Virtual Vision and Smart Camera Networks

Faisal Qureshi
Department of Computer Science
University of Toronto, Toronto, ON, M6K3N1, Canada
faisal@cs.toronto.edu

Demetri Terzopoulos
Computer Science Department
University of California, Los Angeles, CA 90095, USA
dt@cs.ucla.edu

Abstract

This paper presents camera network research in the context of a unique synthesis of advanced computer graphics and vision simulation technologies. In particular, we propose the design of and experimentation with simulated camera networks within visually and behaviorally realistic virtual environments. Specifically, we demonstrate a smart camera network comprising static and active simulated video surveillance cameras that provides perceptive coverage of a large virtual public space, a train station populated by autonomously self-animating virtual pedestrians. The readily reconfigurable virtual cameras generate synthetic video feeds that emulate those generated by real surveillance cameras monitoring public spaces. With minimal reliance on a human operator, our new camera network control strategy enables the collection of cameras to collaborate in performing various visual surveillance tasks, such as closely monitoring a pedestrian as (s)he locomotes in and out of the fields of view of individual cameras. The camera network supports task-dependent node selection and aggregation through local decision-making and inter-node communication. We propose a solution to the node selection problem that transforms it into an equivalent constraint satisfaction problem. Our technique is scalable and robust to node failures, as it lacks any central controller. Given the challenges of carrying out such research on a large scale in the real world, our simulation approach, which runs on a high-end commodity PC, has proven to be beneficial for development and experimentation.

1 Introduction

Effective visual coverage of large public spaces, such as a train station or an airport, requires multiple cameras to work together towards common sensing goals. 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 closely monitoring a pedestrian of interest as (s)he meanders through the field-of-view (FOV) of multiple cameras, or zooming in on a particular subject to acquire one or more facial snapshots. Therefore, it is desirable to design camera networks that are capable of performing visual surveillance tasks autonomously, or at least with minimal human intervention. In this paper, we demonstrate a camera network model comprising *uncalibrated* static and active simulated video surveillance cameras that provide perceptive coverage of a large virtual public space—a train station populated by autonomously self-animating virtual pedestrians (Fig. 1)—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 the active pan/tilt/zoom (PTZ) cameras can collaboratively track the pedestrian as (s)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 distributed camera network control strategy that is capable of dynamic task-driven node aggregation through local decision making and inter-node communication.

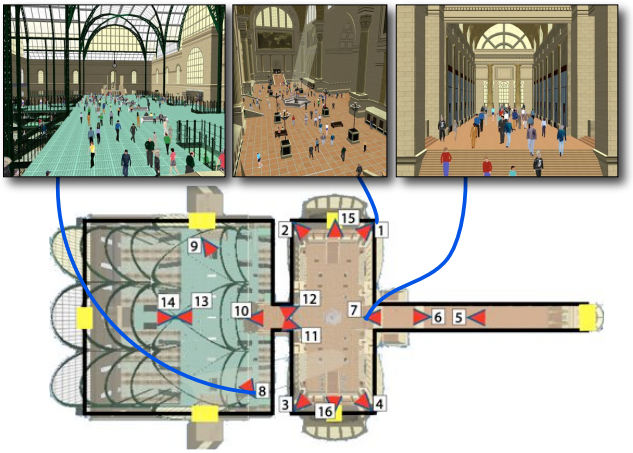

Figure 1: Plan view of the virtual Penn Station environment with the roof not rendered, revealing the concourses and train tracks (left), the main waiting room (center), and the long shopping arcade (right). (The yellow rectangles indicate station pedestrian portals.) An example camera network is shown comprising 16 simulated active (pan-tilt-zoom) video surveillance cameras. Synthetic images from simulated video surveillance cameras 1, 2, and 8 (from [1]).

1.1 Virtual Vision

This type of research would be difficult to carry out in the real world given the cost of deploying and experimenting with an appropriately complex camera network in a large public space the size of a train station. By contrast, the multiple *virtual* cameras, which generate synthetic video feeds that emulate those generated by real surveillance cameras monitoring public spaces, are very easily reconfigurable in the virtual space (Fig. 1). Moreover, the virtual world provides readily accessible ground-truth data for the purposes of camera network algorithm validation. Despite its sophistication, our virtual vision simulator runs on high-end commodity PCs, obviating the need to grapple with special-purpose hardware and software (Fig. 2).

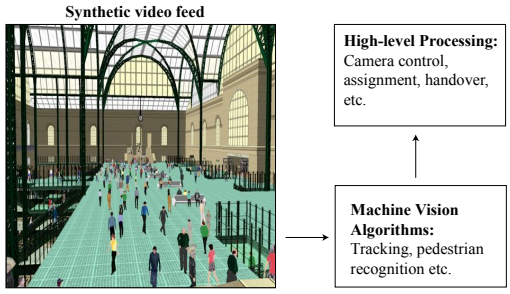

Figure 2: Virtual vision paradigm (image from [1]).

1.2 Smart Camera Network

Many of the characteristics and challenges associated with sensor networks are relevant to the work presented here. A fundamental issue in sensor networks is the selection of sensor nodes that participate in a particular sensing task [2]. The selection process must take into account the information contribution of each node against its resource consumption or potential utility in other uses. Distributed approaches for node selection are preferable to centralized approaches that compromise what are perhaps the greatest advantages of networked sensing—robustness and scalability. Also, in a typical sensor network, each sensor node has local autonomy and can communicate with a small number of neighboring nodes that are within radio communication range. Message delay and message loss are a common occurrence in sensor networks due to bandwidth limitations, interference, etc. One must also account for non-stationary network topology due to node failures, node additions, etc.

Mindful of these issues, we propose a novel camera network control strategy that does not require camera calibration, a detailed world model, or a central controller. The overall behavior of the network is the consequence of the local processing at each node and inter-node communication. The network is robust to node and communication link failures; moreover, it is scalable due to the lack of a central controller. 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 the group and new nodes join it—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. We also present a novel constraint satisfaction problem formulation for resolving group-group interactions.

The work presented here extends that which we presented in an earlier paper [3]. Here, we introduce a novel constraint satisfaction problem formulation for resolving group-group interactions that we identified in our previous paper.

2 Related Work

As was argued in [4, 5], computer graphics and virtual reality technologies are rapidly presenting viable alternatives to the real world for developing sensory systems (see also [6]). Our camera network is deployed and tested within the virtual train station simulator that was developed in [1]. 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. Standard computer graphics techniques enable a photorealistic rendering of the busy urban scene with considerable geometric and photometric detail (Fig. 1).

The problem of forming sensor groups based on task requirements and resource availability has received much attention within the sensor networks community [2]. Reference [7] argues that task-based grouping in *adhoc* camera networks is highly advantageous. Collaborative tracking, which subsumes the above issue, is considered an essential capability in many sensor networks [2]. Reference [8] introduces an information driven approach to collaborative tracking, which attempts to minimize the energy expenditure at each node by reducing inter-node communication. A node selects the next node by utilizing the information gain vs. energy expenditure tradeoff estimates for its neighbor nodes. In the context of camera networks, it is often difficult for a camera node to estimate the expected information gain by assigning another camera to the task without explicit geometric and camera-calibration knowledge, and such knowledge is tedious to obtain and maintain during the lifetime of the camera network. Therefore, our camera networks do without such knowledge; a node needs to communicate with nearby nodes before selecting new nodes.

Nodes comprising sensor networks are usually untethered sensing units with limited onboard power reserves. Hence, a crucial concern in sensor networks is the energy expenditure at each node, which determines the lifespan of a sensor network [9]. Node communications have large power requirements; therefore, sensor network control strategies attempt to minimize the inter-node communication [8]. Presently, we do not address this issue; however, the communication protocol proposed here limits the communication to the active nodes and their neighbors.

Little attention has been paid in computer vision to the problem of controlling active cameras to provide visual coverage of a large public area, such as a train station or an airport [10]. Previous work on camera networks in computer vision has dealt with issues related to low-level and mid-level computer vision, namely segmentation, tracking, and identification of moving objects [11], and camera network calibration [12]. Our approach does not require calibration; however, we assume that the cameras can identify a pedestrian with reasonable accuracy. To this end we employ color-based pedestrian appearance models.

3 Camera Nodes

Each virtual camera node is an autonomous agent capable of communicating with nearby nodes. The Low-level Visual processing Routines (LVR), such as pedestrian tracking and identification, determine the sensing capabilities of a camera node. An augmented hierarchical finite state machine implements the top-level controller of the camera node (Fig. 3). The camera controller is responsible for the overall behavior of the camera node and it takes into account the information gathered through visual analysis (bottom-up) and the current task (top-down). The LVRs are computer vision algorithms that directly operate upon the synthetic video generated by the virtual cameras as well as the information readily available from the 3D virtual world. They mimic the performance of a state-of-the-art pedestrian segmentation and tracking module. Each camera can *fixate* 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 [13]. See [3] for the details.

In its default state, `Idle`, the camera node is not involved in any task. A camera node transitions into the `Computing Relevance` state upon receiving a `query_relevance` message from a nearby node. Using the description of the task that is contained within the `query_relevance` message, and by employing the LVRs, the camera node can compute its relevance to the task. For example, a camera can use visual search to find a pedestrian that matches

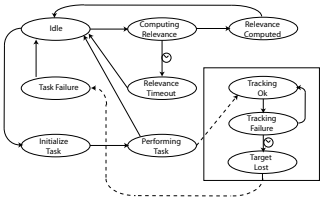

Figure 3: Top-level camera controller.

the appearance-based signature passed by the querying node. The relevance encodes the expectation of how successful a camera node will be at a particular sensing task. The camera returns to the `Idle` state if it fails to compute the relevance due to the fact that it cannot find a pedestrian matching the description. On the other hand, when the camera successfully finds the desired pedestrian, it returns the relevance value to the querying node. The querying node passes the relevance value to the supervisor node of the group, which decides whether or not to include the camera node in the group. The camera goes into the `Performing Task` state upon joining a group, where the embedded child finite state machine hides the sensing details from the top-level controller and enables the node to handle short-duration sensing (tracking) failures. The camera node goes into the `Task Failure` state if it does not recover from the sensing failure within the set time limit. Since an expected message might never arrive due to a communication error or node failure, all states other than the `Performing Task` state have built-in timers (not shown in Fig. 3) that allow the camera node to transition into the default state rather than wait indefinitely for a message from another node.

4 Camera Network Model

The camera network communication scheme that enables task-specific node organization 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 evolves throughout the lifetime of the task. At any given time, multiple groups might be active, each performing its respective task. Group formation is determined by the local computation at each node and the communication between the nodes. Each node computes its relevance to a task in the same currency. Our approach is inspired by behavior-based autonomous agents whose overall intelligent behavior is a consequence of the interaction between many simple processes, called behaviors, rather than being the result of some powerful central processing facility. We leverage the interaction between the individual nodes to generate global, task-directed behavior.

From the standpoint of user interaction, we have identified two kinds of sensing queries: 1) where the queried camera itself can measure the phenomenon of interest—e.g., when a human operator selects a pedestrian to be tracked within a particular video feed—and 2) when the queried node might not be able to perform the required sensing and needs to route the query to other nodes. For instance, an operator can request the network to count the number of pedestrians wearing green shirts. Currently, the network supports only the first kind of queries, which suffice for setting up collaborative tracking tasks.

4.1 Node Grouping

Node grouping commences when a node, n, receives a sensing query. In response to the query, the node sets up a named task and creates a single-node group. Initially, node n is chosen as the supervisor node. To recruit new nodes for the current task, node n begins by sending `query_relevance` messages to its neighboring nodes,

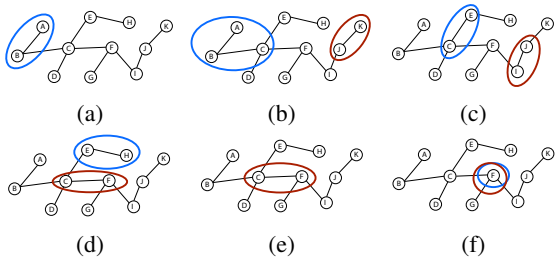

Figure 4: Grouping and conflict resolution. (a) Group 1: A and B; possible candidate: C (b) Group 1: A, B, and C; possible candidates: E, F, and D. Group 2: J and K; possible candidate: I. (c) Group 1: E and C; possible candidates: H, F, D, and B. Group 2: J and I; possible candidate: F. (d) Group 1: E and H; possible candidate C. Group 2: C and F; possible candidates: 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., two groups tracking two pedestrians that started walking together.

N_n. A subset N' of N_n respond by sending their relevance values for the current task. Upon receiving the relevance values, node n selects a subset M of N' to include in the group, and sends `join` messages to the chosen nodes. When there is no resource contention between groups (tasks)—e.g., when only one task is active, or when multiple tasks that do not rely upon the same nodes for successful operation are active—the selection process is relatively straightforward; node n picks those nodes from N' that have higher relevance values. On the other hand, a conflict resolution mechanism is required when multiple groups vie for the same nodes (the next section describes a scheme to handle this situation). A node that is not already part of any group can join the group upon receiving a `join` message from the supervisor of that group. After receiving the `join` message, a subset M' of M elect to join the group and respond with a `join_ok` message.

For groups of more than one node, every group member sends out `query_relevance` messages to the nearby nodes that are not already part of the group. The supervisor node is responsible for group-level decisions, so member nodes forward to the group supervisor all the group-related messages, such as the `relevance` messages from candidates for group membership. The supervisor node sends "heartbeat" messages to other member nodes, which respond with their current relevance values, at regular intervals. The supervisor node uses the most recent relevance values to decide when to drop a member node. When a supervisor detects that its own relevance value for the current task is below the predefined threshold, it selects a new supervisor from amongst the member nodes. The group vanishes when the last node leaves the group.

4.2 Conflict Resolution

A conflict resolution mechanism is needed when multiple groups require the same resources (Fig. 4(e-f)). The problem of assigning cameras to the contending groups can be treated as a Constraint Satisfaction Problem (CSP) [14]. Formally, a CSP consists of a set of variables $\{v_1, v_2, v_3, \cdots, v_k\}$, a set of allowed values $\mathrm{Dom}[v_i]$ for each variable v_i (called the domain of v_i), and a set of constraints $\{C_1, C_2, C_3, \cdots, C_m\}$. The solution to the CSP is a set $\{v_i \leftarrow a_i \,|\, a_i \in \mathrm{Dom}[v_i]\}$, where the a_is satisfy all the constraints.

We treat each group g as a variable, whose domain consists of the non-empty subsets of the set of cameras with relevance values (w.r.t. g) greater than a predefined threshold. The constraints restrict the assignment of a camera to multiple groups. We define a constraint C_{ij} as $a_i \cap a_j = \{\Phi\}$, where a_i and a_j are camera assignments to groups g_i and g_j, respectively; k groups give rise to $k!/2!(k-2)!$ constraints.

We can then define a CSP problem $P = (G, D, C)$, where $G =$

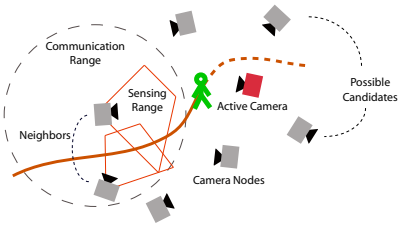

Figure 5: A camera network for video surveillance consists of camera nodes that can communicate with other nearby nodes. Collaborative tracking requires that cameras organize themselves to perform camera handover when the tracked subject moves out of the sensing range of one camera and into that of another.

$\{g_1, g_2, \cdots, g_k\}$ is the set of groups (variables) with non-empty domains, $S = \{\text{Dom}[g_i] \,|\, i \in [1,k]\}$ is the set of domains for each group, and $C = \{C_{ij} \,|\, i, j \in [1,k], i \neq j\}$ is the set of constraints. To solve P, we employ *backtracking* to search systematically through the space of possibilities. We find all solutions, rank these solutions according to the relevance values for cameras (w.r.t. the each group), and select the best solution to find the optimal assignments. When P does not have a solution, we recursively solve smaller CSP problems $P' = (G', D', C')$, where $G' \subset G$ and D' and C' are defined accordingly, until we have found a solution to a smaller problem P'.

A node initiates the conflict resolution procedure upon identifying a group-group conflict; e.g., when it intercepts a `query_relevance` message from multiple groups, or when it already belongs to a group and it receives a `query_relevance` message from another group. The conflict resolution procedure begins by *centralizing* the CSP in one of the supervisor nodes that uses *backtracking* to solve the problem. The result is then conveyed to the other supervisor nodes.

CSPs have been studied extensively in the computer science literature and there exist more powerful variants of the basic backtracking method; however, we employ the basic backtracking approach in the interest of simplicity, and it can easily cope with the size of problems encountered in the current setting. A key feature of the conflict resolution scheme proposed here is centralization, which requires that all the relevant information is gathered at the node that is responsible for solving the CSP. For smaller CSPs, the cost of centralization is easily offset by the speed and ease of solving the CSP. One can perhaps avoid centralization by using a scheme for distributed CSPs [15].

4.3 Node Failures and Communication Errors

The proposed communication model takes into consideration node and communication failures. Communication failures are perceived as camera failures; for example, when a node is expecting a message from another node, and the message never arrives, the first node concludes that the second node is malfunctioning. A node failure is assumed when the supervisor node does not receive the node's response to successive heartbeat messages, and the supervisor node removes the problem node from the group. Conversely, when a member node does not receive a heartbeat message from the supervisor node within a set time limit, it assumes that the supervisor node has experienced a failure and selects itself to be the supervisor of the group. An actual or perceived supervisor node failure can therefore give rise to multiple single-node groups performing the same task. Multiple groups assigned to the same task are merged by demoting all of the supervisor nodes of the constituent groups, except one. Currently, demotion is carried out based upon the unique ID assigned to each node; among the conflicting nodes, the one with the highest ID is selected to be the group supervisor. Consider, for example, a group comprising three nodes a, b, and c, node a being the supervisor node. When a fails, b and c form two single-node groups and continue to perform the sensing task. In due course, nodes b and c discover each other—e.g., when b intercepts a `query_relevance` message from c—and they form a new group comprising b and c, demoting node c in the process. Thus, the proposed communication model is able to handle node failures.

5 Video Surveillance

Now, consider how a network of dynamic cameras may be used in the context of video surveillance (Fig. 5). A human operator spots one or more suspicious pedestrians 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 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 fields of view, pedestrian motion prediction models, occlusion models, and pedestrian movement pathways may allow (in some sense) *optimal* allocation of cameras; however, it is cumbersome and in most cases infeasible to acquire such a world model. Our approach eschews such a knowledge base. We assume only that a pedestrian can be identified by different cameras with reasonable accuracy and that the camera network topology is known *a priori* (any two cameras that are within communication range of each other are considered neighbours). A direct consequence of this approach is that the network can easily be modified through removal, addition, or replacement of camera nodes.

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 (see [3] for details). The computed relevance values are used by the node selection scheme described above 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 an insufficient number of free nodes are available for the current task.

6 Results

To date, we have tested our smart camera network system with up to 16 stationary and pan-tilt-zoom virtual cameras, and we have populated the virtual Penn station with up to 100 pedestrians. The camera 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 experiences increasing difficulty in following the correct pedestrian, so the surveillance task fails (and the cameras return to their default settings).

For the example shown in Fig. 6, we placed 16 active PTZ cameras in the train station, as shown in Fig. 1. An operator selects the pedestrian with the red shirt in Camera 7 (Fig. 6(e)) and initiates the "follow" task. Camera 7 forms the task group and begins tracking the pedestrian. Subsequently, Camera 7 recruits Camera 6, which in turn recruits Cameras 2 and 3 to track the pedestrian. Camera 6 becomes the supervisor of the group when camera 7 loses track of the pedestrian and leaves the group. Subsequently, Camera 6 experiences a tracking failure, sets Camera 3 as the group supervisor, and leaves the group. Cameras 2 and 3 track the pedestrian during her stay in the main waiting room, where she also visits a vending machine. When the pedestrian starts walking towards the concourse, Cameras 10 and 11 take over the group from Cameras 2 and 3. Cameras 2 and 3 leave the group and return to their default modes. Later, Camera 11, which is now acting as the group's supervisor, recruits Camera 9, which tracks the pedestrian as she enters the concourse.

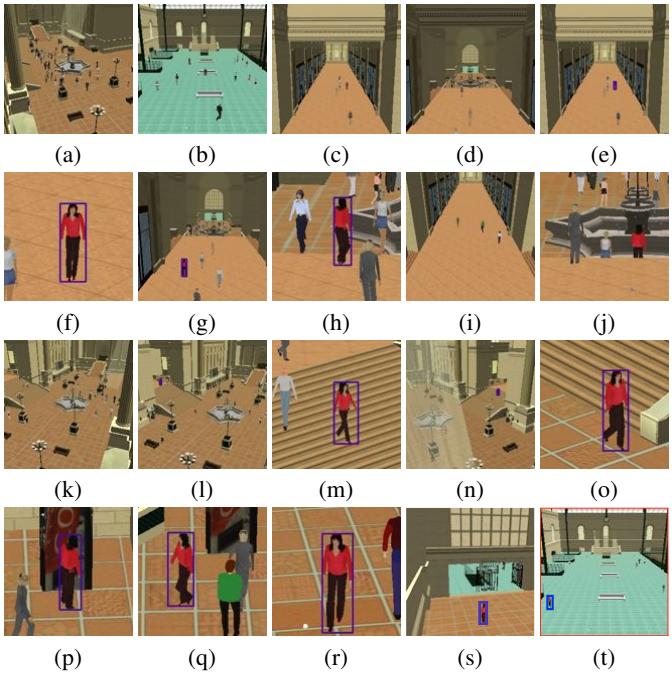

Figure 6: A pedestrian is successively tracked by cameras 7, 6, 2, 3, 10, and 9 (see Fig. 1) for 14 minutes as she makes her way through the station to the concourse. (a–d) Cameras 1, 9, 7, and 8 observing the station (elapsed time: 30 sec). (e) Operator selects a pedestrian in feed 7 (1.7 min). (f) Camera 7 has zoomed in on the pedestrian (2 min). (g) Camera 6, which is recruited by Camera 7, acquires the pedestrian (2.2 min). (h) Camera 6 zooms in on the pedestrian (3 min). (i) Camera 7 reverts to its default mode after losing track of the pedestrian—it is now ready for another task (3.5 min). (j) Camera 6 has lost track of the pedestrian (4.2 min). (k) Camera 2 (3 min). (l) Camera 2, which is recruited by Camera 6, acquires the pedestrian (4 min). (m) Camera 2 tracking the pedestrian (4.3 min). (n) Camera 3 is recruited by the Camera 6; Camera 3 has acquired the pedestrian (4 min). (o) Camera 3 zooming in on the pedestrian (5 min). (p) Pedestrian is at the vending machine (6 min). (q) Pedestrian is walking towards the concourse (13 min). (r) Camera 10 is recruited by Camera 3; Camera 10 is tracking the pedestrian (13.4 min). (s) Camera 11 is recruited by Camera 10 (14 min). (t) Camera 9 is recruited by Camera 10 (15 min).

7 Conclusion

We envision future video surveillance systems to be networks of stationary and active cameras capable of providing perceptive coverage of extended environments with minimal reliance on human operators. Such systems will require not only robust, low-level vision routines, but also novel camera network methodologies. The work presented in this paper is a step toward the realization of smart camera networks and our initial results appear promising.

A unique and important aspect of our work, is that we have developed and demonstrated our prototype video surveillance system in a realistic virtual train station environment populated by lifelike, autonomously self-animating virtual pedestrians. Our sophisticated camera network simulator should continue to facilitate our ability to design such large-scale networks and experiment with them on commodity personal computers.

The overall behavior of our prototype smart camera 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 central controller, we expect the proposed approach to be robust and scalable.

Acknowledgments

The research reported herein was made possible 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 their invaluable contributions to the implementation of the Penn Station simulator.

8 References

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

[2] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich, "Collaborative signal and information processing: An information directed approach," *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1199–1209, 2003.

[3] F. Qureshi and D. Terzopoulos, "Towards intelligent camera networks: A virtual vision approach," in *Proc. The Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS05)*, Beijing, China, Oct. 2005, pp. 177–184.

[4] D. Terzopoulos and T. Rabie, "Animat vision: Active vision in artificial animals," *Videre: Journal of Computer Vision Research*, vol. 1, no. 1, pp. 2–19, Sept. 1997.

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

[6] A. Santuari, O. Lanz, and R. Brunelli, "Synthetic movies for computer vision applications," in *Proc. 3rd IASTED International Conference: Visualization, Imaging, and Image Processing (VIIP 2003)*, no. 1, Spain, Sept. 2003, pp. 1–6.

[7] J. Mallett, "The role of groups in smart camera networks," Ph.D. dissertation, Program of Media Arts and Sciences, School of Architecture, Massachusetts Institute of Technology, Feb. 2006.

[8] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration for tracking applications," in *IEEE Signal Processing Magazine*, Mar. 2002, vol. 19, pp. 61–72.

[9] M. Bhardwaj, A. Chandrakasan, and T. Garnett, "Upper bounds on the lifetime of sensor networks," in *IEEE International Conference on Communications*, no. 26, 2001, pp. 785 – 790.

[10] R. Collins, A. Lipton, H. Fujiyoshi, and T. Kanade, "Algorithms for cooperative multisensor surveillance," *Proceedings of the IEEE*, vol. 89, no. 10, pp. 1456–1477, Oct. 2001.

[11] 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, USA, Sept. 2002, pp. 517–520.

[12] D. Devarajan, R. J. Radke, and H. Chung, "Distributed metric calibration of ad-hoc camera networks," To appear in ACM Transactions on Sensor Networks, 2006.

[13] 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, Nov. 2005, pp. 131–139.

[14] J. K. Pearson and P. G. Jeavons, "A survey of tractable constraint satisfaction problems," Royal Holloway, University of London, Tech. Rep. CSD-TR-97-15, July 1997.

[15] M. Yokoo, *Distributed Constraint Satisfaction: Foundations of Cooperation in Multi-agent Systems*. Berlin, Germany: Springer-Verlag, 2001.