Towards Intelligent Camera Networks: A Virtual Vision Approach

Faisal Z. Qureshi[1] and **Demetri Terzopoulos**[2,1]

[1]Department of Computer Science, University of Toronto, Toronto ON M5S 3G4, Canada
[2]Courant Institute of Mathematical Sciences, New York University, New York, NY 10003, USA

Abstract

The goals of this paper are two-fold: (i) to present our initial efforts towards the realization of a fully autonomous sensor network of dynamic video cameras capable of providing perceptive coverage of a large public space, and (ii) to further the cause of exploiting visually and behaviorally realistic virtual environments in the development and testing of machine vision systems. In particular, our proposed sensor network employs techniques that enable a collection of active (pan-tilt-zoom) cameras to collaborate in performing various visual surveillance tasks, such as keeping one or more pedestrians within view, with minimal reliance on a human operator. The network features local and global autonomy and lacks any central controller, which entails robustness and scalability. Its functionality is the result of local decision-making capabilities at each camera node and communication between the nodes. We demonstrate our surveillance system in a virtual train station environment populated by autonomous, lifelike virtual pedestrians. Our readily reconfigurable virtual cameras generate synthetic video feeds that emulate those generated by real surveillance cameras monitoring public spaces. This type of research would be difficult in the real world given the costs of deploying and experimenting with an appropriately complex camera network in a large public space the size of a train station.

1. Introduction

Recent advances in camera and video technologies have made it possible to network numerous video cameras together in order to provide visual coverage of large public spaces such as airports or train stations. As the size of the camera network grows and the level of activity in the public space increases, it becomes infeasible for human operators to monitor the multiple video streams and identify all events of possible interest, or even to control individual cameras in performing advanced surveillance tasks, such as zooming in on a particular subject of interest to acquire one or more facial snapshots. Consequently, a timely challenge for computer vision researchers is to design camera sensor networks capable of performing visual surveillance tasks autonomously, or at least with minimal human intervention.

Even if there were no legal obstacles to monitoring people in public spaces for experimental purposes, the cost of deploying a large-scale camera network in the real world and experimenting with it can easily be prohibitive for computer vision researchers. As was argued in [1], however, computer graphics and virtual reality technologies are rapidly presenting viable alternatives to the real world for developing 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 [2] proposed a *Virtual Vision* approach to designing surveillance systems using a virtual train station environment populated by fully autonomous, lifelike virtual pedestrians that perform various activities (Figure 1). Within this environment, virtual cameras generate synthetic video feeds (Figure 2). 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.

Within the virtual vision paradigm, we propose a sensor network architecture capable of performing common visual surveillance tasks with minimal operator assistance. Once an operator monitoring surveillance video feeds spots a pedestrian involved in some suspicious activity, or a visual behavior analysis routine selects such a pedestrian automatically, the cameras decide amongst themselves how best to observe the subject. For example, a subset of cameras can collaboratively track the pedestrian as he weaves through the crowd. The problem of assigning cameras to follow pedestrians becomes challenging when multiple pedestrians are involved. To deal with the myriad of possibilities, the cameras must be able to *reason* about the dynamic situation. To this end, we propose a sensor network communication model whose nodes are capable of task dependent self-organization through local and global decision making. Each node is aware of its neighbors.[1]

Our proposed sensor network is novel insofar as it does

[1] The neighborhood of a node A can be defined automatically as the set of nodes that are, e.g., within nominal radio communications distance of A [4]. References [5, 6] present schemes to learn sensor network topologies.

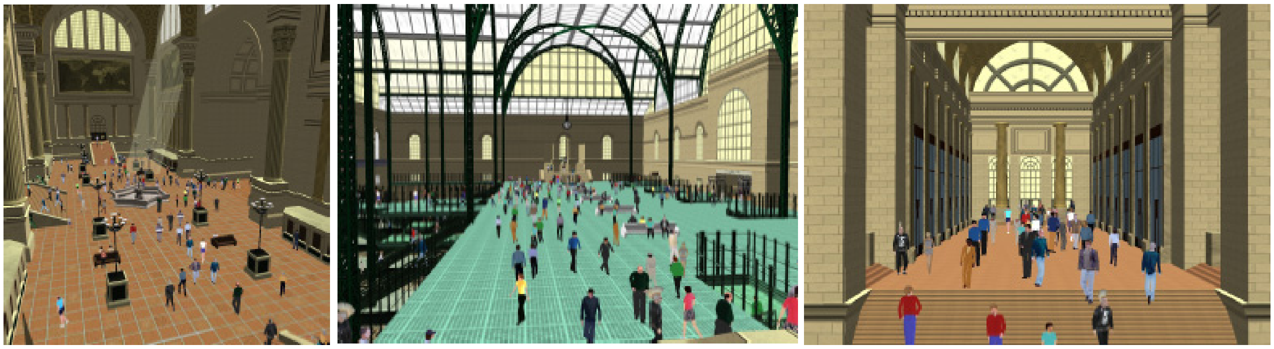


Figure 1: A large-scale virtual train station populated by self-animating virtual humans (from [3]).

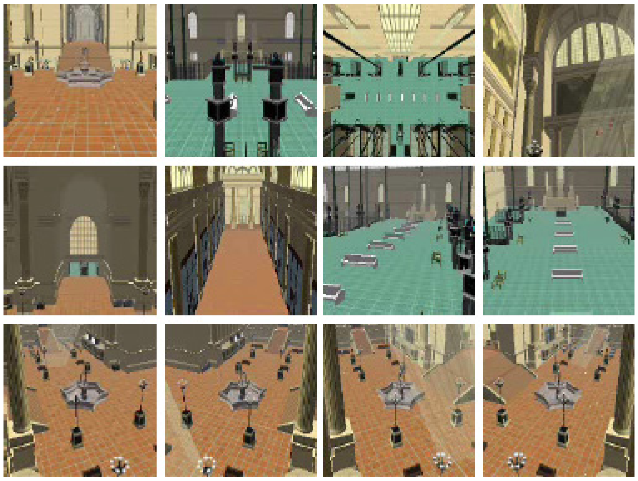


Figure 2: Virtual Vision. Synthetic video feeds from multiple virtual surveillance cameras situated in the (empty) Penn Station environment.

not require camera calibration, a detailed world model, or a central controller. The overall behavior of the network is the result of local decision making at each node and internode communication. Visual surveillance tasks are performed by groups of one or more camera nodes. These groups, which are created on the fly, define the information sharing parameters and the extent of collaboration between nodes. A group keeps evolving—i.e., old nodes leave and new nodes join the group—during the lifetime of the surveillance task. One node in each group acts as the group supervisor and is responsible for group level decision making.

Our sensor network is deployed and tested within the virtual train station simulator that was developed in [3]. 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 1000 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 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 (Figure 1).

In this paper, we first develop new image-based *fixate* and *zoom* algorithms for active cameras. Next, we propose a sensor network framework particularly suitable for designing camera networks for surveillance applications. Finally, we demonstrate the advantages of developing and evaluating this sensor network framework within our virtual world environment. The remainder of the paper is organized as follows: Section 2 covers relevant prior work on camera networks. We explain the low-level vision emulation in Section 3. In Section 4, we develop the behavior models for camera nodes. Section 5 introduces the sensor network communication model. In Section 6, we demonstrate the application of this model in the context of visual surveillance. We present our initial results in Section 7 and our conclusions and future research directions in Section 8.

2. Related Work

Previous work on camera networks has dealt with issues related to low and medium-level computer vision, namely identification, recognition, and tracking of moving objects [7]. The emphasis has been on model transference from one camera to another, which is required for object identification across multiple cameras [8]. Many researchers

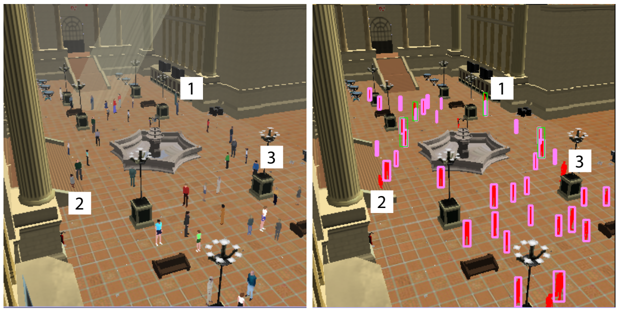


Figure 3: Pedestrian segmentation and tracking. (1) Multiple pedestrians are grouped together due to poor segmentation. (2) Noisy pedestrian segmentation results in a tracking failure. (3) Pedestrian segmentation and tracking failure due to occlusion.

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 stationary and actively controlled camera nodes [9, 10]. Our scheme does not require calibration; however, we assume that the cameras can uniquely identify a pedestrian with reasonable accuracy. To this end we employ color-based pedestrian appearance models.

Multiple cameras have also been employed either to increase the reliability of the tracking algorithm [11] (by overcoming the effects of occlusion or by using 3D information for tracking) or to track an object as it meanders through the fields of view (FOVs) of different cameras. In most cases, object tracking is accomplished by combining some sort of background subtraction strategy and an object appearance/motion model [12].

Little attention has been paid to the problem of controlling/scheduling active cameras to provide visual coverage of a large public area, such as a train station or an airport. [13] use a stationary wide-FOV camera to control an active tilt-zoom camera. The cameras are assumed to be calibrated and the total coverage of the cameras is restricted to the FOV of the stationary camera. [14] presents a scheme for scheduling available cameras in a task-dependent fashion. Here, the tasks are defined within a world model that consists of the ground plane, traffic pathways, and detailed building layouts. The scheduling problem is cast as a temporal logic problem that requires access to a central database consisting of current camera schedules and viewing parameters. This scheme is not scalable due to the central decision-making bottleneck.

3. Local Vision Routines

Each camera has its own suite of visual routines for pedestrian recognition, identification, and tracking, which we dub "Local Vision Routines" (LVRs). The LVRs are computer vision algorithms that directly operate upon the synthetic video generated by virtual cameras and the information readily available from the 3D virtual world. They mimic the performance of a state-of-the-art pedestrian recognition and tracking module. 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 and image vignetting. More sophisticated rendering schemes would address this limitation.

We employ appearance-based models to track pedestrians. Pedestrians are segmented to construct unique and robust color-based pedestrian signatures (appearance models), which are then matched across the subsequent frames. Pedestrian segmentation is carried out using 3D geometric information and background modeling/subtraction. The quality of 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 [12]. Color-based signatures are sensitive to illumination changes; however, this shortcoming can be mitigated by operating in HSV space instead of RGB space.

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. Appendix A provides additional algorithmic details regarding our pedestrian tracking approach.

The tracking module mimics the performance of a state-of-the-art tracking system (Figure 3). For example, it loses track due to occlusions, poor segmentation, or bad lighting. Tracking sometimes locks on the wrong pedestrian, especially if the 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.

4. Camera Behaviors

We treat each camera as a behavior-based autonomous agent. The overall behavior of the camera is determined by the LVR (bottom-up) and the current task (top-down). 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* (Figure 4).

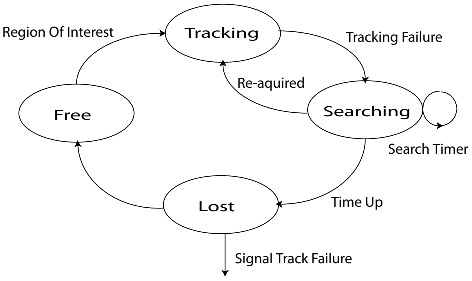


Figure 4: Top-level camera controller.

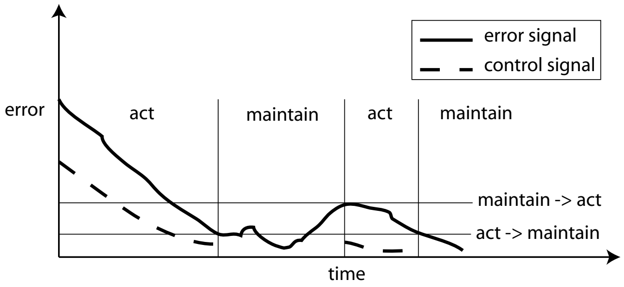


Figure 5: Dual-state controller for fixation and zooming.

Each camera can *fixate* and *zoom* in on an object of interest. 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 forces the PD controller to try continuously to minimize the error metric without ever succeeding, 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 altogether and does nothing (Figure 5).

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 (Figure 6, Row 1). 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 suspicious pedestrian (Figure 6, Row 2). The details of the *fixate* and *zoom* routines are given in Appendix A.

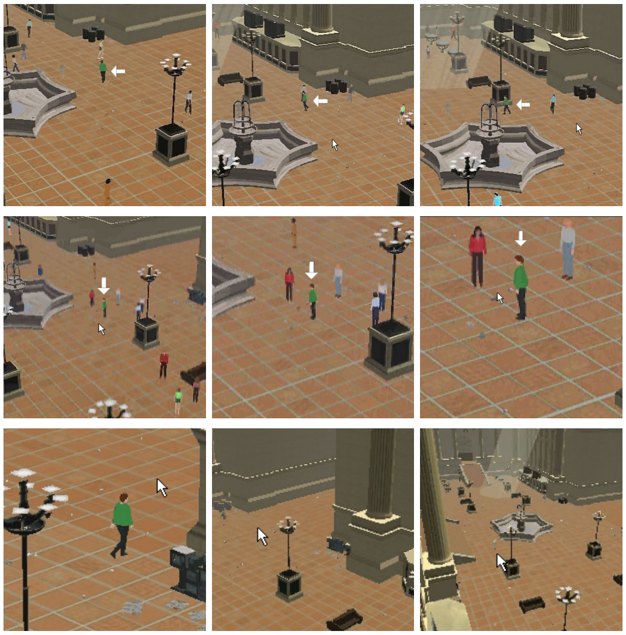


Figure 6: Row 1: A fixate sequence. Row 2: A zoom sequence. Row 3: Camera returns to its default settings upon losing the pedestrian; it is now ready for another task.

5. Sensor Network Model

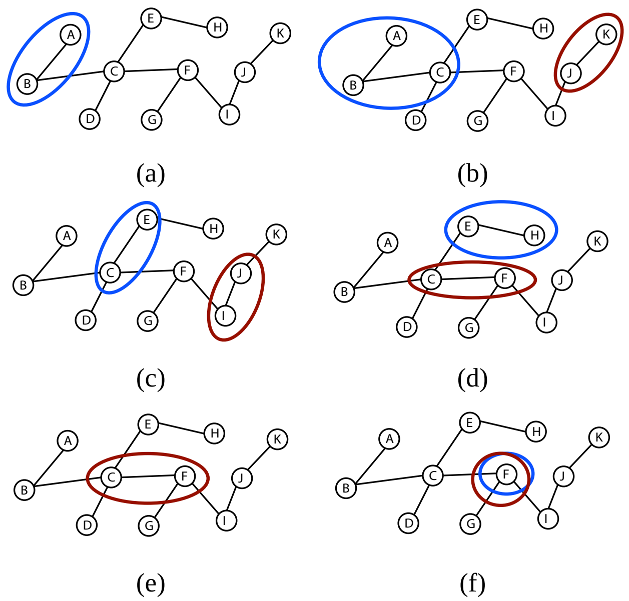
We now explain the sensor network communication scheme that allows task-specific node organization. The idea is as follows: A human operator presents a particular sensing request to one of the nodes. In response to this request, relevant nodes self-organize into a group with the aim of fulfilling the sensing task. The group, which formalizes the collaboration between member nodes, is a dynamic arrangement that keeps evolving throughout the lifetime of the task. At any given time, multiple groups might be active, each performing its respective task.

The following procedure sets up the task-specific node groups.

Task-specific group formation & evolution (Figure 7)

```
Let $N$ be the sets of all nodes
Node $n_i$ receives a query
Node $n_i$ sets up a *named* task and broadcasts it to other nodes
    $N' = N - \{n_i\}$[2]
A subset $R$ of nodes $N'$ respond by sending their relevance
    for the task {*Local decision making*}
Node $n_i$ ranks the nodes $R$ using the information returned by
    these nodes, chooses a subset $G$ of $R$, and forms the group
    $G \cup \{n_i\}$, which is responsible for this task
Node $n_i$ is the designated supervisor of the group
**while** The task is active **do**
```

[2] In the context of computer vision, it is sufficient to broadcast the task to the neighboring nodes.



(a) (b) (c) (d) (e) (f)

Figure 7: (a) Group 1: A and B; possible candidate: C (b) Group 1: A, B, and C; possible candidate: E, F, and D. Group 2: J and K; possible candidate: I. (c) Group 1: E and C; possible candidate: H, F, D, and B. Group 2: J and I; possible candidate: F. (d) Group 1: E and E; possible candidate C. Group 2: C and F; possible candidate: B, D, G, I, and E. (e) Group 1 and 2 require the same resources, so Group 1 vanished; task failure. (f) A unique situation where both groups successfully use the same nodes, e.g., imagine two groups tracking two pedestrians that started walking together.

```
    $n_i$ monitors the group and decides when to add/remove a
    node and when to designate another node to act as the su-
    pervisor {Group level or global decision making}
end while
```

Group formation is determined by the local computation at each node and the communication between the nodes. We require each node to compute its relevance to a task in the same currency. Our approach draws inspiration from behavior-based autonomous agents where the popularly held belief is that, rather than being the result of some powerful central processing facility, the overall intelligent behavior is a consequence of the interaction between many simple processes, called behaviors. We leverage the interaction between the individual nodes to generate global task-directed behavior.

Node failures are handled by simply dropping the nodes that have not communicated with the group in the previous x seconds. We have not yet implemented a supervisor node failure recovery strategy; however, we propose to replicate the supervisor's *mental state* in every other node of the group, which is straightforward when the nodes are identical. That would allow another node to take over when the supervisor fails to communicate with the group in the previous x seconds.

6. Video Surveillance

We now consider how a sensor network of dynamic video cameras might be used in the context of surveillance. A human operator spots a suspicious pedestrian(s) in one of the video feeds and, for example, requests the network to "track this pedestrian," "zoom in on this pedestrian," or "track the entire group." The successful execution and completion of these tasks requires intelligent allocation and scheduling of the available cameras; in particular, the network must decide which cameras should track the pedestrian and for how long.

A detailed world model that includes the location of cameras, their FOVs, pedestrian motion prediction models, occlusion models, and pedestrian movement pathways might allow (in some sense) *optimal* allocation and scheduling of cameras; however, it is cumbersome and in most cases infeasible to acquire such a world model. Our approach does not require such a knowledge base. We only assume that a pedestrian can be identified by different cameras with reasonable accuracy and that camera network topology is known *a priori*. A direct consequence of this approach is that the network can be easily modified through removal, addition, or replacement of camera nodes.

6.1. Computing Camera Node Relevance

The accuracy with which individual camera nodes are able to compute their relevance to the task at hand determines the overall performance of the network. Our scheme for computing the relevance of a camera to a video surveillance task encodes the intuitive observations that 1) a camera that is currently free should be chosen for the task, 2) a camera with better tracking performance with respect to the task should be chosen, 3) the turn and zoom limits of cameras should be taken into account when assigning a camera to a task; i.e., a camera that has more leeway in terms of turning and zooming might be able to follow a pedestrian for a longer time, and 4) it is better to avoid unnecessary reassignments of cameras to different tasks, as that might degrade the performance of the underlying computer vision routines.

Upon receiving a task request, a camera node returns a relevance metric—a list of attribute-value pairs describing relevance to the current task across multiple dimensions (Table 1)—to the supervisor node, which converts this metric into a scalar relevance value r using the following equation:

$$r = \begin{cases} \exp\left(-\frac{(c-1)^2}{2{\sigma_c}^2} - \frac{(\theta-\hat{\theta})^2}{2{\sigma_\theta}^2} - \frac{(\alpha-\hat{\alpha})^2}{2{\sigma_\alpha}^2} - \frac{(\beta-\hat{\beta})^2}{2{\sigma_\beta}^2}\right) & \\ \qquad\qquad\qquad\qquad\qquad \text{when } s = \text{free} & \\ t \quad \text{when } s = \text{busy} & \end{cases} \tag{1}$$

where $\hat{\theta} = (\theta_{\text{min}} + \theta_{\text{max}})/2$, $\hat{\alpha} = (\alpha_{\text{min}} + \alpha_{\text{max}})/2$, and $\hat{\beta} = (\beta_{\text{min}} + \beta_{\text{max}})/2$, and where θ_{min} and θ_{max} are extremal

status	=	$s \subset \{\text{busy}, \text{free}\}$
quality	=	$c \in [0, 1]$
fov	=	$\theta \in [\theta_{\text{min}}, \theta_{\text{max}}]$ degrees
x_turn	=	$\alpha \in [\alpha_{\text{min}}, \alpha_{\text{max}}]$ degrees
y_turn	=	$\beta \in [\beta_{\text{min}}, \beta_{\text{max}}]$ degrees
time	=	$t \in [0, \infty)$ seconds
task	=	$a \subset \{a_i \mid i = 1, 2, \cdots\}$

Table 1: The relevance metric returned by a camera node relative to a new task request. The supervisor node converts the metric into a scalar value representing the relevance of the node for the particular surveillance task.

field of view settings, α_{min} and α_{max} are extremal rotation angles around the x-axis (up-down), and β_{min} and β_{max} are extremal rotation angles around the y-axis (left-right). The values of the variances σ_c, σ_θ, σ_α, and σ_β associated with each attribute are chosen empirically (for our experiments, we assigned $\sigma_\theta = \sigma_\alpha = \sigma_\beta = 5.0$ and $\sigma_c = 0.2$).

The computed relevance values are used by a greedy algorithm to assign cameras to various tasks. The supervisor node gives preference to the nodes that are currently free, so the nodes that are part of another group are selected only when the required number of free nodes are unavailable for the current task. We are investigating more formal schemes for reasoning about camera assignment. Also, we have not yet fully resolved the implications of group-group interactions (Figure 7).

6.2. Surveillance Tasks

We have implemented an interface that presents the operator a display of the synthetic video feeds from multiple virtual surveillance cameras (c.f., Figure 2). The operator can select a person in any video feed and instruct the camera network to perform one of the following tasks: 1) follow the person, 2) capture a high-resolution snapshot, or 3) zoom-in and follow the person. The network then automatically assigns cameras to fulfill the task requirements. The operator can also initiate multiple tasks, in which case either cameras that are not currently occupied are chosen for the new task or some cameras are reassigned to the new task. The task outline is as follows:

Task Outline

1: Operator selects a pedestrian in camera c and specifies one of the following tasks: follow, capture snapshot, follow and zoom.
2: c computes color-based pedestrian signature
3: c creates an appropriate task and broadcasts it to the adjacent cameras N_c
4: Each node in N_c computes its relevance to the task proposed by c
5: c ranks nodes in N_c according to their relevance
6: c utilizes the ranking to assign a subset G_c of N_c to the task at hand. $G_t = G_c \cup \{c\}$ is the task-specific group
7: c sends the pedestrian signature to every camera in G_c along with any information that might be helpful in positively identifying the pedestrian in question
8: Each node in G_c uses its visual and behavior routines to locate and track the pedestrian {*Each node has a suite of search routines to help re-acquire a pedestrian after short-duration tracking losses.*}
9: Each node in G_t also polls its neighborhood for possible addition to the group
10: Each node in G_t continuously updates its relevance for the purpose of supervisor selection and node removal

7. Results

To date, we have tested our visual sensor network system with up to 4 pan-tilt-zoom cameras, one at each corner of the main waiting room of the virtual train station. Our initial results are promising. The sensor network correctly assigned cameras in most cases. Some of the problems that we encountered are related to pedestrian identification and tracking. As we increase the number of virtual pedestrians in the train station, the identification and tracking module has difficulty following the correct pedestrian, so the surveillance task fails (and the cameras just return to their default settings).

Figure 8 illustrates a “follow” task sequence. An operator selects the pedestrian with the green shirt in Camera 1 (top row). Camera 1 forms a group with Camera 2 (bottom row) to follow and zoom in on the pedestrian. At some point, Camera 2 loses the pedestrian (due to occlusion), and it invokes a search routine, but it fails to reacquire the pedestrian. Camera 1, however, is still tracking the pedestrian. Camera 2 leaves the group and returns to its default settings.

8. Conclusions and Future Work

We envision future surveillance systems to be networks of stationary and active cameras capable of providing perceptive coverage of extended 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. The work presented in this paper is a step toward the realization of new sensor networks.

The overall behavior of our network is governed by local decision making at each node and communication between the nodes. Our approach is new insofar as it does not require camera calibration, a detailed world model, or a central controller. We have intentionally avoided multi-camera tracking schemes that assume prior camera network calibration which, we believe, is an unrealistic goal for a large-scale camera network consisting of heterogeneous cameras. Similarly, our approach does not expect a detailed world model which, in general, is hard to acquire. Since it lacks any cen-

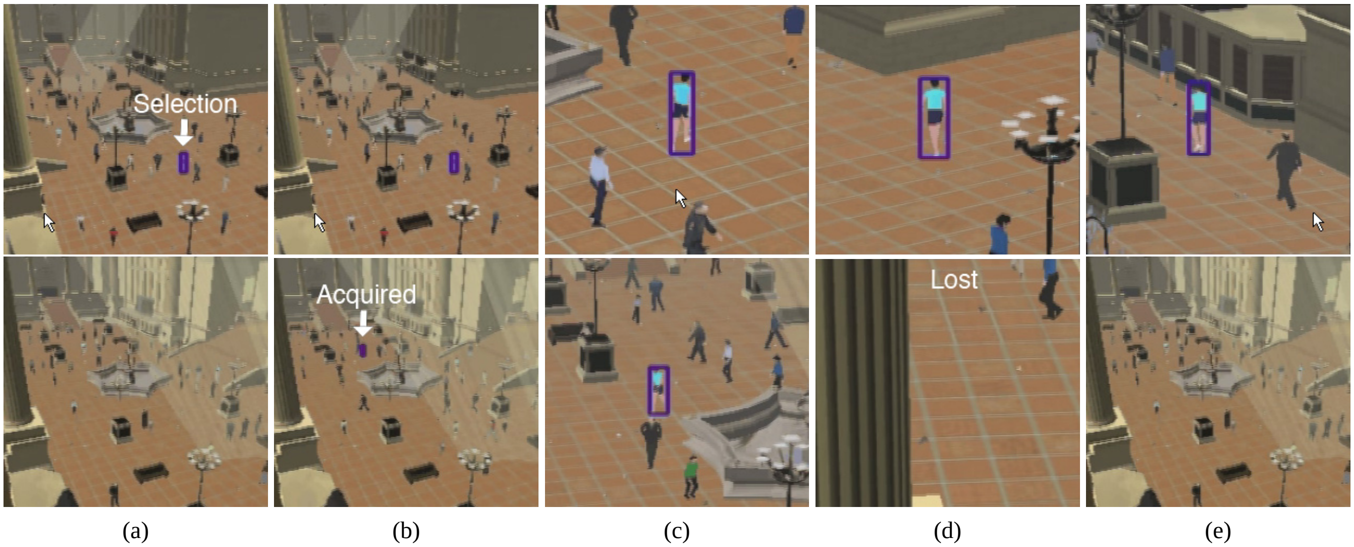


Figure 8: “Follow” sequence. (a) The operator selects a pedestrian in Camera 1 (upper row). (b) and (c) Camera 1 and Camera 2 (lower row) are tracking the pedestrian. (d) Camera 2 loses track. (e) Camera 1 is still tracking; Camera 2 has returned to its default settings.

tral controller, we expect the proposed approach to be robust and scalable.

We have developed and demonstrated our surveillance system in a virtual train station environment populated by autonomous, lifelike pedestrians, and our initial results appear promising. Our simulator should continue to facilitate our ability to design such large-scale networks and experiment with them on commodity personal computers.

We are currently pursuing a *Cognitive Modeling* [15, 16] approach to node organization and camera scheduling [17]. We are also investigating scalability and node failure issues. Moreover, we are constructing more elaborate scenarios involving multiple cameras situated in different locations within the train station, with which we would like to study the performance of the network when it is required to follow multiple pedestrians during their entire stay in the train station.

A. Algorithmic Details

Pedestrian Tracking

```
Require: A color-based signature $h$ (histogram) of the pedestrian
    to be tracked
Render: image $I$
Perform pedestrian segmentation on this frame and compute
    2D bounding boxes for all visible pedestrians
for all bounding boxes do
    Compute pedestrian's color histogram $h_i$ using $I$ masked
        with the respective bounding box
    Compare $h_i$ with $h$ and store the result
end for
Pick the pedestrian with the highest match score
Update stored signature $h$ when the conditions differ sufficiently
    from those when $h$ was last computed and stored.[3]
```

Fixate Algorithm

```
Let $m$ and $n$ be the dimensions of the video image in pixels
Let $(l, b)$ and $(r, t)$ be the left-bottom and right-top corners of
    the rectangle defining the ROI
Let $f_x$ and $f_y$ be the camera's FOV settings (in degrees) along
    the image's $x$ and $y$ axes, respectively
Center of the image, $\mathbf{c_I} = \left(\frac{m}{2}, \frac{n}{2}\right)$
Center of the ROI, $\mathbf{c_{ROI}} = \left(\frac{l+r}{2}, \frac{b+t}{2}\right)$
Define rectangle $r_1$ with corners $\mathbf{c_I} \mp 0.025 * (m, n)$
Define rectangle $r_2$ with corners $\mathbf{c_I} \mp 0.225 * (m, n)$
if ROI is enclosed within $r_1$ then
    Set state = Maintain
end if
if ROI is not enclosed within $r_2$ then
    Set state = Turn
end if
Error, $(e_x, e_y) = \mathbf{c_I} - \mathbf{c_{ROI}}$
if state is Maintain then
    Control-signal = $(0, 0)$ {rotation about local $x$ and $y$ axes}
else
    Control-signal = $k \left( \langle e_x \rangle \frac{f_x}{m}, \langle e_x \rangle \frac{f_y}{n} \right)$
end if
```

Zoom Algorithm

```
Let $m$ and $n$ be the dimensions of the video image in pixels
Let $(l, b)$ and $(r, t)$ be the left-bottom and right-top corners of
    the rectangle defining the ROI
```

[3] E.g., *zooming* can drastically change the appearance of a pedestrian, thereby confounding traditional appearance-based schemes such as color histogram signatures.

```
Let $f_x$ and $f_y$ be the camera's FOV settings (in *degrees*) along the image's $x$ and $y$ axes, respectively
Desired coverage, $d_r \in [0, 1]$, where 1 means that ROI should cover the entire image.
Let $r$ is the smallest rectangle that encloses the ROI and whose aspect ratio is $\frac{n}{m}$, and let $\text{area}_r = \frac{\text{area of } r}{mn}$
Error, $e = \text{area}_r - d_r$
**if** $\langle e \rangle < 0.01$ **then**
    Set state = *Maintain*
**end if**
**if** $\langle e \rangle > 0.03$ **then**
    Set state = *Act*
**end if**
**if** state is *Maintain* **then**
    Control-signal = 0
**else**
    **if** $e > 0$ **then**
        Control-signal = $\min(1.0, k\langle e \rangle)$
    **else**
        Define rectangle $r_{\text{valid}}$ with corners $(\frac{m}{2}, \frac{n}{2}) \mp 0.45 * (m, n)$
        **if** $r$ is not enclosed within $r_{\text{valid}}$ **then**
            Control-signal = 0
        **end if**$r_{\text{valid}}$ and
        Let $(v_x, v_y)$ be the corner of $r$ that is farthest from the point $\left(\frac{m}{2}, \frac{n}{2}\right)$
        Let $s = \begin{cases} 2\frac{v_x}{m} & \text{if } v_x \geq v_y \\ 2\frac{v_y}{n} & \text{if } v_x < v_y \end{cases}$
        Let $s' = 1 - \frac{s^2}{0.4+s^2}$
        Control-signal = $-\min\left(1.0, k\left(1 - \frac{s^2}{0.4+s^2}\right)\langle e \rangle\right)$
    **end if**
**end if**
```

Acknowledgements

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 of DARPA for his generous support and encouragement. We also thank Wei Shao and Mauricio Plaza-Villegas for their invaluable contributions to the implementation of the Penn Station simulator. Mauricio Plaza-Villegas implemented a prototype virtual vision infrastructure within the simulator.

References

[1] D. Terzopoulos and T. Rabie, "Animat vision: Active vision in artificial animals," *Videre: Journal of Computer Vision Research*, vol. 1, pp. 2–19, September 1997.

[2] D. Terzopoulos, "Perceptive agents and systems in virtual reality," in *Proc. 10th ACM Symposium on Virtual Reality Software and Technology*, (Osaka, Japan), pp. 1–3, October 2003.

[3] W. Shao and D. Terzopoulos, "Autonomous pedestrians," in *Proc. ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, (Los Angeles, CA), pp. 19–28, July 2005.

[4] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 2–16, February 2003.

[5] A. Ihler, J. Fisher, R. Moses, and A. Willsky, "Nonparametric belief propagation for self-calibration in sensor networks," in *Proc. Third International Symposium on Information Processing in Sensor Networks*, (Berkeley, CA), pp. 225–233, April 2004.

[6] D. Marinakis, G. Dudek, and D. Fleet, "Learning sensor network topology through Monte Carlo expectation maximization," in *Proc. IEEE Intl. Conf. on Robotics and Automation*, (Barcelona, Spain), April 2005.

[7] R. Collins, O. Amidi, and T. Kanade, "An active camera system for acquiring multi-view video," in *Proc. International Conference on Image Processing*, (Rochester, NY), pp. 517–520, September 2002.

[8] S. Khan and M. Shah, "Consistent labeling of tracked objects in multiple cameras with overlapping fields of view," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1355–1360, October 2003.

[9] F. Pedersini, A. Sarti, and S. Tubaro, "Accurate and simple geometric calibration of multi-camera systems," *Signal Processing*, vol. 77, no. 3, pp. 309–334, 1999.

[10] T. Gandhi and M. M. Trivedi, "Calibration of a reconfigurable array of omnidirectional cameras using a moving person," in *Proc. 2nd ACM International Workshop on Video Surveillance and Sensor Networks*, (New York, NY), pp. 12–19, ACM Press, 2004.

[11] J. Kang, I. Cohen, and G. Medioni, "Multi-views tracking within and across uncalibrated camera streams," in *Proc. First ACM SIGMM International Workshop on Video Surveillance*, (New York, NY), pp. 21–33, ACM Press, 2003.

[12] N. T. Siebel, *Designing and Implementing People Tracking Applications for Automated Visual Surveillance*. PhD thesis, Dept. of Computer Science. The University of Reading., UK, March 2003.

[13] R. Collins, A. Lipton, H. Fujiyoshi, and T. Kanade, "Algorithms for cooperative multisensor surveillance," *Proceedings of the IEEE*, vol. 89, pp. 1456–1477, October 2001.

[14] C. J. Costello, C. P. Diehl, A. Banerjee, and H. Fisher, "Scheduling an active camera to observe people," in *Proc. 2nd ACM International Workshop on Video Surveillance and Sensor Networks*, (New York, NY), pp. 39–45, ACM Press, 2004.

[15] J. Funge, X. Tu, and D. Terzopoulos, "Cognitive modeling: Knowledge, reasoning and planning for intelligent characters," in *Proc. ACM SIGGRAPH 99* (A. Rockwood, ed.), pp. 29–38, Aug. 1999.

[16] F. Qureshi, D. Terzopoulos, and P. Jaseiobedzki, "Cognitive vision for autonomous satellite rendezvous and docking," in *Proc. IAPR Conference on Machine Vision Applications*, (Tsukuba Science City, Japan), pp. 314–319, May 2005.

[17] F. Qureshi and D. Terzopoulos, "Surveillance camera scheduling: A virtual vision approach," in *Proc. Third ACM International Workshop on Video Surveillance and Sensor Networks*, (Singapore), November 2005. In press.