overhead as job swapping (pausing/resuming jobs, saving/restoring job states, etc.) consumes resources.

In the camera scheduling application, for example, job swapping involves, among other overheads, locating the other pedestrian and initializing the tracking routine. Preemption is especially relevant to our application, as without it a PTZ camera can potentially track a pedestrian indefinitely without ever succeeding in recording video suitable for further biometric analysis. We therefore allow preemption. When there are multiple pedestrians in the scene, a PTZ camera is assigned to a pedestrian for some fixed maximum duration, after which the PTZ camera can be reassigned even if the video recording was unsuccessful.[3] The pedestrian whose video recording is terminated is treated as a new arrival and, later, the same or a different camera can attend to the pedestrian.

Figure 9 shows examples of the scheduling of a single camera to observe multiple pedestrians with and without preemption. Without preemption, the camera observes pedestrian 1 until sufficient video is collected, ignoring all other pedestrians (Fig. 9a). By contrast, preemption prevents the camera from being monopolized by pedestrian 1, so the camera can attend to pedestrians 2, 3, and 4 even before sufficient video of pedestrian 1 has been acquired (Fig. 9b). The camera later resumes observing pedestrian 1; however, the pedestrian leaves the scene before suitable video has been collected. Preemption is not advantageous in all situations and it can have unintended side effects. In Fig. 9c, pedestrian 1 was successfully observed; however, with preemption, none of the pedestrians were observed long enough in Fig. 9d.

For more effective preemption, we have adopted a multi-class pedestrian (job) model (Fig. 11a, b). Every pedestrian is assigned to a class based on how many times the pedestrian has been observed successfully by a PTZ camera. Each sensed pedestrian is initialized as a member of class 0 and advances to the next higher class after each successful observation. The class numbers of pedestrians together with their arrival times determine their positions in the priority queue, with priority given to pedestrians in lower classes. For example,

[3] An observation is deemed successful when the recorded video is suitable for further biometric analysis. The duration of the video depends on a number of factors, including the requirements of the biometric analysis routine, the quality of the recorded video, etc.

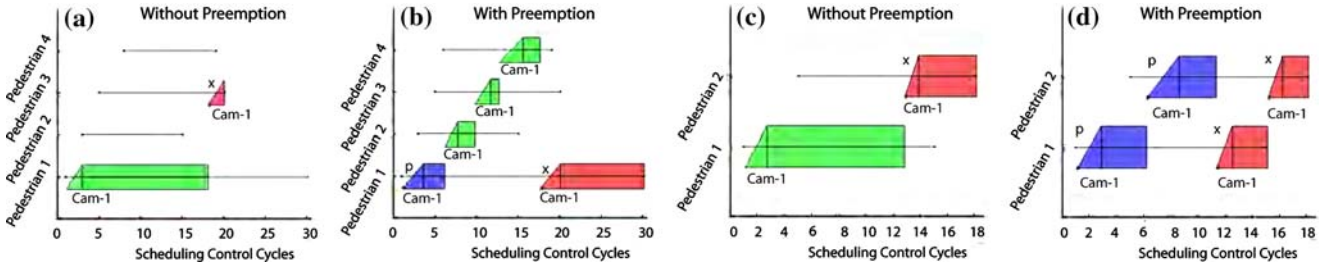

Fig. 9 Scheduling a single camera to observe multiple pedestrians with and without preemption. **a**, **b** A camera is scheduled to observe four pedestrians. **c**, **d** A camera is scheduled to observe two pedestrians. *Triangles* represent lead times, *rectangles* represent processing times. *Green* indicates a successful recording, *blue* (or 'p') indicates a preemptive abort. *Red* (or 'x') indicates a failed recording due to the departure of a pedestrian

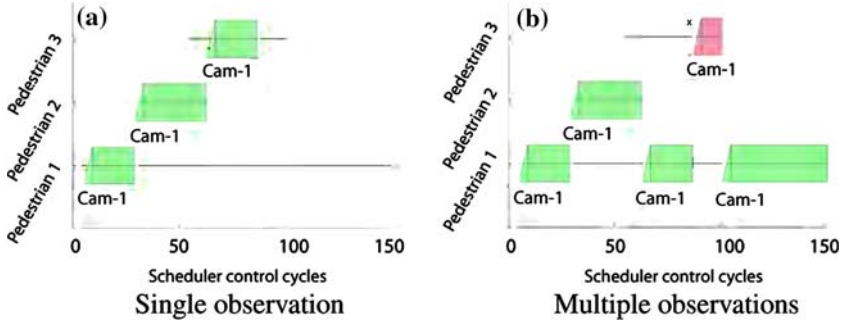

Fig. 10 Scheduling a single camera to observe three pedestrians in single-observation and multiple-observations modes. A pedestrian's position in the priority sequence is computed using the FCFS policy

cameras currently observing pedestrians belonging to classes $1, 2, 3, \ldots$, are immediately reassigned to class 0 pedestrians that have not yet been recorded (Fig. 11b).

Figures 10 and 11 compare the naive FCFS-based priority calculation against the multi-class priority computation scheme. The multi-class priority scheme is irrelevant when each pedestrian is observed only once. In single observation mode, each pedestrian is observed exactly once; e.g., in Fig. 10a the camera is free after pedestrian 3's departure, yet the camera is not assigned to observe pedestrian 1 who is still present. However, it is desirable to observe a pedestrian multiple times when possible, in order to collect more data. In Fig. 10b, the camera is instructed to observe every pedestrian for as many times as possible, ergo pedestrian 1 is observed twice. When the camera finishes observing pedestrian 2, pedestrian 1 is ahead of pedestrian 3 in the priority sequence since the arrival time of pedestrian 1 preceded the arrival time of pedestrian 3.[4] The scheduler uses the naive FCFS priority computation, which does not differentiate between pedestrians 1 and 3 based on their class memberships: pedestrians 1 and 3 belong to Classes 1 and 0, respectively. Therefore, the camera is assigned to observe pedestrian 1 *again* and pedestrian 3 goes unnoticed.

[4] In the multiple observations setting, the arrival time of a pedestrian is updated after each successful recording; e.g., in Fig. 11b, the arrival time of pedestrian 1 was changed from the true arrival time, 5, to 23 after the pedestrian was successfully recorded.

Multi-class priority calculation resolves the above issue (Fig. 11a). After the camera has successfully observed pedestrian 2, priority is given to pedestrian 3 (class 0) over pedestrian 1 (class 1), even though the arrival time of pedestrian 1 precedes that of pedestrian 3. Consequently, the camera is assigned to Pedestrian 3 and the camera successfully observes all three pedestrians. Furthermore, multi-class priority calculation allows more intelligent preemption. Consider Fig. 11b: the camera is observing pedestrian 1 for the second time when pedestrian 3 enters the scene. Upon sensing the new arrival, the camera immediately aborts observing pedestrian 1 (class 1) and attends to the previously unseen pedestrian (pedestrian 3, class 0).

6.2 Problem formulation

The standard three-field notation for describing scheduling problems [27] is $\alpha|\beta|\gamma$, where α describes the processing environment, β encodes the job characteristics, and γ is the performance criterion for the scheduling algorithm. Using this notation, we can describe our camera scheduling problem as

$$P \mid r_j, \text{online-time-nclv}, \text{pmtn} \mid \sum U_j; \tag{4}$$

i.e., find a schedule on m processors that minimizes the total unit penalty when jobs j with deadlines d_j are released at time r_j. The jobs require arbitrary processing times and preemption (pmtn) is allowed. Minimizing the

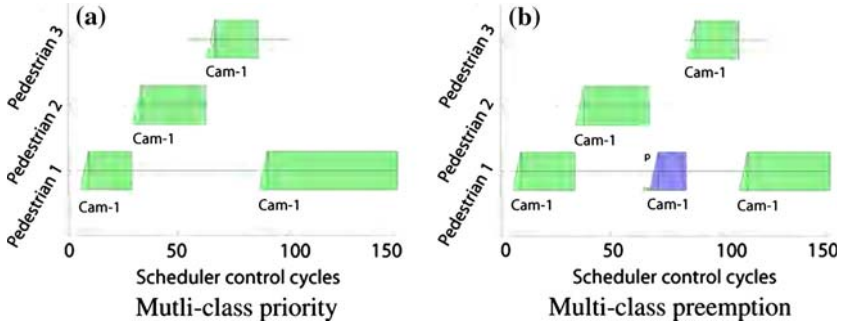

Fig. 11 A single camera is scheduled to observe three pedestrians in multiple observation mode. The pedestrians' positions in the priority sequence is calculated using their arrival times and class memberships

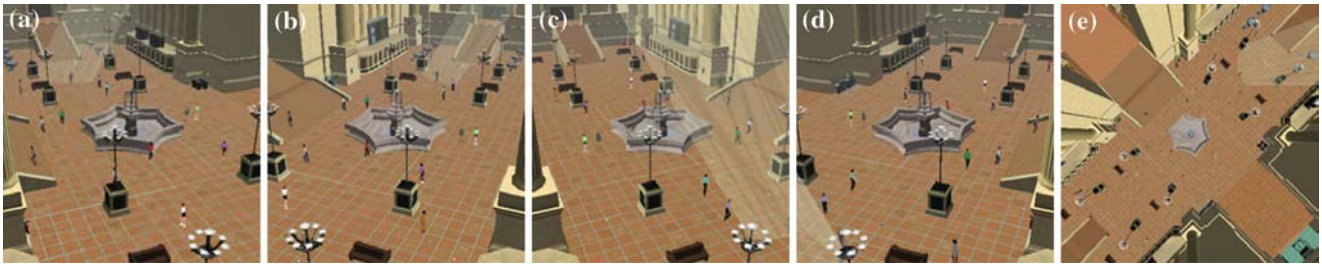

Fig. 12 **a–d** Images from wide-FOV passive cameras situated at the four corners of the main waiting room in the train station. **e** Image from a fish-eye camera mounted at the ceiling of the waiting room. These static cameras are calibrated, enabling the 3D positions of observed pedestrians to be estimated through triangulation

total unit penalty is akin to maximizing the number of jobs successfully completed prior to their deadlines. If C_j is the completion time of a job j, then

$$U_j = \begin{cases} 0 & \text{if } C_j \leq d_j; \\ 1 & \text{otherwise.} \end{cases} \tag{5}$$

The complexity of problem (4) is not known. However, the simpler problem $P2 \mid r_j, \text{pmtn} \mid \sum U_j$,[5] is at least NP-hard [28]. Hence, our formulation of the camera scheduling problem is likely NP-hard. Consequently, we resort to a greedy algorithm for scheduling cameras to observe pedestrians.

The obvious nonclairvoyant online algorithms are Round Robin (RR) and Shortest Elapsed Time First (SETF). RR devotes identical processing resources to all jobs, whereas SETF devotes *all* resources to the job that has been processed the least. As SETF is known to perform poorly when jobs are not fully parallelizable [29], we use weighted RR, a variant of RR. The weighted RR scheduling scheme is used to assign jobs to multiple processors with different load capacities. Each processor is assigned a weight indicating its processing capacity and more jobs are assigned to the processors with higher weights. We model each PTZ camera as a processor whose weights, which quantify the suitability of a camera with respect to observing a pedestrian, are adjusted dynamically. The weights are determined by two factors: (1) the number of adjustments the camera needs to make in the PTZ coordinates to fixate on the pedestrian and (2) the distance separating the pedestrian from the camera. In order to ensure fairness, we use a *First Come, First Served* (FCFS) priority scheme to select jobs that should be assigned to a processor (preemption forces job priorities to vary over time). Additionally, FCFS is said to be optimal when the optimization criterion is to minimize the maximum flow time, which is a measure of the quality of service defined as $C_j - r_j$ [26].

On the one hand, a camera that requires small adjustments in the PTZ coordinates to fixate on a pedestrian usually needs less *lead time* (the total time required by a PTZ camera to fixate on a pedestrian and initiate video recording) than a camera that needs to turn more drastically in order to bring the pedestrian into view. Consequently, we assign a higher weight to a camera that needs less redirection in order to observe the pedestrian in question. On the other hand, a camera that is closer to a pedestrian is more suitable for observing this pedestrian, since such an arrangement can potentially avoid occlusions, tracking loss, and subsequent re-initialization, by reducing the chance of another pedestrian intervening between the camera and the subject being recorded. The camera weights with respect to a pedestrian are computed as

[5] I.e., find a schedule on two processors that minimize the total unit penalty under release time constraints r_j, where release times are known a priori and preemption is allowed.

Fig. 13 Sample close-up images captured by the PTZ active cameras

$$w = \begin{cases} \exp\left(-\frac{(\theta-\hat{\theta})^2}{2\sigma_\theta{}^2} - \frac{(\alpha-\hat{\alpha})^2}{2\sigma_\alpha{}^2} - \frac{(\beta-\hat{\beta})^2}{2\sigma_\beta{}^2}\right) \\ \qquad \text{if the camera is free;} \\ 0 \qquad \text{if the camera is busy,} \end{cases} \tag{6}$$

where $\hat{\theta} = (\theta_{\min} + \theta_{\max})/2$, $\hat{\alpha} = (\alpha_{\min} + \alpha_{\max})/2$, and $\hat{\beta} = (\beta_{\min} + \beta_{\max})/2$, and where $\theta_{\min}$ and $\theta_{\max}$ are extremal field of view settings, $\alpha_{\min}$ and $\alpha_{\max}$ are extremal rotation angles around the x-axis (up–down), and $\beta_{\min}$ and $\beta_{\max}$ are extremal rotation angles around the y-axis (left–right). The values of the standard deviations $\sigma_\theta, \sigma_\alpha$, and σ_β associated with each attribute are chosen empirically (in our experiments, we assigned $\sigma_\theta = \sigma_\alpha = \sigma_\beta = 5.0$). Here, α and β are the gaze parameters corresponding to the 3D location of the pedestrian as computed by the triangulation process, and θ is an approximate measure of the distance between the camera and the pedestrian. The distance between the camera and the pedestrian can also be approximated by declination angle, which can be estimated from α, under a ground-plane assumption.

A danger of using weighted round-robin scheduling is the possibility that a majority of the jobs will be assigned to the processor with the highest weight. We avoid this situation by sorting the PTZ cameras according to their weights with respect to a given pedestrian and assigning the free PTZ camera with the highest weight to that pedestrian. The FCFS policy breaks ties on the basis of arrival times and pedestrian classes. Pedestrians in class 0 (i.e., never observed by a PTZ camera) have the highest priority. Among pedestrians belonging to class 0, the pedestrian that entered the scene first is selected. Pedestrians belonging to classes 1 or higher are similarly selected on the basis of their arrival times. The arrival times of the pedestrians are maintained by the sensor network and are made available to the PTZ cameras.

The amount of time a PTZ camera will observe a pedestrian depends upon the number of pedestrians in the scene. However, we have specified a minimum time that a PTZ camera must spend observing a pedestrian. This is determined by the minimum length of video sequence required by the biometric routines that perform further evaluation plus the average time it takes a PTZ camera to lock onto and zoom into a pedestrian. To implement preemption, we specify the maximum time that a camera can spend observing a pedestrian when there are multiple pedestrians in the scene. The proposed scheduling scheme strikes a balance between the two often competing goals of following a pedestrian for as long as possible and observing as many pedestrians as possible.

7 Results

We populated the virtual train station with up to 20 autonomous pedestrians that enter, go about their business, and leave the waiting room of their own volition. We tested our scheduling strategy in various scenarios using from 1 to 18 PTZ active cameras. For example, Fig. 12 shows our prototype surveillance system utilizing five wide-FOV static cameras situated within the waiting room of the train station. Figure 13 shows sample close-up images captured by four PTZ active cameras. The system behaved as expected in all cases, correctly tasking the available cameras using weighted round-robin scheduling with an FCFS priority policy.

In a typical experiment (Fig. 14a), when only one PTZ camera is available, pedestrians 1, 2, 4, 7, 9, 10, 13, and 16 were recorded, but pedestrians 3, 5, 6, 8, 11, 12, 14, 15, 17, 18, 19, and 20 go unnoticed since they left the scene before the camera had an opportunity to observe them. Figure 14b, c shows the results from the same run with 2 and 4 active cameras, respectively. In the 2-camera case, even though the performance has improved significantly from the added camera, pedestrians 12, 17, 18, 19, and 20 still go unnoticed. With four PTZ cameras, the system is now able to observe every pedestrian. As expected, the chances of observing multiple pedestrians improve as more cameras become available.

For Fig. 14d, we have populated the virtual train station with only three autonomous pedestrians, leaving all other parameters unchanged. Given that there are now only three pedestrians in the scene, even a single camera successfully observes them. Next, we ran the simulation with 20 pedestrians (Fig. 14e). This time, however, we

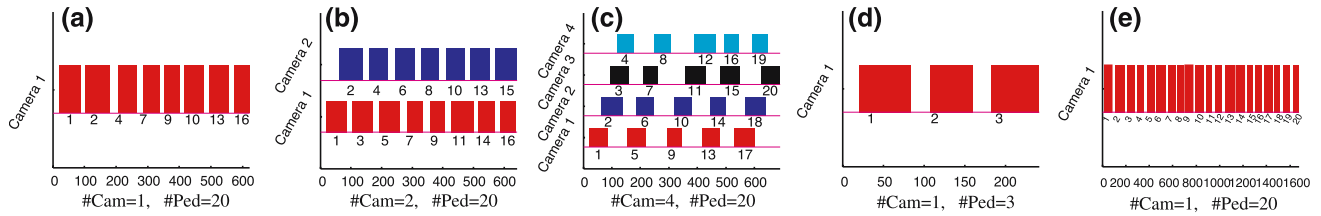

Fig. 14 Pedestrians are assigned unique identifiers based on their entry times; e.g., pedestrian 1 always enters the scene at the same time or before the arrival of pedestrian 2. **a–c** Twenty pedestrians are present in the scene. **a** The scheduling policy for one camera: camera 1 successfully recorded pedestrians 1, 2, 4, 7, 9, 10, 13, and 16. **b–c** Adding more cameras improves the chances of observing more pedestrians. Only pedestrians 12, 17, 18, 19, and 20 go unnoticed when two cameras are available. With four cameras all pedestrians are observed. **d** The scene is populated with three pedestrians. **e** Twenty pedestrians, who tend to linger. The chances of a set of cameras to observe the pedestrians increase when **d** there are fewer pedestrians or when **e** pedestrians tend to linger

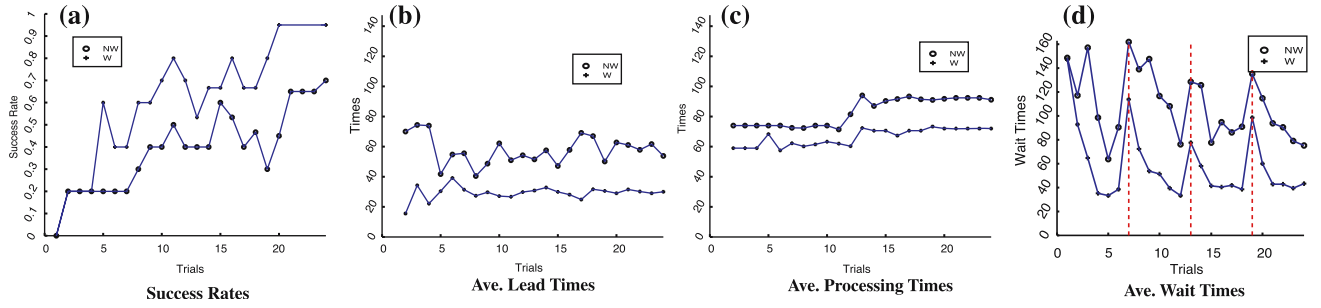

Fig. 15 A comparison of weighted (*W*) and non-weighted (*NW*) scheduling schemes. Equation (6) is used to compute camera weights (i.e., relevances) with respect to a pedestrian. The weighted scheduling strategy outperforms its non-weighted counterpart as is evident from its higher success rates (**a**) and shorter lead (**b**), processing (**c**), and wait (**d**) times. The displayed results are averaged over several runs in each trial scenario. Trials 1–6 involve five pedestrians and from 1 to 6 cameras, respectively. Trials 7–12 involve ten pedestrians and from 3 to 8 cameras, respectively. Trials 13–18 involve 15 pedestrians and 5, 6, 9, 10, 11, and 12 cameras, respectively. Trials 19–24 involve 20 pedestrians with 5, 8, 10, 13, 15, and 18 cameras, respectively

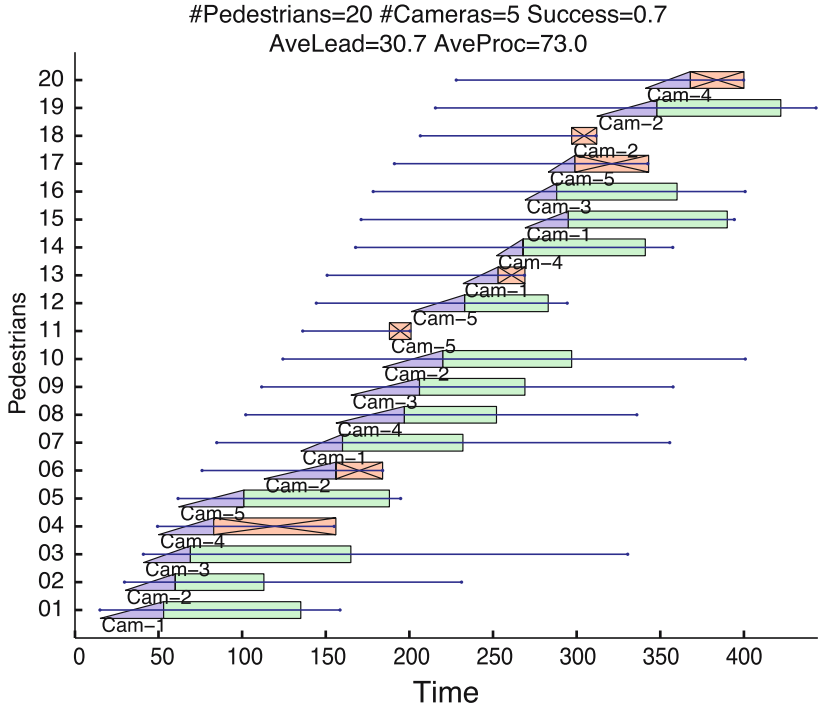

Fig. 16 Scheduling results for Trial #19 using weighted camera assignment. *Blue lines* represent the entry and exit times, the *blue triangles* represent the lead times, the *green rectangles* represent the processing times, and the *red crossed rectangles* represent an aborted attempt at capturing the video of a pedestrian

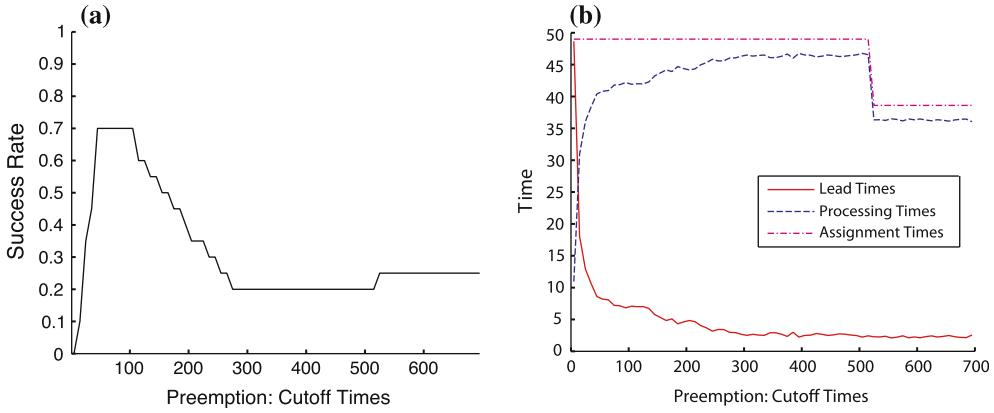

Fig. 17 A single camera is scheduled to observe 20 pedestrians. The performance of the scheduler depends in part upon the preemption cutoff time; i.e., the duration after which a camera is reassigned to observe another pedestrian even if its current assignment has not yet completed. **a** A small cutoff time can result in a greater number of preemptions. Consequently, more time is spent in transitions between pedestrians; **b** The average lead, processing, and assignment times. Note the higher average lead times corresponding to smaller preemption cutoff times; thereby, reducing the overall performance. Alternatively, for large cutoff times, a camera might continue recording a single pedestrian

changed the behavior settings of the pedestrians so they tend to linger in the waiting room. Here too, a single camera successfully observed each of the 20 pedestrians. We conclude that even a few cameras can perform satisfactorily when there are either few pedestrians in the scene or when the pedestrians tend to spend considerable time in the area.

In Fig. 15, we compare the scheduling scheme that treats all cameras equally with the weighted scheduling scheme that takes into account the suitability of any camera in observing a pedestrian. As expected, the weighted scheduling scheme outperforms its non-weighted counterpart, exhibiting higher success rates (as defined by fraction of pedestrians successfully recorded) and lower average lead time, processing time (the time spent recording the video of a pedestrian), and wait time (the time elapsed between the entry of a pedestrian and when the camera begins fixating on the pedestrian). The lower average lead and processing times are a direct consequence of (6) for computing the suitability of a camera to recording a pedestrian. Interestingly, the average wait times do not necessarily decrease as we increase the number of cameras. Figure 16 shows detailed results for the scenario with 20 pedestrians and 5 cameras.

The lack of preemption can impact the overall performance of the scheduler. Choosing an appropriate value for the preemption cutoff time is critical to achieving the balance between the competing goals of observing as many pedestrians as possible and observing each pedestrian for as long as possible. Too small a value will result in the cameras spending most of their time transitioning between pedestrians, whereas too large a value will result in cameras myopically dwelling on pedestrians (Fig. 17). We found that the average camera assignment time is a good indicator of the preemption cutoff time. In the current setting, the camera assignment time equals lead time plus processing time, and the preemption cutoff time should be no less than the average camera assignment time. However, if the variation in camera assignment times is large, then the average assignment time is a poor indicator of preemption cutoff times. Average assignment times can be computed on the fly using, e.g., a running average.

We compared the proposed camera scheduling algorithm in the following six configurations: (1) no preemption, single observation, single class (NPSOSC), (2) no preemption, multiple observation, single class (NPMOSC), (3) no preemption, multiple observation, multi-class (NPMOMC), (4) preemption, single observation, single class (PSOSC), (5) preemption, multiple observation, single class (PMOSC), and (6) preemption, multiple observation, multi-class (PMOMC) (Fig. 18). For these tests, 1–4 cameras were scheduled to observe up to 20 pedestrians. The pedestrians enter the main waiting room of the station in groups of 10. Each group precedes the next by about 150 scheduler control cycles. Each pedestrian spends anywhere from 850 to 1,500 scheduler control cycles in the waiting room. The preemption cutoff time is set to 170.

Preemption appears to be useful when the camera to pedestrian ratio is small or when there is a potential for cameras to continually record videos of the assigned pedestrians without success (a surprisingly common occurrence in camera scheduling due to tracking/occlusion issues). Scheduling without preemption has similar, or in some cases even higher, success rates than

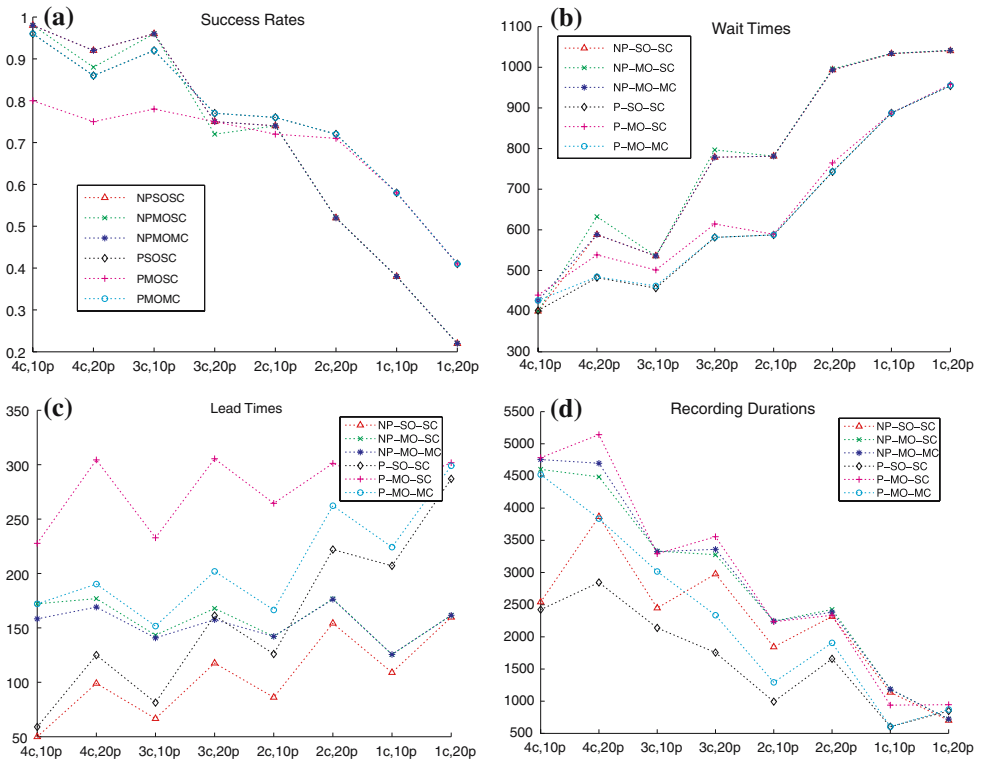

Fig. 18 Comparison of various scheduler configurations. One to four cameras are scheduled to observe 10 or 20 pedestrians. Along the X-axis, letters 'c' and 'p' refer to the number of cameras and pedestrians, respectively for example '4c,10p' denotes the test consisting of scheduling four cameras to observe ten pedestrians. The results are averaged over five trials each

those achieved with preemption when the camera-to-pedestrian ratio is large (e.g., in Fig. 18a, when three or four cameras are available to observe up to 20 pedestrians). The success rates for no-preemption scheduling, however, drop rapidly as the number of available cameras is reduced, and no-preemption scheduling is outperformed by its preemption-savvy counterpart; e.g., when one or two cameras are scheduled to observe 10 or 20 pedestrians (Fig. 18a).

Single observation (SO) scheduling features better success rates than the corresponding multiple observation (MO) scheduling, which can be attributed to the longer wait times for the latter (Fig. 18b), except when the multi-class pedestrian model is employed. Multi-class pedestrian modeling can reduce the wait times as it enables the scheduler to focus on the pedestrians that have been overlooked. When using MO scheduling, the lowest wait times are obtained by combining multi-class pedestrian models with preemption. PMOMC scheduling, for example, achieves wait times comparable to PSOSC with the added advantage of multiple observations. Our tests also confirm the intuition that multiple observation scheduling yields better data (i.e., longer duration video tracks) than single observation scheduling (Fig. 18d). PMOMC offers the highest lead times as a result of the highest number of preemptions combined with high success rates (Fig. 18c).

Our results suggest that PMOMC has the best overall performance: high success rates, low wait times, and longer recorded video durations. NPSOSC has better success rates under favorable circumstances – a good camera to pedestrian ratio and when all pedestrians have similar processing requirements.

8 Conclusion

Future surveillance systems will comprise networks of static and active cameras capable of providing perceptive coverage of extensive environments with minimal reliance on a human operator. Such systems will require not only robust, low-level vision routines, but also novel sensor network methodologies. Building on our earlier work [5], the work presented in this paper is another step toward their realization.

In particular, we have introduced a scheduling strategy for intelligently managing multiple, uncalibrated, active PTZ cameras, supported by several static, calibrated cameras, in order to satisfy the challenging task of automatically recording close-up, biometric videos of pedestrians present in a scene. We have found the PMOMC (preemption, multiple observation, multi-class) scheduling scheme to be the most suitable one for this purpose. At present, predicting pedestrian behaviors is at best an inexact science, so we have intentionally avoided scheduling policies that depend on such predictions.

An interesting feature of our work is that we have demonstrated our prototype surveillance system in a virtual train station environment populated by autonomous, lifelike pedestrians. This simulator facilitates our ability to design large-scale sensor networks in virtual reality and experiment with them on commodity personal computers. The future of such advanced simulation-based approaches appears promising for the purposes of low-cost design and experimentation. Since scalability is an issue when dealing with numerous active cameras spread over a large area, we hope to tackle the scalability issue by investigating distributed scheduling strategies.

Acknowledgments The research reported herein was supported in part by a grant from the Defense Advanced Research Projects Agency (DARPA) of the Department of Defense. We thank Dr. Tom Strat, formerly of DARPA, for his generous support and encouragement. We also thank Wei Shao and Mauricio Plaza-Villegas for implementing the Penn Station simulator and a prototype virtual vision infrastructure within it.

References

1. Qureshi, F., Terzopoulos, D.: Surveillance camera scheduling: a virtual vision approach. In: Proceedings of the 3rd ACM International Workshop on Video Surveillance and Sensor Networks, (Singapore), pp. 131–139 (2005)
2. Terzopoulos, D., Rabie, T.: Animat vision: active vision in artificial animals. Videre. J. Comput. Vision Res. **1**, 2–19 (1997)
3. Terzopoulos, D.: Perceptive agents and systems in virtual reality. In: Proceedings of the 10th ACM Symposium on Virtual Reality Software and Technology, (Osaka, Japan), pp. 1–3 (2003)
4. Shao, W., Terzopoulos, D.: Autonomous pedestrians. In: Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation, (Los Angeles, CA), pp. 19–28 (2005)
5. Qureshi, F., Terzopoulos, D.: Towards intelligent camera networks: a virtual vision approach. In: Proceedings of the 2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS05), (Beijing, China), pp. 177–184 (2005)
6. Pedersini, F.,Sarti, A., Tubaro, S.: Accurate and simple geometric calibration of multi-camera systems. Signal Process. **77**(3), 309–334 (1999)
7. Gandhi, T., Trivedi, M.M.: Calibration of a reconfigurable array of omnidirectional cameras using a moving person. In: Proceedings of the 2nd ACM International Workshop on Video Surveillance and Sensor Networks, (New York), pp. 12–19 (2004)
8. Collins, R., Amidi, O., Kanade, T.: An active camera system for acquiring multi-view video. In: Proceedings of the International Conference on Image Processing, (Rochester), pp. 517–520 (2002)
9. Kang, J., Cohen, I., Medioni, G.: Multi-views tracking within and across uncalibrated camera streams. In: Proceedings of the ACM SIGMM International Workshop on Video Surveillance, (New York), pp. 21–33 (2003)
10. Comaniciu, D., Berton, F., Ramesh, V.: Adaptive resolution system for distributed surveillance. Real Time Imag. **8**(5), 427–437 (2002)
11. Trivedi, M., Huang, K., Mikic, I.: Intelligent environments and active camera networks. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, vol. 2, pp. 804–809 (2000)
12. Stillman, S., Tanawongsuwan, R., Essa, I.: A system for tracking and recognizing multiple people with multiple cameras. Technical Report GIT-GVU-98-25, Georgia Institute of Technology, GVU Center (1998)
13. Khan, S., Shah, M.: Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. IEEE Trans. Pattern Anal. Mach. Intell. **25**, 1355–1360 (2003)
14. Bar-Noy, A., Guha, S., Naor, J., Schieber, B.: Approximating the throughput of multiple machines in real-time scheduling. SIAM J. Comput. **31**(2), 331–352 (2002)
15. Sgall, J.: Online scheduling: a survey. In: On-Line Algorithms: The State of the Art, Lecture Notes in Computer Science, pp. 192–231. Springer, Berlin Heidelberg New York (1998)
16. Ling, T., Shroff, N.: Scheduling real-time traffic in ATM networks. In: Proceedings of IEEE Infocom pp. 198–205 (1996)
17. Givan, R., Chong, E., Chang, H.: Scheduling multiclass packet streams to minimize weighted loss. Queueing Syst. Theory Appl. **41**(3), 241–270 (2002)
18. Collins, R., Lipton, A., Fujiyoshi, H., Kanade, T.: Algorithms for cooperative multisensor surveillance. Proc IEEE, **89**, 1456–1477 (2001)
19. Zhou, X., Collins, R.T., Kanade, T., Metes, P.: A master–slave system to acquire biometric imagery of humans at distance. In: Proceedings of the ACM SIGMM International Workshop on Video Surveillance, (New York), pp. 113–120 (2003)
20. Hampapur, A., Pankanti, S., Senior, A., Tian, Y.-L., Brown, L., Bolle, R.: Face cataloger: Multi-scale imaging for relating identity to location. In: Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, (Washington, DC), pp. 13–21 (2003)
21. Costello, C.J., Diehl, C.P., Banerjee, A., Fisher, H.: Scheduling an active camera to observe people. In: Proceedings of the 2nd ACM International Workshop on Video Surveillance and Sensor Networks, (New York), pp. 39–45 (2004)
22. Santuari, A., Lanz, O., Brunelli, R.: Synthetic movies for computer vision applications. In: Proceedings of the 3rd IASTED International Conference: Visualization, Imaging, and Image Processing (VIIP 2003), no. 1, (Spain), pp. 1–6 (2004)
23. Bertamini, F., Brunelli, R., Lanz, O., Roat, A., Santuari, A., Tobia, F., Xu, Q.: Olympus: An ambient intelligence architecture on the verge of reality. In: Proceedings of the 12th International Conference on Image Analysis and Processing, (Italy), pp. 139–145 (2003)
24. Siebel, N.T.: Designing and Implementing People Tracking Applications for Automated Visual Surveillance. PhD Thesis, Department of Computer Science. The University of Reading (2003)
25. Swain, M.J., Ballard, D.M.: Color indexing. Int. J. Comput. Vision **7**, 11–32 (1991)
26. Fiat, A., Woeginger, G.J. (eds.): Online Algorithms, The State of the Art, vol. 1442 of Lecture Notes in Computer Science. Springer, Berlin Heidelberg New York (1998)
27. Graham, R.L., Lawler, E.L., Lenstra, J.K., Kan, A.H.G.R.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discrete Math **5**, 287–326 (1997)
28. Du, J., Leung, J.-T., Wong, C.: Minimizing the number of late jobs with release time constraints. J. Combinatorial Math. Combinatorial Comput **11**, 97–107 (1992)
29. Dobson, G.: "Scheduling independent tasks on unrelated processors. J. ACM **13**, 705–716 (1984)