DOTS:
Support for Effective Video Surveillance

要　旨

DOTS（Dynamic Object Tracking System）は、研究目的で実際のオフィス環境に設置された、屋内型、リアルタイム、マルチカメラの監視システムである。DOTS では、監視員が注目領域を効果的に監視したり人を追跡することが出来るように、ビデオ分析技術と UI コンポネントを一体化したシステムとなっている。DOTS のビデオ分析技術では、特徴レベル画像前景部セグメント化を行い、複雑な条件の元でも信頼性の高い結果が得られている。また、多人数の人を追跡する効果的アプローチを組み込み、個々のカメラから得られる結果をマルチカメラの軌跡として一体化している。DOTS の UI では、簡単に参照することでユーザーが重要なイベントに注目できるようにしている。また、同一 UI の中の異なる表現により、より簡単なナビゲーションが出来る空間情報も提供される。オフィスのあるビルの廊下や公共の場に設置された 20 台以上のカメラを用い、我々は、この DOTS を 1 年にわたって定常的に利用してきた。その経験により、ビデオ監視性能を向上させる多くの違いを導き出す事が出来た。

Abstract

DOTS (Dynamic Object Tracking System) is an indoor, real-time, multi-camera surveillance system, deployed in a real office setting. DOTS combines video analysis and user interface components to enable security personnel to effectively monitor views of interest and to perform tasks such as tracking a person. The video analysis component performs feature-level foreground segmentation with reliable results even under complex conditions. It incorporates an efficient greedy-search approach for tracking multiple people through occlusion and combines results from individual cameras into multi-camera trajectories. The user interface draws the users' attention to important events that are indexed for easy reference. Different views within the user interface provide spatial information for easier navigation. DOTS, with over twenty video cameras installed in hallways and other public spaces in our office building, has been in constant use for a year. Our experiences led to many changes that improved performance in all system components.

執筆者

Andreas Girgensohn
Don Kimber
Jim Vaughan
Tao Yang
Frank Shipman
Thea Turner
Eleanor Rieffel
Lynn Wilcox
Francine Chen
Tony Dunnigan

FX Palo Alto Laboratory, Inc.

Reprinted from DOI :
http://doi.acm.org/10.1145/1291233.1291332.

1. Introduction

Video surveillance systems are common in commercial, industrial, and residential environments. A common surveillance activity is to track important people, or people exhibiting suspicious behavior, as they move from camera to camera. With the decreasing cost of video hardware, the number of video streams per installation is increasing [9]. The increased scale creates difficulties for humans trying to recognize important events as they happen and to track people through the monitored space.

Many commercial surveillance systems target a small number of susceptible areas in a business and make use of "hot spots" or "trip wires" to identify activities of interest. Such systems include those offered by Panasonic [24], Vidient [30], Vistascape [31], and ObjectVideo [23]. Although these techniques enable many useful functions, including tailgate detection, counting people passing through a region, and detecting an intruder in a forbidden area, other surveillance functions are needed to secure a building.

We have developed a multi-camera surveillance system, DOTS (Dynamic Object Tracking System), for monitoring an office building, where one goal is to track people of interest. Network cameras are placed to cover the public spaces, where the system automatically detects and tracks people. Our goals are similar to those of IBM's Smart Surveillance System [12] that combines several techniques into a more complete surveillance solution. Specifically, our system combines video analysis and user interface techniques in a realistic, long-term setting to improve the performance of video surveillance systems.

This paper, unlike prior papers [10, 17, 35], presents for the first time all system components, including several new system components: a 3D surveillance viewer; navigation aids to detected events; an interactive hot spot definition with immediate event retrieval; face binding to tracked people; and summaries showing those faces. The main contributions are the innovative user interfaces for surveillance tasks that make use of techniques for tracking people and the long-term experience with a surveillance system in a real office setting.

DOTS includes a network video recorder (NVR) that records video from digital network cameras. The term 'digital video recorder,' also applicable to our recorder, is more commonly used in situations where analog video is digitized before being recorded. In addition to the NVR, the system combines activity analysis, object tracking, and event detection. Its configurable user interface aids humans in locating activities and important events, and tracking people across cameras.

DOTS tracks people by mapping foreground shapes from calibrated cameras into a 3D scene model. Information from several cameras is fused using the 3D trajectory information to estimate the final object correspondence across multiple cameras. The video player user interface automatically keeps a tracked person in view. Users may also drag iconic representations of tracked people to control the video playback.

We also created a user interface to aid in manually tracking a person from camera to camera. The Spatial Multi-Video (SMV) player conveys spatial proximity by surrounding the central main video display with multiple smaller video displays that are placed such that a person walking out of one camera view will likely appear in an adjacent camera view in the direction that they were moving in the main display (see Figure 4).

In the next section, we discuss related work. Then, we describe the major components for displaying the video and analysis results in the user interface, recording video, tracking people, and detecting events. We conclude with experiences with our system and suggestions for future research.

2. Related Work

There are a large number of commercial systems for video surveillance. Some are under development by companies who specialize in public safety and military solutions [11, 15]. Others are available from facilities management and engineering companies [9] and from the technology sector [1, 23, 30]. Some systems are beginning to provide geographic context information [19, 31].

A primary focus of surveillance research has been automatic tracking of activity across cameras [3, 6, 33, 34]. Our system supports automatic tracking and provides a user interface for situations where human assistance is required to augment the automatic tracking. Work on user interfaces for security video has emphasized the post-event task of video forensics [13]. Such systems have incorporated interactive visualizations of video content [8], the use of timelines [4], and virtual environments [28]. In comparison, there is limited research into user interfaces that aid real-time tracking. Iannizzotto and colleagues report on the use of gestures to control video selection and playback in a surveillance interface including a camera bank and a map [14]. Wang et al. describe techniques for identifying and visualizing salient aspects of a video feed [32].

Considering video interfaces related to our user interfaces more generally, many systems provide keyframes for navigation. The Rframes system [2] provides a list of keyframes that a user can scroll through to find a segment of interest and to start video playback. While those systems use keyframes as navigation aids, they do not synchronize the display of multiple camera streams.

3. User Interface

The DOTS user interface (see Figure 1) displays multiple streams of recorded or live video. It provides automatic object tracking and visualization based on video analysis results stored in a database. The user interface includes a camera bank, a timeline with event display, a floor plan, and a main player area. The main player area displays important camera views such as those that show a person currently being tracked. When automatic person-tracking is insufficient, the Spatial Multi-Video (SMV) player described below supports manual tracking of a person from camera to camera [10].

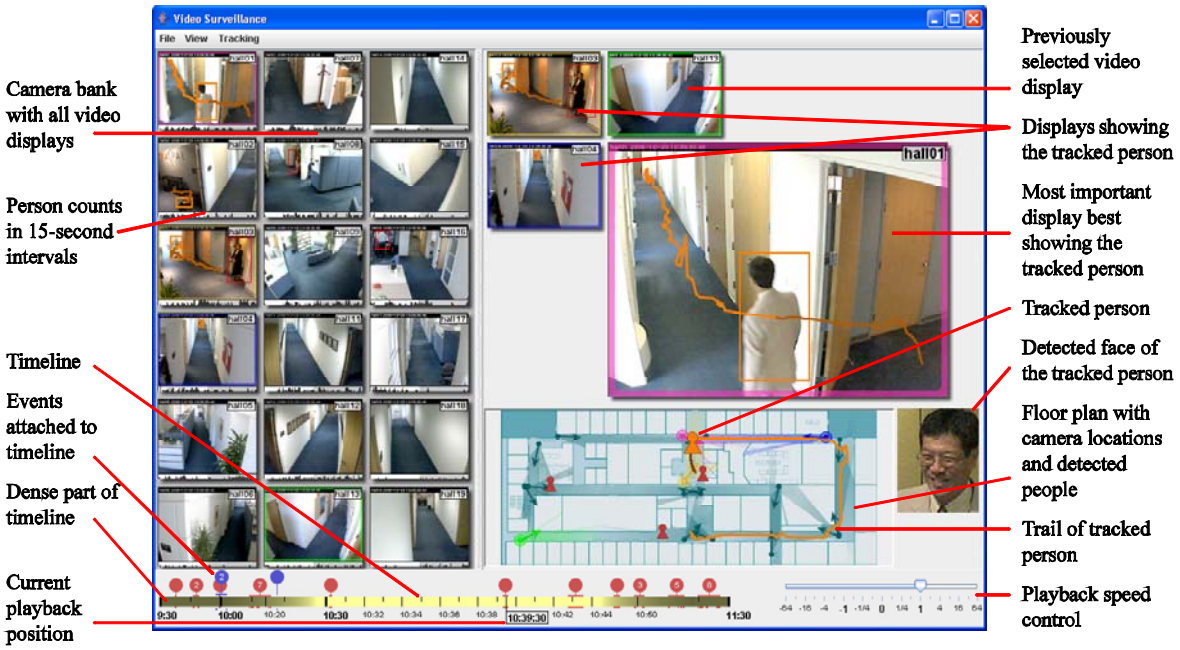

Figure 1. Multi-stream video player.

To provide a better spatial view, DOTS also offers a 3D model of the building as a user interface for person tracking. The positions of tracked people are marked in the 3D space by displaying foreground pixels from camera views showing those people. The 3D camera can automatically follow a tracked person (see Figure 5).

DOTS provides summaries of detected events or faces in the form of web pages containing thumbnails of interesting regions in video images. Clicking on any of these thumbnail images causes the video player to skip to the corresponding time and to emphasize the camera view showing the selected event.

3.1 Camera Bank

The camera bank (left side of Figure 1) presents views of all cameras in low resolution and at a low frame rate. If there is insufficient room to display all views at once, the camera bank displays all cameras by scrolling through them at regular intervals. All displays are synchronized to the same playback position; skipping to a different position in the timeline controls all video displays. Users can manually select cameras in the camera bank to see a larger display at a higher frame rate in the main video playback area. The views in the camera bank include a small graph of the level of activity in that camera view over the last 30 minutes. Selecting a point in the graph causes the player to adjust to that point in time and display that camera view in the main viewer.

3.2 Timeline

Video playback is controlled by a non-linear timeline that is synchronized to all displayed video streams (see bottom of Figure 1). It uses a detailed linear scale for the video around the current playback position shown in yellow and a less detailed linear scale for the video far away from the playback position. A non-linear scale provides the transition between those parts of the timeline. The timeline is color-coded with denser areas of time being darker. Such a timeline can be used to access several days of video while still providing detailed control around the current playback position. Users may choose the level of detail or select a single, less detailed, linear scale. Controls let the user increase and decrease the playback speed and reverse the playback direction.

3.3 Event Navigation

Attached to the timeline are circles indicating detected events. Clicking on a circle navigates to the corresponding time and switches the main player to camera views appropriate for the event. If multiple events are too close together in the timeline, a single aggregate circle shows the number of grouped events. An event consists of a label, a start and end time, and one or more cameras that can view the event location. DOTS supports externally generated events such as from door sensors or RFID tags. We currently include non-video based events from detected keyboard activity on certain computers and from micro switches at doors.

Events, such as people entering and leaving the building, are detected with hot spot analysis and added to a database. DOTS provides two types of hot spots: motion-based and region-based. When the ratio of foreground to background pixels within a motion hot spot exceeds a user-specified threshold, an event is triggered. Region-based hotspots are triggered when a tracked object intersects the hot spot in a particular direction. Detection of doors opening and closing is performed by motion hotspots, preferably located so that they are not triggered by motion other than from the door of interest. DOTS associates objects with the appropriate event.

Also, we support users in interactively defining hot spots by letting them mark a region either in a video display or on the floor plan. In the latter case, we retrieve the times of all tracked object trajectories that intersect

the area in the floor plan and process them just like the times of the tracked regions. Such a user-defined hot spot generates a database query for all tracked regions that intersect the hot spot. In less than a second all tracking regions from the database that intersect the hot spot are retrieved, grouped into events, and displayed along the timeline.

3.4 Floor Plan

Security video installations often require security personnel to monitor hundreds of video streams. A floor-plan interface component enables security personnel to select which video streams to include in the multi-stream video player. The floor plan displays the location of each camera, its field of view, the cameras being shown in the main viewer, and the objects being tracked. The floor plan can be panned and zoomed for more detail.

Cameras are color coded for identification (see Figures 1 and 2). For each camera, a colored shaded area indicates what the camera can see, taking walls into considerations. In an earlier version, we only provided the shaded area to indicate the camera view direction. User studies indicated this was not sufficient. We then added the arrow to indicate the view direction.

When a user selects a set of cameras with the mouse, the camera video streams are displayed in the video player area. Clicking on a position away from a camera selects all the cameras that can see that part of the map. When video streams are selected for display in the video player area, the video stream display is animated from the map position to the video player area. A small fading keyframe is left behind to anchor the animation to the map. The keyframe and the video display are also connected via a wire frame during the animation to provide better orientation.

3.5 Playback by Object Manipulation

Scrubbing, a method of controlling the video frame time by mouse motion along a timeline or slider, is used for fine-level control. This interaction technique allows the precise positioning of the video at a point where objects or people in the video are at certain positions or moving in a particular way. Scrubbing and off-speed playback, such as slow motion or fast forward, are useful but have limitations. No single playback speed or scale factor for mapping mouse motion to time changes is appropriate for all tasks or all portions of video. Rather than indirect

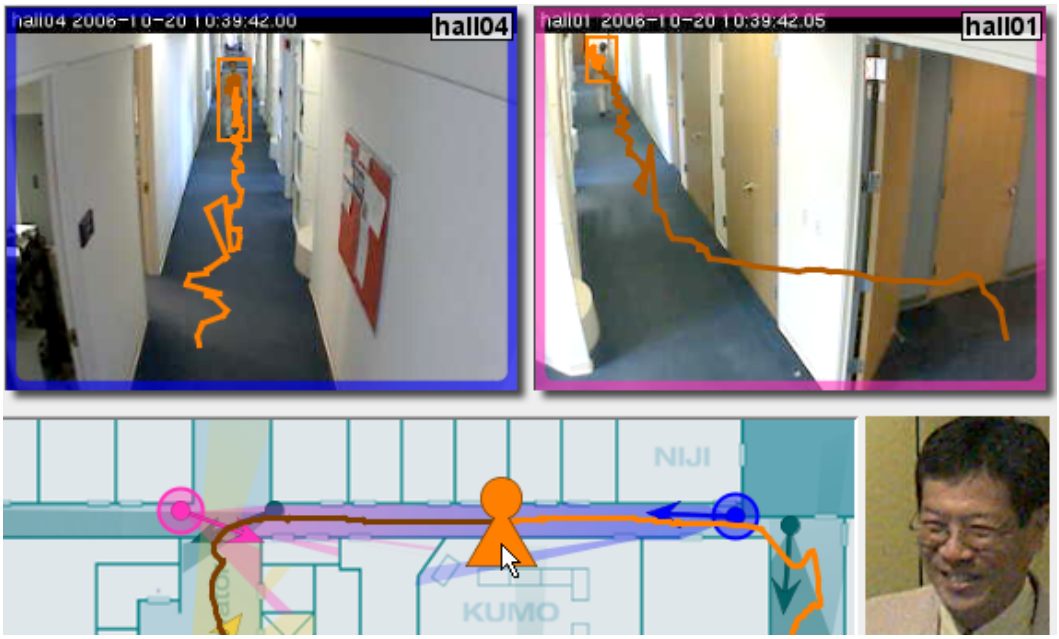

Figure 2. Video playback control by dragging on the floor plan.

scrubbing control by multiple sliders, a more natural interface is to let users directly grab and move objects along their trails. Within DOTS, objects may be grabbed and moved in a video window or floor plan view that schematically indicates positions of people by icons (see Figure 2) [17]. For example, a user may drag a person along their trail to view video at any given point, or drag a parked car to move to the points in the video where the car was parked. Such a user interface shifts the user experience from passively watching time-based video to directly interacting with it. Given the object trail information, whether in a single-camera view or a floor-plan view, a user can move objects to different points on their trails with the mouse. The object motion is constrained by the object trail such that the object can only be moved to locations where it was observed at some time. It is often desirable to determine when an object reached a particular point or to see all objects that were near that point. Right-clicking on a position displays a radial menu presenting the candidates (see Figure 3). For each candidate, the video image is cropped to the outline of the object to provide a good view. After selecting one of the candidates, the corresponding object is selected and the video display skips to the corresponding time.

3.6 Main Player Area

The main player area displays one or more video streams at high frame rates. The size of a video stream display indicates its relative importance as determined by either the user or the system. Importance decays exponentially over time so that video streams stay in view for some time after they become important. Otherwise, cameras in the main viewer get replaced too quickly. Video streams are animated out of the way as a new video stream becomes important.

Users can switch between automatic and non-automatic selection modes and can override the automatic selection at any time. Our automatic view-switching is similar to Wang et al.' s use of experiential sampling to

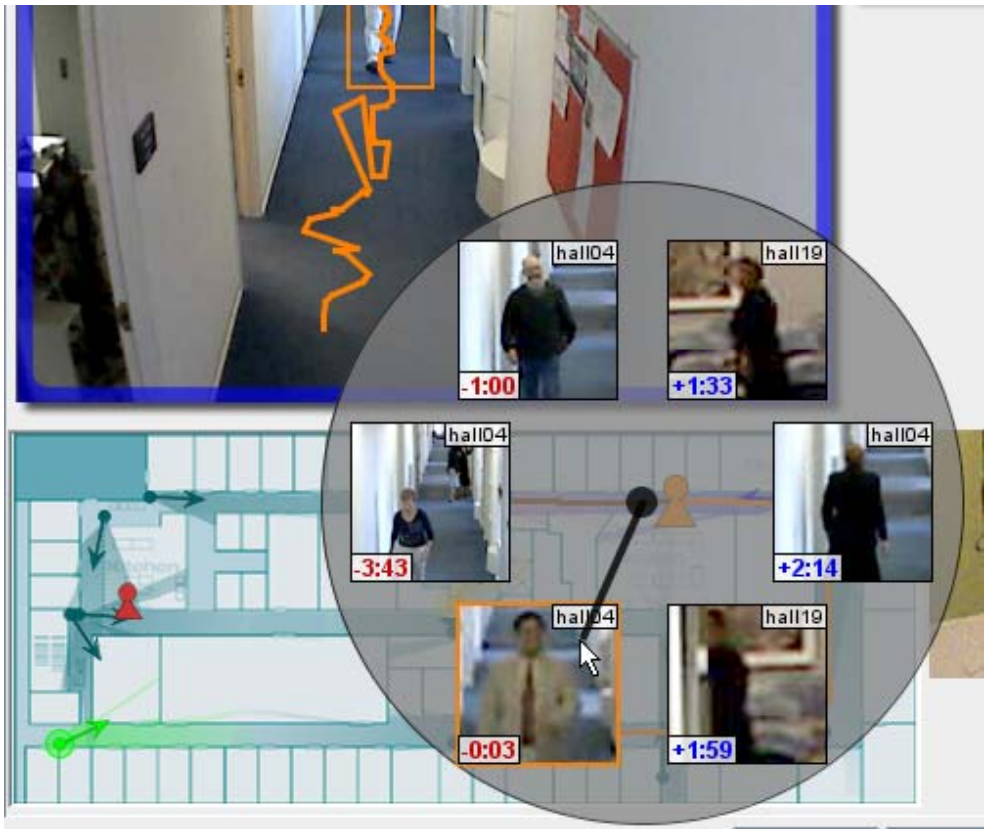

Figure 3. Pie menu to select a nearby person to be tracked. The numbers in the bottom-left indicate the relative time.

switch the view of surveillance cameras [33]. However, they do not describe a user interface with that feature. In addition to the manual mode, our system provides three automatic modes. The first mode shows the camera views that have the most activity as detected by the foreground segmentation. With much activity, this mode can be very busy. The second mode keeps a tracked person in view and switches to different camera views when the person enters those views. Finally, in the absence of user interaction or unusual events, the system enters a mode similar to a screen saver where cameras are periodically selected in a spatial order resembling the rounds of a security guard.

3.7 Spatial Multi-Video Player

When manually tracking a person walking from camera view to camera view, it is difficult for users to predict the camera view in which a tracked person might appear after walking out of the main camera view. Using the floor plan or a camera bank grouped by spatial proximity can help, but with many cameras, the images in the camera bank tend to be small so that it can be difficult for the users to locate and recognize a tracked person in those images.

To better support this task, we created an alternative to the main player area. The Spatial Multi-Video (SMV) player selects and organizes its contents primarily based on geographic relations between the main camera and the other cameras (see Figure 4). Rather than displaying all camera views, only views in close proximity are shown. Multiple smaller views surround the central view; a person walking out of the main camera's view will likely appear in the camera view adjacent to the direction they walked out. Users may click on any of the displayed video streams to select a new main camera view. Using this technique, users can follow activity from camera view to camera view. When changing the camera view, the movement is animated around the perimeter of the main view to indicate the view rotation and to keep the user oriented.

When using a map in combination with the spatial display (see Figure 4), the map is centered on the newly selected camera and

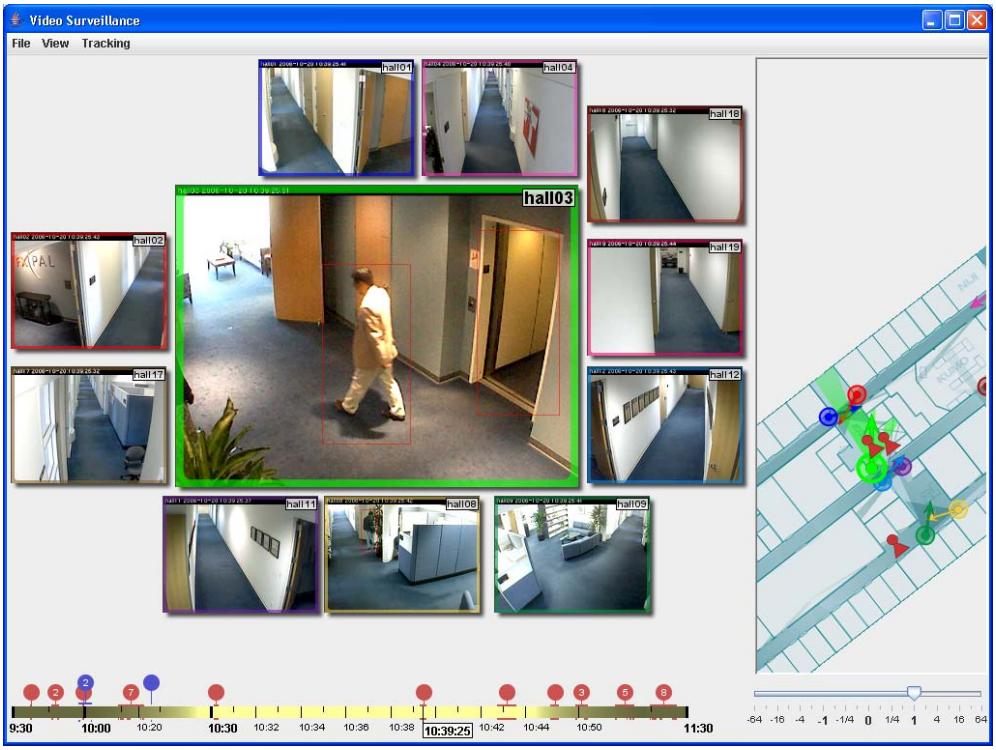

Figure 4. SMV player.

rotated such that the camera's view direction is to the top. The rotating map keeps the user oriented by simulating the way a hiker rotates a paper map to line up with a trail at a junction. The map movement is animated and the animation duration is synchronized to that of the peripheral camera views in the spatial display.

We conducted a user study with 16 participants that measured a subject's person tracking performance with the SMV player and with a traditional player with a camera bank and a single large display [10]. Both interfaces were tested with and without a floor plan. Spatial cues provided by either the floor plan or the SMV player led to superior performance. The floor plan was not needed when the SMV player was provided. Many of the study participants liked the floor plan that rotated synchronously with the SMV player. However, this feature did not provide a statistically significant advantage and was seen as distracting by some participants.

3.8 3D Video Player

The DOTS user interface suite includes a 3D viewer that displays segmented foreground regions of tracked people in a simple 3D model of the surveillance area. Foreground segments are shown as 'billboards' facing the virtual camera, placed at the tracked position of the person. Arbitrary viewpoints are supported, but two modes provide particularly useful mobile views. One places the virtual view at the position of a tracked person to help a surveillance user understand what the tracked person can see as they move. The other automatically chooses the best camera view of a tracked person. As the person moves around and the best camera view changes, the virtual view smoothly transitions to the next camera. Choosing a virtual view above and behind the camera helps users understand the spatial context of the camera. For example, Figure 5 shows a tracked person from a virtual position slightly above and behind the camera from which the image was taken (Figure 6).

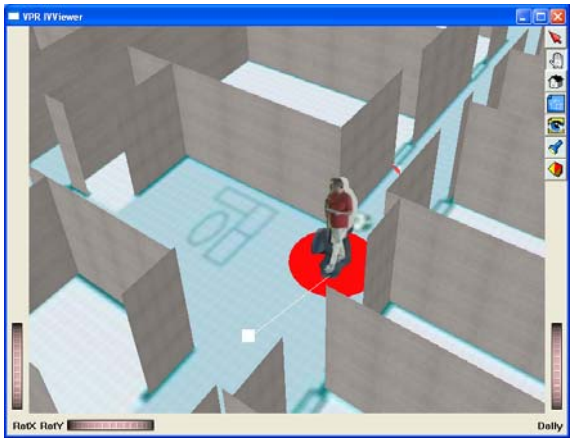

Figure 5. 3D viewer with foreground pixels.

Figure 6. Source of foreground pixels for 3D viewer.

Models are produced by a tool that allows tracing over a floor plan image to define walls, but they could be imported from CAD or architectural files. The floor is texture mapped with the building floor plan, and surfaces are currently given simple texture maps. Soon we will apply textures from the cameras onto the model, as is done in the video flashlight system [27]. The 3D model also helps the tracker handle situations in which people are semi-occluded, e.g., by cubicle walls. We found that in addition to being a good way to view video, the 3D viewer is useful for administration of DOTS; the viewer helps in selecting good positions for camera placement.

3.9 Summary Web Pages

DOTS provides three web-based graphical summaries: detected faces, regions intersecting with hotspots, and key frames for enter/exit events. Figure 7 shows thumbnails of the faces

extracted from the video images, as described in Section 5.5. For hot spot events, the web page shows the foreground region that intersected the hot spot. Users may request either a page showing images representing the bounding box of the foreground region or just the actual foreground pixels. For the latter case, the DOTS web server generates PNG images where all background pixels are transparent. In the case of door events (described in Section 3.3), a region is chosen based on several heuristics. For example, regions containing motion towards the camera from which they were captured are preferred over regions containing the opposite motion.

All summary web pages can control the video playback of an associated video player. By default, the associated player is assumed to have the same IP address as the web browser. It is also possible to specify the IP address of the video player as a parameter for the summary page. A mouse click on any image in the summary page causes the video player to skip to the corresponding time and to emphasize the camera views showing the selected event.

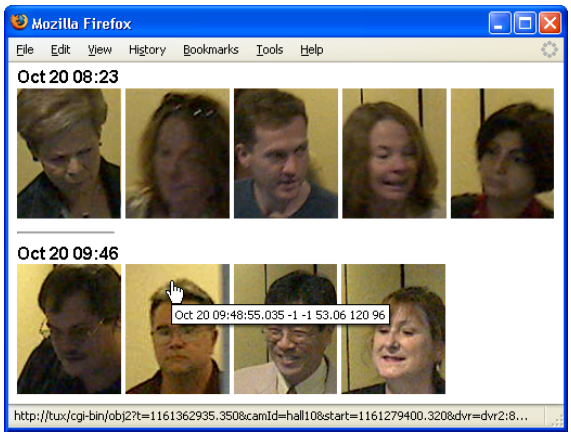

Figure 7. Summary web page with detected faces.

4. Architecture

Our infrastructure is composed of digital network cameras in hallways and other public places, a network video recorder (NVR), a database server that stores meta-information about the video, a web server delivering cropped images and summary pages, and several analysis and user interface system components. Figure 8 shows the software and hardware architecture of the system. The core of the system is a server that hosts the NVR, the database, and the web server. The NVR retrieves video from the network video cameras. The video analysis components run on several PCs, retrieve live video data from the NVR, and store the analysis results in the database. Finally, user interface components access and display data from the NVR, the database, and the web server. The architecture is designed to be open and scalable. Cameras can be added, and multiple user interfaces and new analysis modules can run simultaneously.

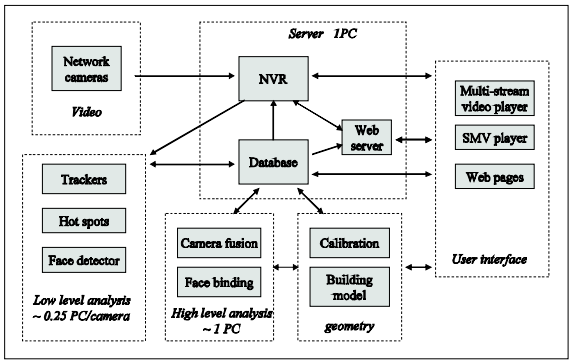

Figure 8. DOTS architecture.

4.1 Recorded Video

The NVR provides an HTTP-based API for synchronizing the playback of several video streams, for modifying the playback speed and direction, and for administrative tasks. A server with two single-core 3.2 GHz Intel Xeon processors, 4 GB RAM, and eight 400 GB hard disks in a non-RAID configuration runs the Java NVR software under Linux.

The NVR acts as a distributor of live video that gives large numbers of clients access to the same set of video streams. This design provides a major benefit over accessing the cameras directly because camera frame rates drop as more clients access them. Parallel access to recorded video is limited by disk seek latency. The video feeds are divided across the disks such that each disk records data from about three cameras to reduce disk seeks. While many clients can access video,

concurrent access of video recorded at different times can lead to decreased performance. We experimented with our own buffering scheme but found that the file system cache provided sufficient buffering.

We record video from Axis 210A IP cameras [5] that provide video as Motion JPEG via HTTP or as MPEG-4 streams via RTP. We focussed on Motion JPEG because it is simpler to process and provides better support for seeking frames at different times. Also, Motion JPEG does not require a dedicated codec on the client side so that our Java user interface can provide animated video displays.

Currently, we record video from 23 cameras at 15 frames per second at both 640x480 and 320x240 resolution. The cameras can provide 30 frames per second in a single resolution but we decided against the higher storage and network bandwidth requirements. Recording two streams from each camera simultaneously avoids the need for the NVR to decode JPEG images and saves bandwidth and decoding time for clients needing lower-resolution images. This design results in about 40 GB per camera per 24 hours (70% for the larger resolution), requiring close to 1 TB per day in storage for 23 cameras.

Using MPEG-4 would reduce the disk space and bandwidth requirements by a factor of 10. Unfortunately, that would make it computationally infeasible to decode all video streams at the same time in the user interface, even at low resolutions and frame rates. Furthermore, random access into the recorded video would be more difficult. To reduce storage requirements, we are developing a technique that reduces the frame rate or quality of recorded video during less interesting periods. This provides high frame rates at interesting parts and infrequently updated views during periods of inactivity. This technique is similar to approaches described by Korshunov and Ooi [18] and Pillai et al. [25]. Another possibility is to transcode the video images to a format such as MPEG-4 with the caveats mentioned above.

4.2 Video Access and Synchronization

The NVR maintains named playback cursors to control video playback and to synchronize the playback of several video streams. All video streams using the same cursor are synchronized. Clients on different systems may use the same cursor to show the same video (e.g., in a multi-screen environment). Modifying the playback position of a cursor effectively provides a remote control for video players. The playback cursor also encapsulates playback speed and direction. We successfully played multiple video streams at up to 1000 times the normal speed before the frame rate degraded. Such high speeds are useful for skimming video. Playback cursors may also maintain a constant offset to other playback cursors and thus provide a temporal context, for example, when tracking a person.

5. Object Tracking

DOTS detects foreground objects in a camera view by performing a foreground segmentation consisting of a pixel-level background modeling and a feature-level subtraction approach. It incorporates an efficient greedy-search approach for tracking multiple people through occlusion. Our system uses the calibrated camera information and a model of the building geometry to estimate each object's position given the bounding box associated with the object. Motion vectors for objects are then computed based on the geometry provided by the camera-view information. Camera views in which the object will likely appear next are deduced using those vectors.

5.1 Single Camera Video Analysis

The analysis detects foreground regions in each video frame. For most purposes, the bounding box of a foreground region is sufficient. For each detected region, we store in the database the frame time, the bounding box, and a 1-bit PNG image representing the

mask of the foreground pixels inside the bounding box.

The first processing step segments objects from the background using a Gaussian mixture model for pixel-level background modeling [29]. We convert color images to grayscale prior to computing the model, so both color or grayscale images are handled. The model is adaptively updated with information from new frames.

Feature-level subtraction is used for foreground segmentation that is robust to shadows, similar colors, and illumination changes. First, we use the foreground density around each pixel to determine a candidate set of foreground pixels. If more than 10% of the pixels in a local region are labeled as foreground pixels based on pixel-level background subtraction [29], the pixel centered in the region is considered as a candidate foreground pixel. Then, we compute the similarity of the neighborhood foreground and background images centered around each candidate foreground pixel using a normalized cross-correlation. The integral image method [20] is used for efficiency. For each neighborhood with low similarity, the pixel centered in the neighborhood is labeled as foreground. Figure 9 illustrates stages of the segmentation algorithm in situations with similar color and with shadows.

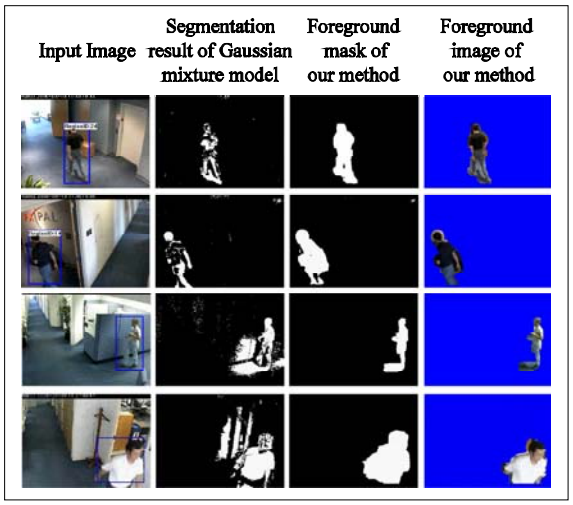

Figure 9. Foreground segmentation.

5.2 Tracking Humans Through Occlusion

Techniques for tracking multiple occluded humans can be categorized as either merge-split (MS) or straight-through (ST). In the MS approach, overlapping objects are encapsulated into a new group. When one of the objects splits from the group, its identity is reestablished using appearance features [21] such as color, texture and shape. When the number of objects in a group is larger than two, the MS method frequently fails because it is hard to tell how many objects are inside each new blob when splitting occurs. In the ST method, individual objects are tracked through the occlusion. Many ST methods [7] adopt bottom-up segmentation and use an appearance model to assign each pixel to a certain tracker.

We treat the problem of single-camera object tracking through occlusion as a track-based segmentation problem in the joint object space. First, a data association module classifies complex interactions between moving blobs. For single camera tracking, we build on prior work [7] using a correspondence matrix to classify an object's interactions with other objects into five classes: appear, disappear, continue, merge, and split. Merges and splits are entered into the database so that if tracked object regions are merged to form new regions, and subsequently split to form other regions, it is possible to determine which future regions are descendents and, hence, candidates to be the tracked object.

Next, identity maintenance is handled by a track-based Bayesian segmentation algorithm using appearance features. A greedy-search-based, occlusion-handling module decodes in real-time the best configuration of occluded objects in a group. Appearance features computed during tracking are used to estimate the matching scores in a Bayesian framework. When a merge occurs, the objects involved are identified. The search for the most probable configuration becomes a maximum likelihood estimation problem.

The position of each object is identified one at a time in a greedy manner. At each step, the location of the object with the highest observation probability, generally the object closest to the camera, is determined, and the computed location is used in the next step when computing the observation probability of each of the remaining objects. We estimate the observation probabilities based on the color histograms of the respective objects. Figure 10 illustrates the results of this process with several tracked objects.

Figure 10. Objects maintain their identities through occlusion.

5.3 World Coordinates

In our installation, cameras are mounted near the ceiling with oblique downward views. We measured the location of each camera in three dimensions. We also estimated pan, tilt, yaw, and field of view by matching up well-known points in the world (e.g., corners of walls) with their views in a camera. Our system currently does not support PTZ cameras but those could be supported if they either reported their orientation or if we maintained a collection of camera images associated with previously calibrated orientations that could be compared to the current camera image.

Our system uses calibrated camera information and a model of the building geometry to estimate each object's position given the bounding box associated with the object. We focus on tracking objects that move along the floor such as people and carts. Given the position of a bounding box within an image, the system determines how likely it is that the object's top, bottom, or both, are visible. For example, if the bounding box is flush against the bottom of a camera view, or against a surface known to be the top of a cubicle wall, it is likely that the feet are not visible (Figure 11). An additional complication in Figure 11 is that the person on the left is sitting and thus an estimate of the height is more difficult.

When an object's bottom point is visible, the projection of this point onto the floor plane gives a good estimate of the object's position. When the bottom is not visible, an estimate of the object's height and the image position of

its top give an estimate. Objects' heights may be estimated when both their top and bottom are visible. False results, such as an object's reflection in a framed picture, can be filtered out by detecting that the bottom of a bounding box intersects a wall rather than the floor.

Figure 11. Estimating locations of partially visible people.

5.4 Multiple Camera Tracking

Once objects are tracked in single-camera views, camera handoff determines the likelihood that multiple tracks result from the same object [35]. Based on the relations among camera views, the handoff methods can be roughly classified as overlapping views [16] and non-overlapping views [26]. Our system uses a new handoff function based on a spatial-temporal matching ratio for all pairs of tracks in two views. The camera handoff module operates in two steps: initialization and track matching.

In the initialization step, camera calibration is performed and the camera streams are synchronized temporally across the local network by the NVR. The minimum and maximum transition time of an object passing from one camera to another at normal speed are automatically learned by single camera tracking results of the training data, in which one person freely walks across all cameras.

In the track matching step, once a new object appears in a certain camera, the handoff module will be triggered. Matching ratios between the newly detected track and each track in all connected cameras are computed. The track in a connected camera with maximum matching ratio is selected as a candidate for handoff. If the maximum matching ratio is larger than a threshold, the two tracks will be labeled as the same object.

For cameras with overlapping fields-of-view, the matching ratio is estimated as the fraction of time during the overlap interval that the tracks are within a world distance threshold. In our surveillance system, people may pass from one camera view to another through "blind" regions in which they cannot be seen. In those cases, previously learned times between cameras are used to match tracks.

5.5 Face Detection and Face Binding

DOTS uses a face detection algorithm to capture faces of people in strategic areas, such as entrances and exits. Detected faces are associated to the objects representing tracked people using spatial and temporal proximity. Where multiple faces are captured for the same person, the sharpest image, the one with the most high-frequency content, is chosen. By binding the face with the object, DOTS provides users with a facial view of tracked objects even when no such view is or was available in the current camera view (see bottom-right of Figure 1).

6. Experiences and Evolution

The design of DOTS was influenced by other video analysis projects at our lab. One project involved collecting video from two Japanese post offices for one week each to support business process analysis. In that project, we had difficulties with the different tools for video recording and analysis. Many of these issues are handled much better in DOTS because of its improved NVR, better database structure, and improved user interface. DOTS will make it easier to undertake similar projects in the future. We also gained much experience during the year DOTS has been in constant use in our lab. In response, we made many changes to improve performance in all

components of the system. We divide these insights into privacy consideration, user interface, data storage, object tracking, and camera adjustments.

6.1 Privacy Considerations

When our cameras were installed, many of our staff had concerns with privacy, video capture with cameras and storage for long periods, and use of the video data for research and general surveillance. As a result, we held discussion with our staff whose input and feedback led to the creation of a usage policy and video capture guidelines. Use was solely for research purposes with access to video limited to researchers working on the system. In addition, everyone signed a consent form indicating their permissions to record their image for research and to use that video in presentations. Options were "Yes", "Yes, with review before use externally", or "No". Permission to record for research purposes was given by 98% of the entire staff. 93% of the entire staff consented to our use of the data in presentations.

The camera positions were selected to get adequate coverage of the public spaces of the building without recording in any private spaces, such as offices. Video was deleted from the DVR 24 to 31 hours after it was captured, unless we alerted personnel that a recording would be saved for a longer period. Notification that the recording would be saved included when and why the video would be recorded (e.g., for a demo or for a research study). This allowed people who were not comfortable with being recorded to take measures so that they were not included in video saved beyond the normal period.

6.2 User Interface

DOTS began with a relatively simple interface with VCR-like controls for watching synchronized video feeds. Given the heuristic nature of the object recognition and tracking data, early on, we were often left scanning long periods of video to determine whether the system was working as desired.

As a result of the desire to quickly navigate to periods of recognized activity, we added additional navigational abilities. The activity graphs for individual cameras in the camera bank is one such example. These graphs allow users to jump to a location and time by clicking on the small graph and then the use of the main timeline for finer-grained movement in time. The popup menus for locating tracks going through a location on the map are another interface for locating recent activity in a particular location.

Originally the user interface did not include a map. That made it difficult to understand where each camera was within the space being observed. Focusing on a particular clip, as presented by the video player, was easy. Some context was provided by including a bank of cameras, similar to those found it the current incarnation. Adding a map and then a timeline greatly improved the context problem but increased the complexity of the user interface, making attention to a particular video stream more difficult. As more functionality was added, the potential for visual overload increased. A number of visual cues were used to aid comprehension and tracking. First, color and synchronous motion are used throughout the user interface to link related camera views and tracked objects. Second, we used wire frames and persistent traces to anchor synchronous movement of tracked objects through the visualizations. Finally, the cameras, video player, and video bank are also synchronized and related by color.

6.3 Data Storage

In the post office project, metadata from different analysis tools and trackers was kept in separate files that were hard to maintain and did not support efficient or flexible queries for analysis. That experience led us to store DOTS' analysis results in a relational database rather than a separate file for each video stream, e.g., in MPEG-7 format. This

design decision allows fast and incremental access to the data and supports additional visualizations such as displaying the count of the tracked people over time.

Reducing the latency imposed by the database required several changes to the database design. The analysis components insert millions of records each day that are synchronized to the video frame rate. Initially, we used MySQL's MyISAM storage engine [22]. This engine locks whole tables, so parallel inserts into a table while other applications retrieve data from the same table caused bottlenecks. Switching to the InnoDB engine with row-level locking addressed some of those issues. Proper index design was important: superfluous indices lead the query optimizer astray and in general slow down inserts. Since addressing those issues, the database has provided a convenient means to store and access all data related to our video surveillance system.

6.4 Object Tracking

Many environmental conditions needed to be handled by the tracker. For example, initial versions of the tracker separated people's feet from their upper bodies due to similarity in carpet color and commonly worn blue jeans. Each outer office has windows, so that as people walk by the doorways, a shadow would be cast on the opposite wall, that would be mistakenly identified as foreground. We handled these and related problems by using local information around a pixel to identify foreground, in combination with geometry information to constrain a person's location. We also developed a real-time method that handles occlusion for small numbers of people. Future research is needed to handle larger numbers of people, such as when a meeting lets out.

In object tracking for a real-time system, there is a trade-off between low latency and greater accuracy. Low latencies are possible by maintaining an instantaneous hypothesis about the state of the world. Greater accuracy, however, can arise from reasoning over longer periods of time. To address this trade-off, we do two things. First, the DOTS analysis modules immediately enter object tracking hypotheses into the database, but update the tables when the hypotheses change. Second, the user interface regularly polls the database for changes in tracking metadata. For example, when the elevator door opens and a person emerges, a single region around the person and the door appears; both the person and the door are considered one "object." As the person walks away, a new region appears around the person, while the initial region remains around the door. When the initial hypothesis grouping the person with the elevator door becomes invalid, the tracking module reassigns the object identifiers of the two objects.

The object tracking problem at the elevator door also causes problems for the face-binding component. The delayed separation of a person from the elevator door causes a corresponding delay from the face detection to the availability of a corresponding object trajectory. The face-binding component is configured with empirically defined heuristics to cope with this, but there are occasions when there is ambiguity about the associations of faces to object trajectories. We are currently investigating modifications to the Gaussian mixture model to persist the Gaussian models of the background with the elevator door open, that typically only happens for a few seconds. It is anticipated that this will allow tracking of people from the moment that they are first visible.

6.5 Camera Adjustments

Most of our cameras are attached to 4-inch stands sticking through ceiling tiles. While the joint at the end of the stand is securely locked, the stand can rotate at its base. Thus, cameras occasionally rotate by small amounts after having been touched, e.g., by cleaning personnel. Such camera adjustments cause

problems with tracking such that people suddenly appear to walk through walls. We have addressed this problem in two ways. First, we created a user interface to simplify camera calibration. This interface is based on our video player and superimposes wall outlines on the video image. Seven sliders allow changes to the camera position and orientation that immediately update the wall-outline overlay. Second, we save reference images from every camera several times a day. Those images are submitted to a web service that determines the best affine transform between subsequent images by matching corresponding points. Administrators are alerted via email if there is a change. We are also working on using the affine transform to automatically update the camera orientation.

7. Conclusions

DOTS is an integrated system for office video surveillance. It combines an infrastructure for recording network video cameras with a video analysis component for detecting events and tracking people, and a user interface that can quickly access recorded and live video. DOTS uses the results of the video analysis to guide users' attention to interesting events for more effective monitoring in systems with many video streams.

We gained a better understanding of issues in video surveillance through the year of DOTS' deployment in our office. We improved our analysis methods and the user interface, but additional work remains. For example, we are working on detecting higher-level events such as unusual behavior, fights or falls.

Recently, we installed DOTS at a second site that offers new challenges. The second installation has cubicles with low walls in contrast to hallways and offices at the original site. This environment requires us to refine the person tracking to handle partial visibility of persons. Such environmental differences highlight the need for further work to make DOTS more robust to new conditions.

References

[1] 3VR Security. http://www.3vr.com/

[2] F. Arman, R. Depommier, A. Hsu, and M.-Y. Chiu, Content-based browsing of video sequences. In Proc. ACM Multimedia 94, San Francisco, CA, pp. 97-103, 1994.

[3] E. Ardizzone, M. La Cascia, G. Lo Re, and M. Ortolani. An integrated architecture for surveillance and monitoring in an archaeological site. Proc. ACM Workshop on Video Surveillance and Sensor Networks, pp. 79-86, 2005.

[4] P.K. Atrey, M.S. Kankanhalli, and R. Jain. Timeline-based information assimilation in multimedia surveillance and monitoring systems. Proc. ACM Workshop on Video Surveillance and Sensor Networks, pp. 103-112, 2005.

[5] Axis Communications. http://www.axis.com/

[6] R. Cucchiara. Multimedia surveillance systems. Proc. ACM Workshop on Video Surveillance and Sensor Networks, pp. 3-10, 2005.

[7] R. Cucchiara, C. Grana, and G. Tardini. Track-based and object-based occlusion for people tracking refinement in indoor surveillance. Proc. ACM 2nd International Workshop on Video Surveillance & Sensor Networks, pp. 81-87, 2004.

[8] G. Daniel and M. Chen. Video Visualization. Proc. IEEE Visualization 2003, pp. 409-416, 2003.

[9] Energy Control, Inc. http://energyctrl.com/security.htm

[10] A. Girgensohn, F. Shipman, T. Turner, and L. Wilcox. Effects of presenting geographic context on tracking activity between cameras. Proc. SIGCHI Conference on Human Factors in

Computing Systems, pp. 1167-1176, 2007.
[11] Guardian Solutions. http://www.guardiansolutions.com/
[12] A. Hampapur, L.M. Brown, J. Connell, M. Lu, H. Merkl, S. Pankanti, A.W. Senior, C.-F. Shu, and Y.-L. Tian. Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking. IEEE Signal Processing Magazine, Vol. 22, No. 2, pp. 38-51, 2005.
[13] L. Huston, R. Sukthankar, J. Campbell, and P. Pillai. Forensic video reconstruction. Proc. ACM Workshop on Video Surveillance and Sensor Networks, pp. 20-28, 2004.
[14] G. Iannizzotto, C. Costanzo, F. La Rosa, and P. Lanzafame. A multimodal perceptual user interface for video-surveillance environments. Proc. Conference on Multimodal Interfaces, pp. 45-52, 2005.
[15] Johnson Controls. http://www.johnsoncontrols.com/security/Default.htm
[16] S. Khan and M. Shah. Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. Proc. IEEE Trans. on Pattern Analysis and Machine Intelligence, Volume 25, No. 10, pages 1355-1360, October 2003.
[17] D. Kimber, T. Dunnigan, A. Girgensohn, F. Shipman, T. Turner, and T. Yang. Trailblazing: video playback control by direct object manipulation. ICME, 2007.
[18] P. Korshunov and W.T. Ooi. Critical video quality for distributed automated video surveillance. Proc. ACM Multimedia, pp. 151-160, 2005.
[19] L3 Communications. http://www.l3praetorian.com/
[20] J.P. Lewis. Fast normalized cross-correlation. In Vision Interface, 1995.
[21] S. McKenna, S. Jabri, Z. Duric, and H. Wechsler. Tracking Groups of People. Proc. Computer Vision and Image Understanding, 2000.
[22] MySQL. http://www.mysql.com/
[23] ObjectVideo. http://www.objectvideo.com/products/
[24] Panasonic. Security Products. http://www.panasonic.com/business/security/
[25] P. Pillai, Y. Ke, and J. Campbell. Multi-fidelity storage. Proc. ACM Workshop on Video Surveillance and Sensor Networks, pp. 72-79, 2004.
[26] F. Porikli and A. Divakaran. Multi-camera calibration, object tracking and query generation. Proc. IEEE International Conference on Multimedia and Expo, Vol. 1, pages 653-656, July 2003.
[27] H.S. Sawhney, A. Arpa, R. Kumar, S. Samarasekera, M. Aggarwal, S. Hsu, D. Nister, and K. Hanna. Video flashlights: real time rendering of multiple videos for immersive model visualization. Proceedings of the 13th Eurographics Workshop on Rendering, pp. 157-168, 2002.
[28] I.O. Sebe, J. Hu, S. You, and U. Neumann. 3D video surveillance with augmented virtual environments. Proc. ACM Workshop on Video Surveillance and Sensor Networks, pp. 107-112, 2003.
[29] C. Stauffer, W. Eric, and L. Grimson. Learning patterns of activity using real-time tracking. IEEE Trans. on Pattern Analysis and Machine Intelligence, Volume 22, Issue 8, pp. 747-757, 2000.
[30] Vidient. http://www.vidient.com/
[31] VistaScape Security System. http://www.vistascape.com/
[32] G. Wang, T.-T. Wong, and P.-A. Heng. Real-time surveillance video display with salience. Proc. ACM Workshop on Video Surveillance and Sensor Networks, pp. 37-44, 2005.
[33] J. Wang, M.S. Kankanhalli, W. Yan, and R. Jain. Experiential sampling for video surveillance. Proc. ACM Workshop on

Video Surveillance and Sensor Networks, pp. 77-86, 2003.

[34] G. Wu, Y. Wu, L. Jiao, Y.-F. Wang, and E.Y. Chang. Multi-camera spatio-temporal fusion and biased sequence-data learning for security surveillance. Proc. ACM Multimedia, pp. 528-538, 2003.

[35] T. Yang, F. Chen, D. Kimber, and J. Vaughan. Robust people detection and tracking in a multi-camera indoor visual surveillance system. ICME, 2007.

Author's Introduction

Andreas Girgensohn

Andreas Girgensohn is a researcher in the Distributed Collaboration group at FX Palo Alto Laboratory. He is involved in several projects concerned with the design, development, deployment, and evaluation of applications for video monitoring, editing, and indexing and for organizing and presenting digital photos. Andreas received a Ph.D. in computer science from the University of Colorado at Boulder.