A Virtual Vision Simulator for Camera Networks Research

Wiktor Starzyk
wiktor.starzyk@uoit.ca

Adam Domurad
adam.domurad@mycampus.uoit.ca

Faisal Z. Qureshi
http://faculty.uoit.ca/qureshi

Faculty of Science
University of Ontario Institute of Technology
Oshawa, ON L1H7K4 Canada

***Abstract*—Virtual Vision advocates developing visually and behaviorally realistic 3D synthetic environments to serve the needs of computer vision research. Virtual vision, especially, is well-suited for studying large-scale camera networks. A virtual vision simulator capable of generating "realistic" synthetic imagery from real-life scenes, involving pedestrians and other objects, is the sine qua non of carrying out virtual vision research. Here we develop a distributed, customizable virtual vision simulator capable of simulating pedestrian traffic in a variety of 3D environments. Virtual cameras deployed in this synthetic environment generate imagery using state-of-the-art computer graphics techniques, boasting realistic lighting effects, shadows, etc. The synthetic imagery is fed into a visual analysis pipeline that currently supports pedestrian detection and tracking. The results of this analysis can then be used for subsequent processing, such as camera control, coordination, and handoff. It is important to bear in mind that our visual analysis pipeline is designed to handle real world imagery without any modifications. Consequently, it closely mimics the performance of visual analysis routines that one might deploy on physical cameras. Our virtual vision simulator is realized as a collection of modules that communicate with each other over the network. Consequently, we can deploy our simulator over a network of computers, allowing us to simulate much larger camera networks and much more complex scenes then is otherwise possible.**

***Keywords*-virtual vision, camera networks, ptz cameras, pedestrian tracking**

I. Introduction

Multi-camera systems are rapidly evolving from highly specialized wired networks of stationary passive and active cameras designed to provide visual coverage of the scene to *ad hoc* networks of smart camera nodes, capable of near-autonomous operation to support a variety of applications, such as urban and participatory sensing, disaster response, plant and animal habitat monitoring, etc. Whereas traditional multi-camera systems focus primarily on multi-camera scene analysis, smart camera networks are also concerned with camera coordination and control, in-network processing and storage, and resource aware visual analysis. Pre-recorded video, while useful, is not sufficient to study camera control and coordination strategies. Rather one needs access to such a camera network itself. This observation together with the fact that most researchers who are motivated to study camera networks do not have access to physical camera networks of suitable complexity led to the development of the virtual vision paradigm for camera networks research [1].

Virtual vision paradigm for computer vision research advocates employing *reality emulators*—visually and behaviorally realistic 3D environments, richly populated with pedestrians, automobiles, etc.—to carry out camera networks research. Virtual camera networks of suitable complexity can be simulated within these synthetic environments, called *virtual vision simulators*. Using a virtual vision simulator offers several advantages over traditional physical camera setups during ideation, prototyping, and evaluation phases, including:

- legal issues surrounding access to physical camera networks installed in public spaces disappear when dealing with a simulated camera network;
- developing a virtual vision simulator is a major undertaking; however, once such a simulator becomes available, the cost of carrying out camera networks research within this simulator is minimal compared to performing research on a physical camera network—a virtual vision simulator runs on standard PCs and does not require any special hardware;
- virtual vision offers quick prototyping—it is much easier and faster to reconfigure a virtual camera network than it is to reconfigure a physical camera network;
- complex vision and control algorithms that need to be studied in "real time" can be easily studied in a virtual vision simulator by slowing down the virtual clock of the simulated environment;
- virtual vision offers far faster design/evaluation iterations when compared to a physical camera network;
- ground truth is readily available; and
- camera control and coordination algorithms can easily be compared against each other since scenes are perfectly repeatable.

Qureshi and Terzopoulos demonstrated the virtual vision paradigm of camera networks research by designing and studying simulated camera networks [1]. In their work, they relied upon a virtual vision simulator comprising a 3D reconstruction of the Penn train station, inhabited by up to 1000 self-animating pedestrians, developed by Shao and Terzopoulos [2].

Figure 1: A view of our virtual world showing pedestrians walking on an upper floor of an office building. Toronto (Canada) skyline is visible through floor to ceiling panoramic windows. Our scripted pedestrians use motion-capture data to simulate realistic motion and cast dynamic shadows on the floor and the walls.

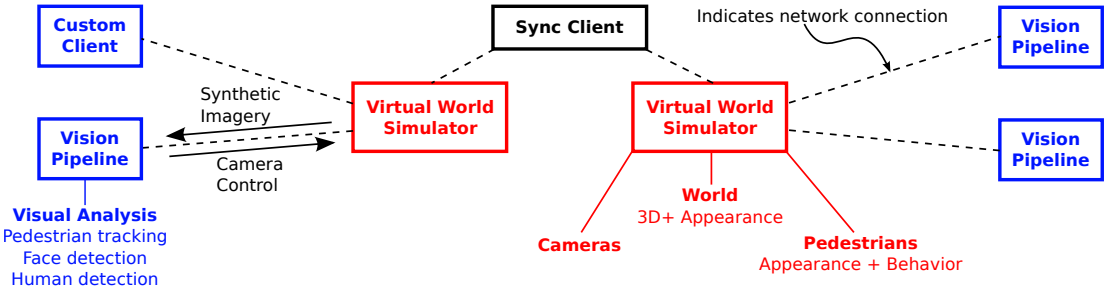


Figure 2: An overview of our distributed virtual vision simulator.

Here, we present a distributed virtual vision simulator. Our simulator is designed to address some of the shortcomings of the virtual vision simulator employed by Qureshi and Terzopoulos in their work on smart camera networks. Specifically, their virtual vision simulator is tied to a single PC; whereas, our simulator can be deployed over a network of computers. The ability to spread the computational load over multiple computers assumes urgency as we begin to simulate richer, more complex synthetic worlds and larger, more sophisticated virtual camera networks. Unlike the virtual vision system used by Qureshi and Terzopoulos [1], our simulator uses state-of-the-art graphics technology to support advanced rendering effects—such as lighting, shadows, transparency—which adds to the visual realism of the scene (Fig. 1). Lastly, we have a developed a vision pipeline that works without any modifications on both synthetic imagery generated by our virtual vision simulator and real video captured by a physical camera network. This capability, we believe, will help us validate our simulator in the future.

II. Related Work

In 1995, Terzopoulos and Rabie introduced a software based approach to designing active vision systems called animat vision [3]. The animat approach replaces physical robots and cameras with artificial animals referred to as animats [4]. The animats are placed in physics-based virtual worlds to study active vision systems. Eschewing the need for real cameras, robots, and other hardware, at least during the early stages of research, this approach promises huge savings in time and money.

In 2003, Terzopoulos proposed using reality emulators for computer vision research [5]. Santuari *et al.* developed a virtual museum simulator, populated with scripted visitors, to study computer vision algorithms [6]. This simulator was aimed at developing low level pedestrian segmentation and tracking algorithms. The virtual museum simulator uses sophisticated 3D rendering techniques with support for global illumination, shadows and different visual artifacts such as motion blur and interlacing.

In 2005, Shao and Terzopoulos developed a train station simulator, populated with self-animating pedestrians (commuters and visitors) [2]. Qureshi and Terzopoulos used this train station simulator to develop the first of its kind virtual vision simulator [1]. They demonstrated their virtual vision simulator by studying high-level camera control and coordination problems in camera networks comprising both passive and active cameras. The virtual vision simulator presented in this paper addresses many of the shortcomings of the virtual vision simulator developed by Qureshi and Terzopoulos in [1]. Our simulator has superior synthetic image quality; it supports subtle lighting effects and dynamic shadows. But more importantly the simulator developed here is distributed. Consequently, it is capable of simulating much larger camera networks and far richer synthetic environments. Lastly, our simulator only uses open source libraries.

III. Contributions and Overview

The contributions of the research presented herein are twofold. First, we develop a distributed virtual vision simulator. We demonstrate the suitability of this simulator for cam-

Figure 3: VW consists of the top floor of an office building situated in downtown Toronto. Toronto skyline is visible through floor-to-ceiling panoramic windows. Sunlight filters through the window and cast dynamic shadows. (Top-Row) Top-down view of the office floor, showing individual offices, conference rooms, common areas, and elevator lobbies. (Bottom-Row) Synthetic imagery captured by 4 different cameras (two left images are captured by PTZ cameras). Notice the subtle lighting and shadow effects, which were missing in the virtual vision simulator developed by Shao and Terzopoulos [1].

era networks research by providing examples of some recent work on smart camera networks that was carried out within this simulator. Second, we present a flexible visual analysis framework capable of analyzing multiple image/video/camera streams simultaneously. The visual analysis framework was designed from ground up to handle imagery captured by a camera network. It works for both synthetic imagery (captured by simulated camera networks) and real imagery (collected by a physical camera network).

The rest of the paper is organized as follows. The next section provides an overview of our virtual vision simulator. Sec. V presents the virtual world engine. We introduce the visual analysis pipeline in the following section. Sec. VII discusses the synchronization unit and the network operation of the virtual vision simulator. We present three camera networks research projects that used our simulator in Sec. VIII. Sec. IX concludes the paper with a brief discussion.

IV. System Overview

Our virtual vision simulator consists of three types of modules: 1) virtual world engine (VW), 2) visual analysis pipeline (VP) and 3) synchronization unit (SYNC). In a typical realization of the virtual vision simulator, one or more instances of VW and VP modules are spread over a network of computers (see Fig. 2). These modules communicate with each other over the network. VP modules analyze images captured by the cameras simulated in VWs and use the results of this analysis to control and coordinate these

(a)

(b)

Figure 4: Starting with just five 3D human models (a), we are able to generate many more pedestrians with unique appearances by modifying the appearance textures (b).

cameras. Typically each camera has its own dedicated VP module. A single instance of a SYNC module ensures that all VW modules are in sync with each other. The next three sections describe each of the three components of our virtual vision simulator.

V. Virtual World Engine

VW is responsible for simulating virtual 3D scenes, inhabited with self-animating pedestrians, automobiles, etc. Specifically our VW models the top floor of an office tower in downtown Toronto, populated with virtual humans who work on this floor (Fig. 3). These virtual humans use motion capture data and scripted way-points for locomotion and exhibit highly-realistic movements. Virtual cameras deployed in this environment can generate synthetic imagery mimicking video captured by a typical surveillance camera. Currently VW supports passive, wide Field-of-View (FOV) and active Pan/Tilt/Zoom (PTZ) cameras. These cameras can be easily configured and placed anywhere in the virtual environment to prototype a camera network having the desired configuration. The office floor is complete with floor-to-ceiling panoramic windows looking out at the Toronto skyline. VW supports dynamic shadows and subtle lighting effects, such as sunlight filtering through glass windows.

A. Virtual Humans

The 3D office floor is inhabited by self-animating virtual pedestrians. The pedestrians exhibit life-like movements (walk, turn, run, stop, etc.) by relying upon motion capture data. These pedestrians move around the environment following scripted paths, defined as a series of "reach waypoint" actions. Our approach provides the user with full control over where each pedestrian is at any given time, while at the same time keeps script complexity manageable. Each pedestrian maintains a queue of these actions and executes them in order.

Currently we have 3D models for three males and two females; these are shown in Fig. 4(a). We are able to generate

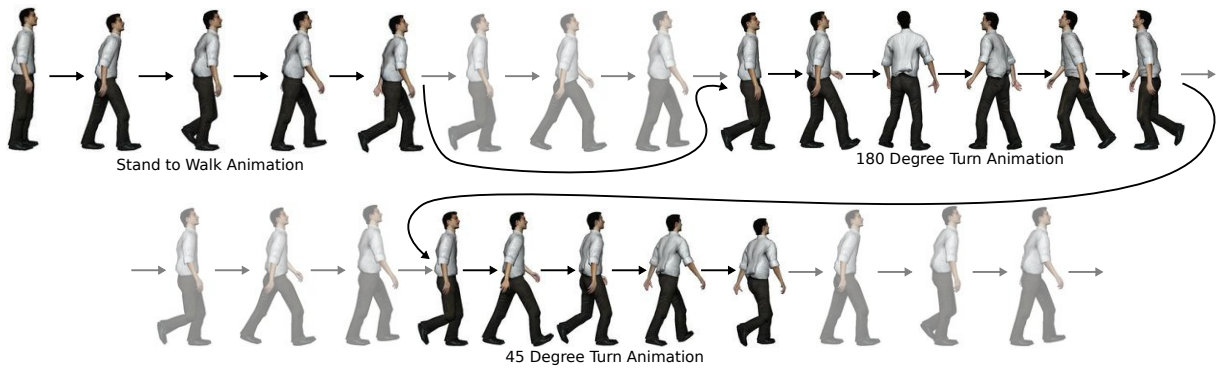


Figure 5: Transition between different locomotion states (walk to run, etc.) is accomplished by defining a motion graph over motion capture data corresponding to different locomotion states. The motion graph minimizes the jumps when transitioning from one state to the other.

many more humans with unique appearances by modifying the textures available for these 5 models (Fig. 4(b)). This allows us to realize scenes with far more individuals then is otherwise possible. Transitions between different states, e.g., walk to turn left 90 degrees, is accomplished by defining a motion graph on top of the motion capture data available for each low-level state (Fig. 5) [7]. Locomotion states currently available for each human are: 1) walk, 2) run, 3) stand idle, 4) about turn, 5) turn left 90 degrees, 6) turn right 90 degrees, 7) turn left 45 degrees, 8) turn right 45 degrees, 9) start walking from idle, 10) come to a stop, 11) start running from a walking gait, and 11) run to walk.

B. Virtual Cameras

Our system currently supports passive wide-FOV cameras and active PTZ cameras. Clients can connect to these cameras over the network. In that respect our simulated passive and active cameras behave similarly to typical IP surveillance cameras. Each camera exposes a control interface through which a client can control the camera and request images from this camera. Table. I lists the commands that are available to access, configure, and control these cameras over the network.

We have developed a high-throughput, light-weight protocol for accessing these simulated cameras. Our protocol can be easily mapped to communication protocols used by physical IP cameras, such as Pelco-D standard [8]. It suggests that the camera networks developed within the virtual environment can be easily ported to a physical camera network.

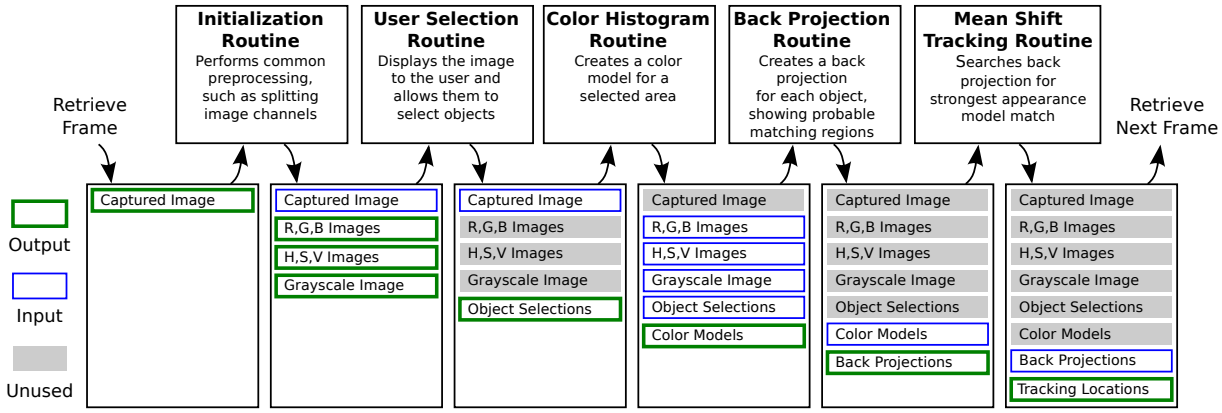
Camera Type	Command	Description
PTZ & wide-FOV	setResolution	Set the resolution of the image
	getImage	Gets the latest image from the camera
PTZ	panLeft	Pan left by θ degrees
	panRight	Pan right by θ degrees
	tiltUp	Tilt up by θ degrees
	tiltDown	Tilt down by θ degrees
	zoomIn	Zoom in by θ degrees
	zoomOut	Zoom out by θ degrees
	default	Reverts the camera to its default settings

Table I: Simulated passive wide-FOV and active PTZ cameras support a set of commands similar to those available in a typical IP camera.

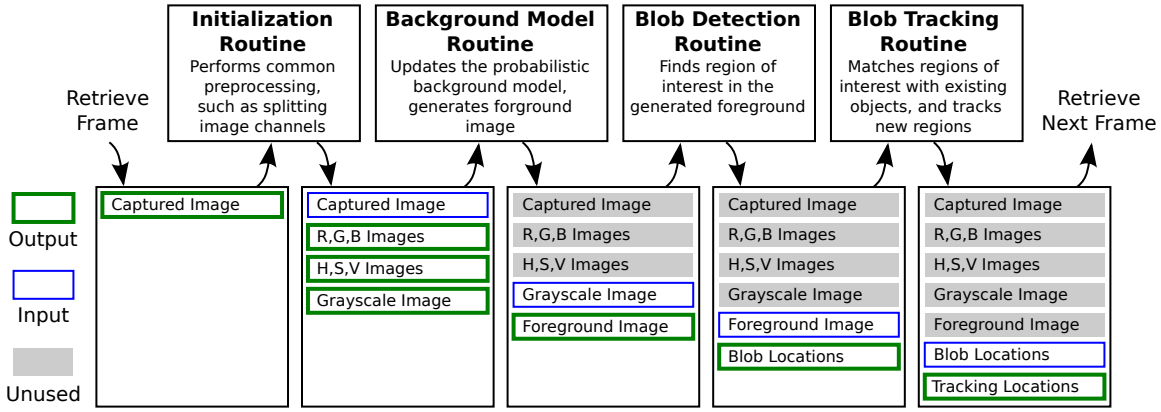
Figure 7: Our visual analysis pipeline is designed from the ground up to work with both synthetic (right) and real video (left) without any modifications. Consequently, our vision pipeline faithfully mimics the performance of a vision pipeline implemented on physical cameras.

VI. Visual Analysis Pipeline

We have developed pedestrian detection and tracking routines that can be put together to construct a visual analysis pipeline, capable of tracking individual pedestrians in videos captured via passive wide-FOV and active PTZ cameras. Currently we have implemented two pipelines: 1) a pedestrian tracker capable of tracking multiple pedestrians in video feeds from passive wide-FOV cameras and 2) a pedestrian tracker capable of tracking one or more "selected" pedestrians in video feeds from active PTZ cameras (Fig. 6).



(a) Visual analysis pipeline for tracking pedestrians in PTZ cameras.



(b) Visual analysis pipeline for tracking pedestrians in wide-FOV cameras.

Figure 6: Visual analysis pipelines are realized as a collection of reusable vision routines.

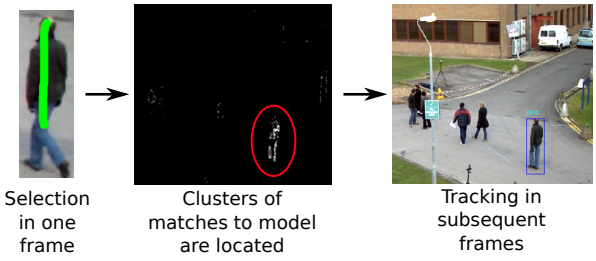


Figure 8: A stroke gesture is provided to select a pedestrian to be tracked in active PTZ cameras. Appearance signatures computed by passive wide-FOV cameras can also be used to track individuals in active PTZ cameras.

These pipelines serve our purpose well, since we are currently only interested in camera networks comprising passive wide-FOV and active PTZ cameras deployed in urban settings for observing pedestrians in the scene. It is important to note that our visual analysis pipelines work equally well for synthetic imagery captured by virtual cameras and real footage captured using physical cameras (Fig. 7). This will make it easier to port camera networks implemented within our virtual vision to real, physical camera networks.

A. *Construction*

Visual analysis pipelines are realized as a set of vision routines (Fig. 6). Whenever a new image arrives, a frame object is constructed. Every vision routine in the pipeline operates upon this frame object in sequence, modifying the frame object in the process. Vision routines that come after can access results computed by previous routines through this frame object. Consequently, the frame object provides a communication and coordination mechanism between the various vision routines that constitute a visual analysis pipeline. E.g., a vision routine can dynamically check if it needs to run a particular algorithm. This can lead to efficiencies and more intelligent management of resources.

So far we have implemented the following set of computer vision routines:

- Routines that produce appearance models of objects, and routines that use these appearance models to construct the back-projection image;
- Routines for foreground detection, using state-of-the-art background subtraction algorithms;

- Routines for blob detection and tracking; and
- Routines for face and head detection.

B. Pedestrian Tracking

The pedestrian tracker for passive wide-FOV cameras relies upon background subtraction to detect and subsequently track pedestrians. This tracker works completely autonomously and does not require any human assistance. After a few minutes of training the tracker is able to pick out and track pedestrians present in the scene. It also constructs appearance based signatures of the pedestrians being tracked. Background subtraction, however, is not available for active PTZ cameras since the background changes as the camera parameters are adjusted. The pedestrian tracker developed for active PTZ cameras, therefore, employs an appearance based signature to track one or more pedestrians. Appearance signatures of pedestrians that need to be tracked are available from the wide-FOV cameras. We have also developed a stroke-based interaction mechanism that allows a user to quickly identify (select) the person to be tracked in the video feed from an active PTZ camera (Fig. 8).

Our visual analysis pipelines are able to track individual pedestrians reasonably well in low density environments, such as office buildings. These will not work in situations involving a large number of pedestrians crammed together in a small physical space, such as a subway station during rush hours. Nevertheless, these visual analysis pipelines enable us to study camera control and coordination strategies in our virtual vision simulator.

VII. Synchronization Unit

Fig. 2 depicts a possible configuration of our virtual vision simulator. Each VW is responsible for simulating the visually and behaviorally realistic 3D environments within which virtual cameras are deployed. VP modules are responsible for analyzing video data captured by simulated cameras. VPs are also able to control the simulated cameras. For example, a VP can choose to pan a PTZ camera to follow an individual of interest. Additionally, VPs are also able to communicate with each other (via the VW or SYNC); e.g., during camera handoffs, etc. VWs and VPs are spread over multiple computers and we use a SYNC unit to ensure that all VWs evolve in lockstep. This ensures that images captured by various cameras and commands issued to these cameras are perfectly synchronized across the whole (simulated) camera network no matter the speed of the computer hosting a particular VW or VP. It also gives us full control over the simulation clock allowing us to easily test cameras with different framerates as well as camera routines that have not been optimized to run in realtime. Of course it is straightforward to simulate asynchronous image capture and processing.

Fig. 9 illustrates how the SYNC unit communicates with multiple VW modules. SYNC unit begins by registering

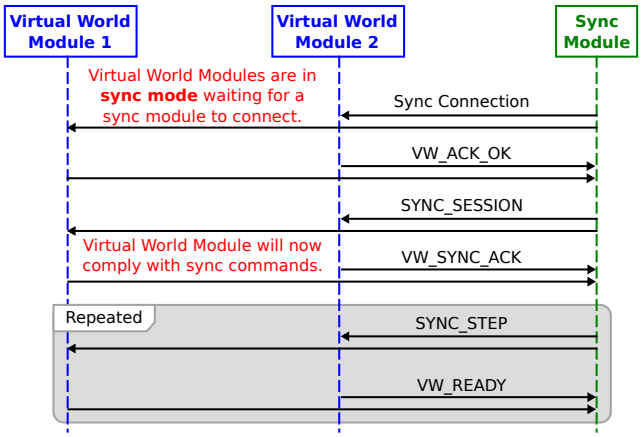


Figure 9: Communication between VWs and SYNC unit.

every VW in its database. Next, SYNC unit, which keeps track of the global clock, sends step signals to each VW unit to take a fixed number, say n, of simulation steps. The next step is only issued once every VW has successfully completed n simulation steps. This ensures that the internal clocks of two VWs are never more than n simulation steps apart. VWs update their clocks upon receiving the sync message.

To create a smart camera network, it is critical that the cameras be able to communicate with each other. There are two ways this can be achieved using our system. When both the sender and the recipient cameras reside on the same VW, the message can be transferred without going through the network interface. However, if the recipient camera is being simulated on a different VW, the message is sent to the SYNC unit which can then pass the message on to the appropriate VW. Alternately our system allows anyone to implement their own method of communication that bypasses VW and SYNC entirely.

VIII. Examples of Camera Networks

Here we briefly describe three prototype camera networks that were deployed and studied within our virtual vision systems.

A. Multi-tasking PTZ Cameras

The first project focuses on developing a system that automatically tunes the sensing parameters of PTZ cameras in response to the scene activity, choosing to capture close-up video when the number of pedestrians present in the scene is low and electing to capture lower-resolution video as the number of pedestrians increases, thus always keeping every pedestrian in view [9]. These cameras enable the video surveillance system to intelligently respond to scene complexity, automatically capturing close-up imagery of the pedestrians present in the scene when possible, and behaving as wide-FOV cameras when the number of pedestrians

increases. Fig. 10 shows a multi-tasking PTZ camera: the PTZ camera is able to capture higher resolution video of the pedestrians present in the scene when there are only a few pedestrians present; however, it begins to behave like a wide-FOV camera as the pedestrians present in the scene spread out and move away from the camera.

Figure 10: PTZ cameras automatically decide how best to observe a scene. (a) When possible, the PTZ camera selects a higher zoom to capture higher resolution images of the individuals present in the scene. (b) As the individuals spread out and move away from the camera, the PTZ camera selects a lower zoom setting to keep everybody in view, albeit at a much lower resolution. (c) Virtual vision simulator consisted of one VW and one VP module, spread over two computers.

B. *Learning Proactive Control Strategies*

The second project focuses on developing a PTZ camera network that learns proactive control strategies by generalizing and storing the results of a reasoning process (Fig. 11). The reasoning process—which considers both immediate and far-reaching consequences of different camera assignments when constructing a "plan" most likely to succeed (in a probabilistic sense) at the current observation task(s)—is capable of performing camera assignments and handoffs in order to provide persistent coverage of a region. The results of this reasoning activity are then stored as rules in a production system [10]. Later when a similar situation is encountered, the production system bypasses the reasoning process and performs camera assignments. Initially, the camera network relies mostly on the reasoning process; over time, however, camera assignments become instinctive.

C. *Camera Handoffs*

Fig. 12 illustrates camera handoff between a passive wide-FOV camera and a nearby active PTZ camera. In this case, a passive camera detected the individual through background subtraction and began tracking this individual. Once it detected that the individual is about to leave its field-of-view, the passive camera sent the appearance signature

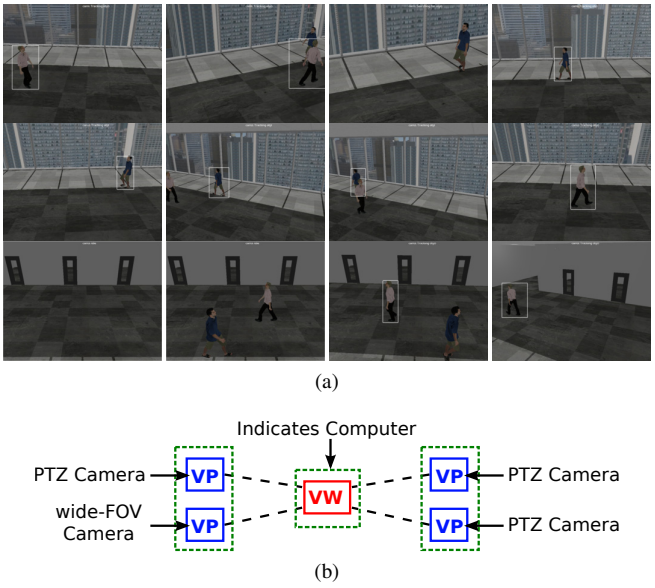


Figure 11: (a) The three rows show three cameras observing two pedestrians as they cross each other on their way to the opposite sides of the lobby. The three cameras are able to perform handoff while keeping both pedestrians in view. This is achieved through a reasoning mechanism that considers both short-term and long-term consequences of camera assignments. (b) Virtual vision simulator consisted of one VW and four VP modules, spread over three computers.

for this individual to a nearby active camera. The active PTZ camera then is able to track the individual using the appearance signature sent to it by the passive camera.

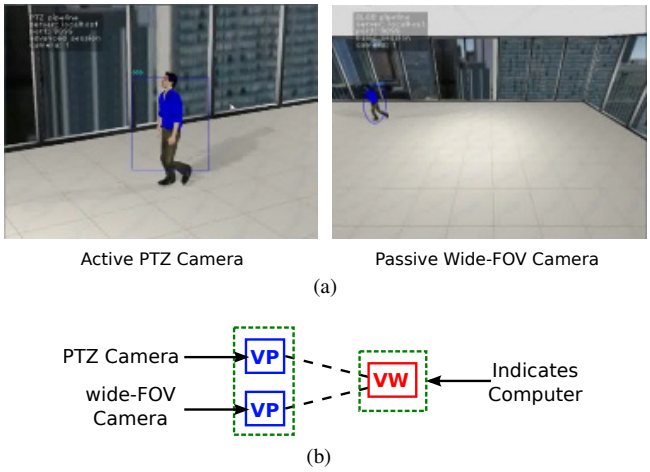


Figure 12: A passive wide-FOV camera hands off a pedestrian to a PTZ camera by sending it the pedestrians appearance signature. (b) Virtual vision simulator consisted of one VW and two VP modules, spread over two computers.

Figure 13: A view of our virtual vision simulator showing an outdoor scene.

IX. Conclusion

Reality Emulators—visually and behaviorally realistic environments, inhabited with life-like flora and fuana—have the potential to revolutionize camera networks research. These synthetic environments can serve as software laboratories within which simulated camera networks can be deployed, tested and evaluated. Inspired by this vision, here we present a 3D environment, along with the necessary camera models, communication infrastructure, and computer vision routines, that can be beneficial for camera networks research. With the ability to quickly change the scene, the number of pedestrians and the locations and properties of the cameras, it is easy for researchers to study their camera networks in a variety of settings (See Fig. 13).

We are currently working to improve the quality of our virtual pedestrians so that they can better interact with each other and their surroundings. We also plan to implement the Pelco-D protocol for the simulated cameras, to further facilitate the transfer of camera networks algorithms from our simulated environment to physical camera networks. Finally, we hope to improve the quality of the imagery produced by our simulator by adding support for lens effects such as depth of field, noise and motion blur.

Acknowledgements

The authors would like to thank Natural Science and Engineering Research Council (NSERC) of Canada for their generous financial support of this research. We would also like to thank Xerox Foundation for their support.

References

[1] Qureshi, Faisal Z and Terzopoulos, Demetri, "Smart camera networks in virtual reality," *Proceedings of the IEEE (Special Issue on Smart Cameras)*, vol. 96, no. 10, pp. 1640–1656, October 2008.

[2] W. Shao and D. Terzopoulos, "Autonomous Pedestrians," *Graphical Models*, vol. 69, no. 5-6, pp. 246–274, September 2007.

[3] D. Terzopoulos and T. F. Rabie, "Animat vision: Active vision in artificial animals," in *Proceedings of the Fifth International Conference on Computer Vision*, ser. ICCV '95. Washington, DC, USA: IEEE Computer Society, 1995, pp. 801–. [Online]. Available: http://portal.acm.org/citation.cfm?id=839277.840048

[4] S. W. Wilson, "The animat path to ai," in *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*. Cambridge, MA, USA: MIT Press, 1990, pp. 15–21. [Online]. Available: http://portal.acm.org/citation.cfm?id=116517.116519

[5] D. Terzopoulos, "Perceptive agents and systems in virtual reality," in *Proceedings of the ACM symposium on Virtual reality software and technology*, 2003.

[6] R. B. A. Santuari, O. Lanz, "Synthetic movies for computer vision applications," in *Proceedings of the third IASTED International Conference on Visualization, Imaging, and Image Processing*, 2003.

[7] L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," *ACM Transaction on Graphics*, vol. 21, pp. 473–482, July 2002.

[8] "Pelco-d protocol manual," pp. 1–8, March 1999.

[9] W. Starzyk and F. Z. Qureshi, "Multi-tasking smart cameras for intelligent video surveillance systems," in *Proc. 8th IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS 2011)*, Klagenfurt, Austria, August 2011, pp. 1–6.

[10] F. Z. Qureshi and W. Starzyk, "Learning proactive control strategies for ptz cameras," in *Proc. Fifth ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 2011)*, Ghent, Belgium, August 2011, pp. 1–6.