Image Understanding Workshop

Proceedings of a Workshop
held in
Monterey, California

November 20–23, 1998

Volume I

Sponsored by:
Defense Advanced Research Projects Agency
Information Systems Office

This document contains copies of reports prepared for the DARPA Image Understanding Workshop. Included are Principal Investigator reports and technical results from the basic and strategic computing programs within DARPA/ISO-sponsored projects and certain technical reports from selected scientists from other organizations.

Distributed by:
Morgan Kaufmann Publishers Inc.
340 Pine Street, 6th Floor
San Francisco, Calif. 94104-3205
ISBN: 1–55860–583–5
Printed in the United States of America

Event Recognition and Reliability Improvements for the Autonomous Video Surveillance System

Frank Z. Brill, Thomas J. Olson, and Christopher Tserng
Texas Instruments
P.O. Box 655303, MS 8374, Dallas, TX 75265
brill@csc.ti.com, olson@csc.ti.com, tserng@csc.ti.com

Abstract

This report describes recent progress in the development of the Autonomous Video Surveillance (AVS) system, a general-purpose system for moving object detection and event recognition. AVS analyses live video of a scene and builds a description of the activity in that scene. The recent enhancements to AVS described in this report are: (1) use of collateral information sources, (2) camera hand-off, (3) vehicle event recognition, and (4) complex-event recognition. Also described is a new segmentation and tracking technique and an evaluation of AVS performing the best-view selection task.

1. Introduction

The Autonomous Video Surveillance (AVS) system processes live video streams from surveillance cameras to automatically produce a real-time map-based display of the locations of people, objects and events in a monitored region. The system allows a user to specify alarm conditions interactively, based on the locations of people and objects in the scene, the types of objects in the scene, the events in which the people and objects are involved, and the times at which the events occur. Furthermore, the user can specify the action to take when an alarm is triggered, e.g., to generate an audio alarm or write a log file. For example, the user can specify that an audio alarm should be triggered if a person deposits a briefcase on a given table between 5:00pm and 7:00am on a weeknight. Section 2 below describes recent enhancements to the AVS system. Section 3 describes progress in improving the reliability of segmentation and tracking. Section 4 describes an experiment that quantifies the performance of the AVS "best view selection" capability.

This research was sponsored in part by the DARPA Image Understanding Program.

2. New AVS functionality

The structure and function of the AVS system is described in detail in a previous IUW paper [Olson and Brill, 1997]. The primary purpose of the current paper is to describe recent enhancements to the AVS system. These enhancements are described in four sections below: (1) collateral information sources, (2) camera hand-off, (3) vehicle event recognition, and (4) complex-event recognition.

2.1. Collateral information sources

Figure 1 shows a diagram of the AVS system. One or more "smart" cameras process the video stream to recognize events. The resulting event streams are sent to a Video Surveillance Shell (VSS), which integrates the information and displays it on a map. The VSS can also generate alarms based on the information in the event streams. In recent work, the VSS was enhanced to accept information from other sources, or "recognition devices" which can identify the objects being reported on by the cameras. For example, a camera may report that there is a person near a door. A recognition device may report that the person near the door is Joe Smith. The recognition device may be a badge reader, a keypad in which a person types their PIN, a face recognition system, or other recognition system.

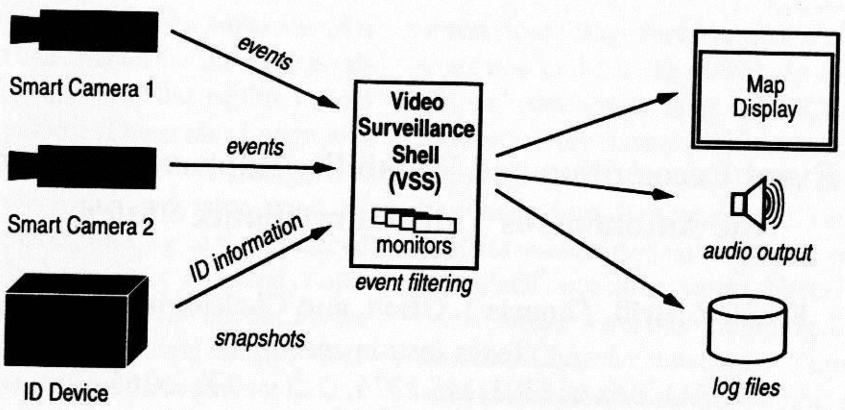


Figure 1: AVS system diagram

The recognition device we have incorporated is a voice verification system. The user stands in a predefined location in the room, and speaks his or her name. The system matches the utterance to previously captured examples of the person speaking their name, and reports to the VSS if there is a match. The VSS now knows the identity of the person being observed, and can customize alarms based on the person's identity.

A recognition device could identify things other than people, and could classify actions instead of objects. For example, the MIT Action Recognition System (MARS) recognizes actions of people in the scene, such as raising their arms or bending over. MARS is trained by observing examples of the action to be recognized and forming "temporal templates" that briefly describe the action [Davis and Bobick, 1997]. At run time, MARS observes the motion in the scene and determines when the motion matches one of the stored temporal templates. TI has obtained an evaluation copy of the MARS software and used it as an recognition device which identifies actions, and sends the result to the AVS VSS. We successfully trained MARS to recognize the actions of opening a door, and opening the drawer of a file cabinet. When MARS recognizes these actions, it sends a message to the AVS VSS, which can generate an appropriate alarm.

2.2. Camera hand-off

As depicted in Figure 1, the AVS system incorporates multiple cameras to enable surveillance of a wider area than can be monitored via a single camera. If the fields of view of these cameras are adjacent, a person can be tracked from one monitored area to another. When the person leaves the field of view of one camera and enters another, the process of maintaining the track from one camera view to another is termed *camera hand-off*. Figure 2 shows an area monitored by two cameras. Cam-

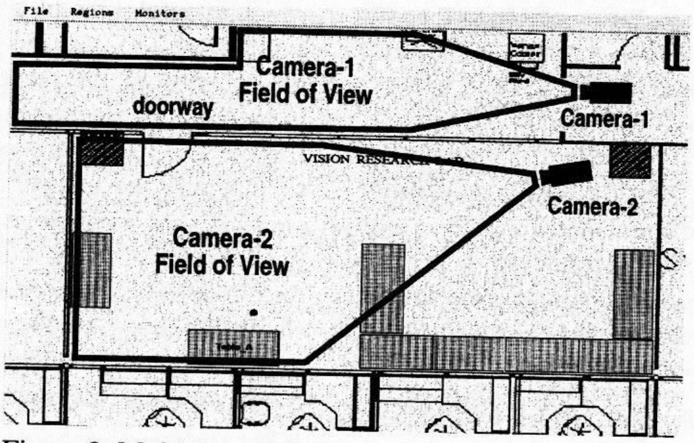


Figure 2: Multiple cameras with adjacent fields of view

era-1 monitors the hallway, and Camera-2 monitors the interior of the room. When a person moves through the doorway to enter the room from the hall or vice-versa, camera hand-off is necessary to enable the system to know that the person that was being monitored in the hall via Camera-1 is the *same* as the person being monitored in the room via Camera-2.

The AVS system accomplishes camera hand-off by integrating the information from the two cameras in the map coordinate system. The AVS "smart" cameras report the locations of the monitored objects and people in map coordinates, so that when the VSS receives reports about a person from two separate cameras, and both cameras are reporting the person's coordinates at about the same map location, the VSS can deduce that the two separate reports refer to the same person. In the example depicted in Figure 2, when a person is standing in the doorway, both cameras can see the person and report his or her location at nearly the same place. The VSS reports this as one person, using a minimum distance to allow for errors in location. When Camera-2 first sees a person at a location near the doorway and reports this to the VSS, the VSS checks to see if Camera-1 recently reported a person near the door. If so, the VSS reports the person in the room as the same one that Camera-1 had been tracking in the hall.

2.3. Vehicle event recognition

This section describes extensions to the existing AVS system that enable the recognition of events involving interactions of people with cars. These new capabilities enable smart security cameras to monitor streets, parking lots and driveways and report when suspicious events occur. For example, a smart camera signals an alarm when a person exits a car, deposits an object near a building, reenters the car, and drives away.

2.3.1. Scope and assumptions

Extending the AVS system to handle human-vehicle interactions reliably involved two separable subproblems. First, the system's vocabulary for events and objects must be extended to handle a new class of object (vehicle) and new event types. Second, the AVS moving object detection and tracking software must be modified to handle the outdoor environment, which features variable lighting, strong shadows, atmospheric disturbances, and dynamic backgrounds. The work described here in section 2.3 addresses the first problem, to extend the system for vehicle events in conditions of uniform overcast with little wind. Our approach to handling general outdoor lighting conditions is discussed in section 4.

The method is further specialized for imaging conditions in which:

1. The camera views cars laterally.
2. Cars are unoccluded by other cars.
3. When cars and people overlap, only one of the overlapping objects is moving
4. The events of interest are people getting into and out of cars.

2.3.2. Car detection

The first thing that was done to expand the event recognizing capability of the current system was to give the system the ability to distinguish between people and cars. The system classifies objects as cars by using their sizes and aspect ratios. The size of an object in feet is obtained using the AVS system's image coordinate to world coordinate mapping. Once the system has detected a car, it analyzes the motion graph to recognize new events.

2.3.3. Car event recognition

In principle, car exit and car entry events could be recognized by detecting characteristic interactions of blobs in difference images, in a manner similar to the way AVS recognizes DEPOSIT and REMOVE events. In early experiments, however, this method turned out to be unsatisfactory because the underlying motion segmentation method did not segment cars from people. Whenever the people pass near the car they appear to merge with it, and track is lost until they walk away from it.

To solve this problem, a new approach involving additional image differencing was developed. The technique allows objects to be detected and tracked even when their images overlap the image of the car. This method requires two reference images: one consists of the original background scene (background image), and the other is identical to the first except it includes the car. The system takes differences between the current video image and the original reference image as usual. However, it also differences the current video image with the reference image containing the car. This allows the system to detect objects which may be overlapping the car. Using this technique, it is easy to detect when people enter and exit a car. If an object disappears while overlapping with a car, it probably entered the car. Similarly, if an object appears overlapping a car, it probably exited the car.

2.3.4. Basic method

When a car comes to rest, the following steps are taken. First, the image of the car object is removed from its frame and stored. Then, the car image is merged with the background image, creating an updated reference image containing the car. (Terminology: a *reference car image* is the subregion of the updated reference image that contains the car.) Then, the *car background image*, the region of the original background image that is replaced by the car image, is stored.

For each successive frame, two difference images are generated. One difference image, the *foreground difference image*, is calculated by differencing the current video image with the updated reference image. The foreground difference image will contain all the blobs that represent objects other than the car, including ones that overlap the car. The second difference image, the *car difference image*, is calculated using the car background image. The car difference image is formed from the difference between the current frame and the car background image, and contains the large blob for the car itself. Figures 3 and 4 show the construction and use of these images.

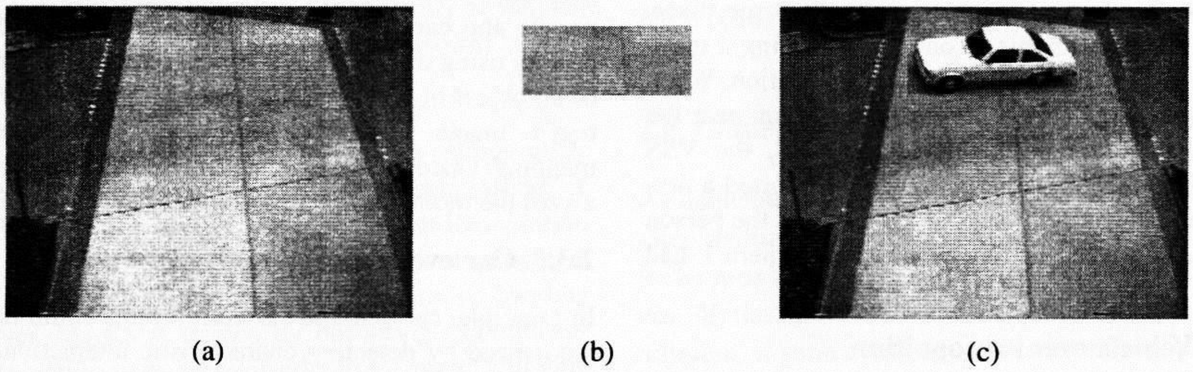


Figure 3: (a) Background image. (b) Car background image. (c) Updated reference image

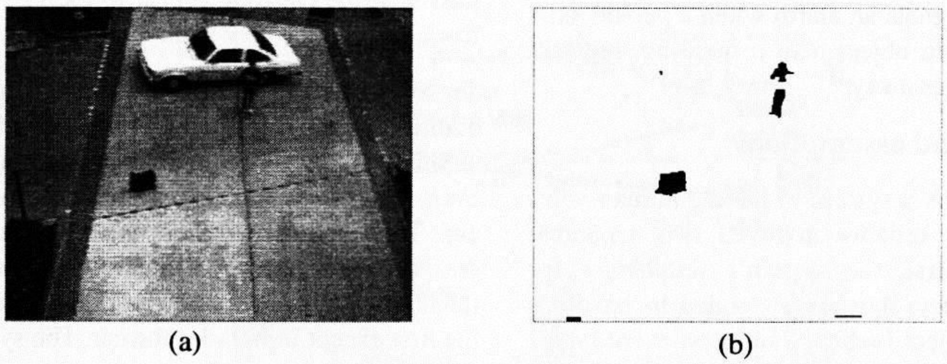


Figure 4: (a) Current video image. (b) Foreground difference image

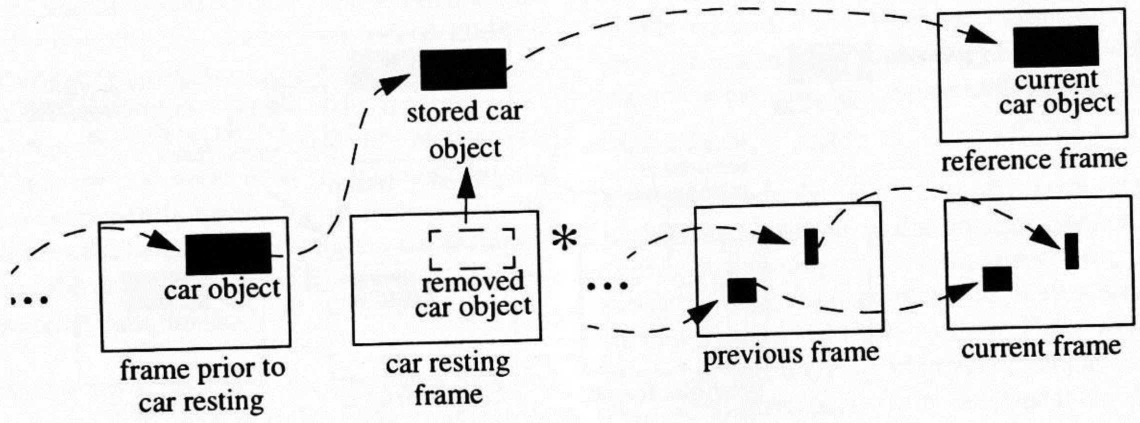


Figure 5: Creation of the motion graph.
The starred frame represents the frame prior to the background image being updated.

The blobs in the foreground difference image are grouped into objects using the normal grouping heuristics and placed in the current frame. The blobs in the car difference image necessarily represent the car, so they are all grouped into one current car object and placed in a special *reference frame*. Normal links occur between objects in the previous frame and objects in the current frame. Additionally, the stored car object, which was removed from its frame, (from Step 1) is linked to the current car object which is in the reference frame. In any given sequence, there is only one reference frame.

Figure 5 demonstrates the creation of this new motion graph. As indicated by the dotted lines, all objects maintain their tracks using this method. Notice that even though the car object disappears from future frames (due to the updated reference image), it is not detected to have exited because its track is maintained throughout every frame. Using this method, the system is able to keep track of the car object as well as any objects overlapping the car. If an object appears intersecting a car object, an INCAR event is reported. If an object disappears while intersecting a car object, an OUTCAR event is reported. Figure 6 shows the output of the system. The system will continue to operate in this manner until the car in the reference frame begins to move again.

When the car moves again, the system reverts to its normal single-reference-image state. The system detects the car's motion based on the movement of its centroid. It compares the position of the centroid of the stored car object with the centroid of the current car object. Figure 7 shows the slight movement of the car.

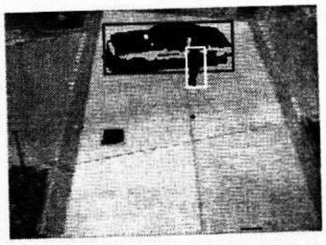


Figure 6: Final output of system

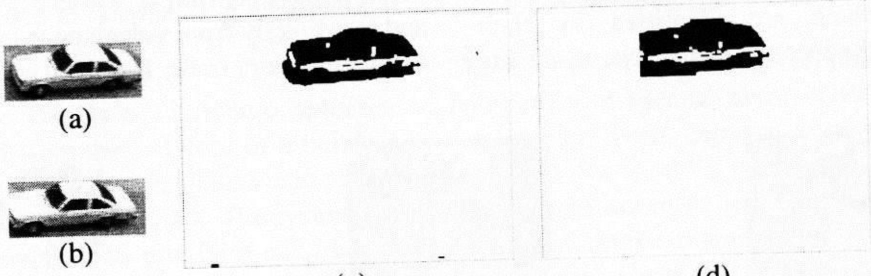


Figure 7: (a) Reference car image. (b) Moving car image.
(c) Reference car difference image. (d) Moving car difference image

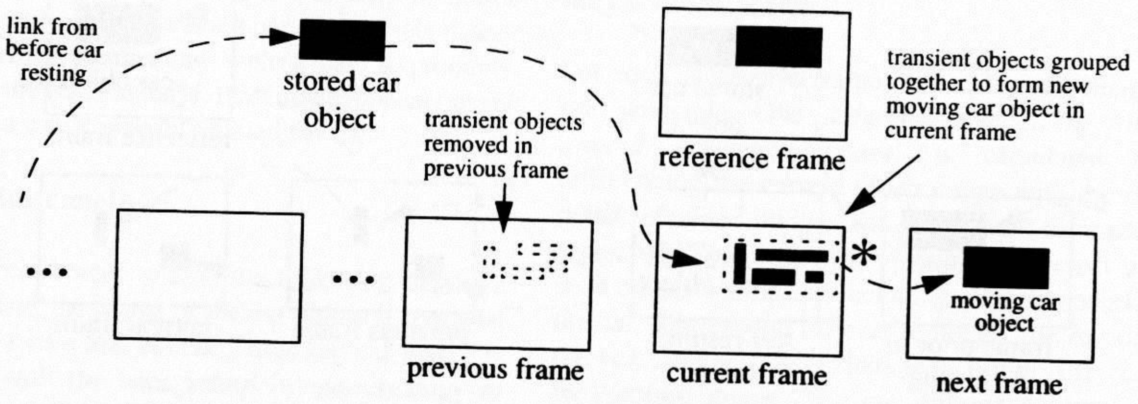

link from before car resting
stored car object
transient objects removed in previous frame
reference frame
transient objects grouped together to form new moving car object in current frame
...
...
*
moving car object
previous frame
current frame
next frame

Figure 8: Restoration of normal differencing. The starred frame represents the last frame prior to the original reference image being restored.

If the centroid locations differ by more than a threshold, the following sequence of events occur to restore the system to its original state:

1. An object representing the moving car is created in the current frame.
2. The stored car object is linked to this new moving car object in the current frame.
3. Objects in the previous frame that intersect the moving car are removed from that frame.
4. The car background image is merged with the updated reference image to restore the original reference image.
5. Normal differencing continues.

Figure 8 demonstrates how the system is restored to its original state. Note that there is one continuous track that represents the path of the car throughout.

When the car begins to move again, transient blobs appear in the foreground difference image due to the fact that the car is in the updated reference image as seen in Figure 9. Therefore, to create a new moving car object in the current frame, these transient objects, which are identified by their intersection with the location of the resting car, are grouped together as one car object. If there are no transient objects, a copy of the stored car object is inserted into the current frame. This way, there is definitely a car object in the current frame to link with the stored car object. Transient objects might also appear in the previous frame when a car is moving. Therefore, these transient objects must be removed from their frame in order to prevent them from being linked to the new moving car object that was just created in the current frame. After the steps described above occur, the system continues as usual until another car comes to rest.

2.3.5. Experiments: disk-based sequences

To test the principles behind the modified AVS system, three sequences of video that represented interesting events were captured to disk. These sequences represented events which the modified system should be able to recognize. Capturing the sequences to disk reduces noise and ensures that the system processes the same frames on every run, making the results deterministic. In addition to these sequences, longer sequences were recorded and run directly from videotape to test how the system would work under less ideal conditions.

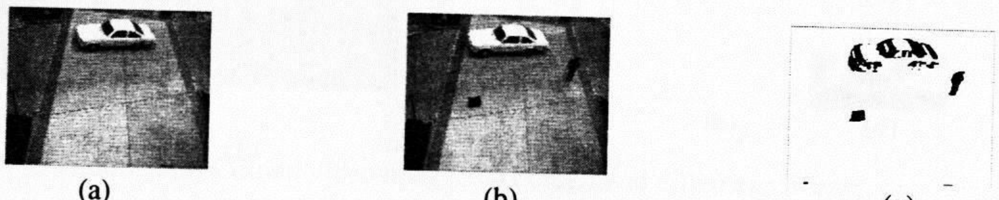


Figure 9: (a) Updated reference image. (b) Current video image. (c) Foreground difference image

2.3.5.1. Simple sequence. The first sequence was filmed from the 3rd story of an office building overlooking the driveway in front of the building. A car drives up and a person exits the car, walks away, deposits a briefcase, and finally reenters the car. Then, the car drives away. In this segment, the system successfully detects the person exiting the car. However, the person entering the car is missed because the person gets grouped with a second person walking near the car.

Further on in the sequence, the car drives up again and a person exits the car, walks away, removes the briefcase, and finally reenters the car. Again, the car drives away. In this segment, both the person entering and exiting the car are recognized. In both these sequences, there was only the one false negative mentioned earlier and no false positives.

2.3.5.2. Pickup sequence. This sequence was filmed in front of a house looking at the street in front of the house. In the sequence, a person walks into the scene and waits at the curb. A car drives up, picks up the person, and drives away. The system correctly detects the person entering the car. There are no false positives or negatives.

2.3.5.3. Drop off sequence. This sequence was filmed in the same location as the previous one. In this sequence, a car drives up and a person is dropped off. The car drives away with the person still standing in the same location. Then, the person walks off. The system correctly detects the person exiting the car and does not report a false enter event when the car moves away.

2.3.6. Experiments: videotaped sequences

These sequences were run on the system straight from videotape. These were all run at a higher threshold to accommodate noise on the videotape. However, this tended to decrease the performance of the system.

2.3.6.1. Dark day. This is a 15 minute sequence that was recorded from the 3rd floor of a building on a fairly dark day. In that time span, 8 cars passed through the camera's field of view. The system detected 6 cars correctly and one false car (due to people grouped together). One car that was not detected was due to its small size. The other car was undetected because the system slowed down (due to multiple events occurring) and missed the images with the car in them. In this sequence, two people entered a car. However, both events were missed because the car was not recognized as resting due to the dark lighting conditions on this rainy day.

2.3.6.2. Cloudy day. This is a 13 minute sequence in the same location as the previous sequence except it is a cloudy day. In this time span, 9 cars passed through the camera's field of view and all of them were detected by the system. There were a total of 2 people entering a car and 2 people exiting a car. The system successfully detected them all. Additionally, it incorrectly reported one person walking near a car as an instance of a person exiting a car.

2.3.6.3. Cloudy day—extended time. This is a 30 minute sequence in the same location as the previous two. In this time span, 28 cars pass through and all of them were detected. The system successfully detected one person exiting a car but missed two others. The two people were missed because the car was on the edge of the camera's field of view and so it was not recognized immediately as a car.

2.3.7. Evaluation of car-event recognition

The modified AVS system performs reasonably well on the test data. However, it has only been tested on a small number of videotaped sequences, in which much of the action was staged. Further experiments and further work with live, uncontrolled data will be required to make the system handle outdoor vehicle events as well as it handles indoor events. The technique of using multiple reference images is interesting and can be applied to other problems, e.g. handling repositioned furniture in indoor environments. For more detail on this method, see [Tserng, 1998].

2.4. Complex events

The AVS video monitoring technology enables the recognition of specific events such as when a person enters a room, deposits or picks up an object, or loiters for a while in a given area. Although these events are more sophisticated than those detected via simple motion detection, they are still unstructured events that are detected regardless of the context in which they occur. This can result in alarms being generated on events that are not of interest.

For example, if the system is monitoring a room or store with the intention of detecting theft, the system could be set up to generate an alarm whenever an object is picked up (i.e., whenever a REMOVE event occurs). However, no theft has occurred unless the person leaves the area with the object. A simple, unstructured event recognition system would generate an alarm every time someone picked up an object, resulting in many false alarms; whereas a system that can recognize complex events could be programmed to only generate an alarm when the REMOVE event is followed by an EXIT event. The EXIT event provides context for the REMOVE event that enables the system to filter out uninteresting cases in which the person does not leave the area with the object they picked up. This section describes the design and implementation of such a complex-event recognition system.

We use the term *simple event* to mean an unstructured atomic event. A *complex event* is structured, in that it is made up of one or more *sub-events*. The sub-events of a complex event may be simple events, or they may be complex, enabling the definition of event hierarchies. We will simply say *event* to refer to an event that may be either simple or complex. In our theft example above, REMOVE and EXIT are simple events, and THEFT is a complex event. A user may also define a further event, e.g., CRIME-SPREE, which may have one or more complex THEFT events as sub-events.

We created a user interface that enables definition of a complex event by constructing a list of sub-events. After one or more complex events have been defined, the sub-events of subsequently defined complex events can be complex events themselves.

2.4.1. Complex-event recognition

Once the user has defined the complex events and the actions to take when they occur, the event recognition system recognizes these events as they occur in the monitored area. For the purposes of this section, we assume *a priori* that the simple events can be recognized, and that the object involved in them can be tracked. In the implementation we will use the methods discussed in [Courtney, 1997, Olson and Brill, 1997] to track objects and recognize the simple events. In order to recognize a complex event, the system must keep a record of the sub-events that have occurred thus far, and the objects involved in them. Whenever the first sub-event in a complex event's sequence is recognized, an *activation* for that complex event is created. The activation contains the *ID* of the object involved in the event, and an *index*, which is the number of sub-events in the sequence that have been recognized thus far. The index is initialized to 1 when the activation is created, since the activation is only created when the first sub-event matches. The system maintains a list of current activations for each defined complex-event type. Whenever any new event is recognized, the list of current activations is consulted to see if the newly recognized (or *incoming*) event matches the next sub-event in the complex event. If so, the index is incremented. If the index reaches the total number of sub-events in the sequence, the complete complex event has been recognized, and any desired alarm can be generated. Also, since the complex event that was just recognized may also be a sub-event of another complex event, the activation lists are consulted again (recursively) to see if the indices of any other complex event activations can be advanced.

To return to our THEFT example, the complex THEFT event has two sub-events, REMOVE and EXIT. When a REMOVE event occurs, an activation for the THEFT event is created, containing the ID of the person involved in the REMOVE event, and an index set to 1. Later, when another event is recognized by the system, the activation is consulted to see if the event type of this new, incoming event matches the next sub-event in the sequence (in this case, EXIT). If the event type matches, the object ID is also checked, in this case to see if the person EXITing is the same as that of the person who REMOVEd the object earlier. This is to ensure that we do not signal a THEFT event when one person picks up an object and a different person exits the area. In a closed environment, the IDs used may merely be track-IDs, in which each object that enters the monitored area is assigned a unique track-ID, and the track-ID is discarded when the object is no longer being tracked. If both the event type and the object ID match, the activation's index is incremented to 2. Since there are only 2 sub-events in the complex event in this example, the entire complex-event has been recognized, and an alarm is generated if desired. Also, since the THEFT event has been recognized, this newly recognized THEFT event may be a sub-event of another complex event. When the complex THEFT event is recognized, the current activations are recursively checked to see if the theft is a part of another higher-level event, such as a CRIME-SPREE.

2.4.2. Variations and enhancements

We have described the basic mechanism of defining and recognizing complex events. There are several variations on this basic mechanism. One is to allow unordered events, i.e., complex events which are simply the conjunction or disjunction of their sub-events. Another is to allow negated sub-events, which can be used to cancel an activation when the negated sub-event occurs. For example, considering the definition for THEFT again, if the person pays for the item, it is not a theft. Also, if the person puts the item back down before leaving, no theft has occurred. A more complete definition of theft is one in which "a person picks up an item and then leaves without putting it back or paying." Assuming we can recognize the simple events REMOVE, DEPOSIT, PAY, and EXIT, the complex THEFT event can now be expressed as the ordered list (REMOVE, ~DEPOSIT, ~PAY, EXIT), where "~" indicates negation. Another application of the complex event with negated sub-events is to detect suspicious behavior in front of a building. The normal behavior may be for a person to park the car, get out of it, and then come up into the building. If the person parks the vehicle and leaves the area without coming up into the building, this may be a car bombing scenario. If we can detect the sub-events for PARK, OUTCAR, ENTER-BUILDING, and EXIT, we can define the car-bombing scenario as (PARK, OUTCAR, ~ENTER-BUILDING, EXIT).

Another variation is to allow the user to label the objects involved in the events, which facilitates the ability to specify that two object be different. Consider a different car bombing scenario in which two cars pull up in front of the building, and a person gets out of one car and into the other, which drives away. The event definition must specify that there are two *different* cars involved: the car-bomb and the getaway-car. This can be accomplished by labelling the object involved when defining the event, and giving different labels to objects which must be different.

Finally, one could allow multiple activations for the same event. For example, the desired behavior may be that a separate THEFT event should be signalled for each item stolen by a given person, e.g., if a person goes into a store and steals three things, three THEFT events are recognized. The basic mechanism described above signals a single THEFT event no matter how many objects are stolen. We can achieve the alternate behavior by creating multiple activations for a given event type, differing only in the ID's of the objects involved.

2.4.3. Implementation in AVS

We have described a method for defining and recognizing complex events. Most of this has been implemented and incorporated into the AVS system. This subsection describes the current implementation.

AVS analyzes the incoming video stream to detect and recognize events such as ENTER, EXIT, DEPOSIT, and REMOVE. The primary technique used by AVS for event recognition is motion graph matching as described in [Courtney, 1997]. The AVS system recognizes and reports these events in real time as illustrated in Figure 10. When the person enters the monitored area, an ENTER event is recognized as shown in the image on the left. When the person picks up an object, a REMOVE event is recognized, as depicted in the center image below. When the person exits the area, the EXIT

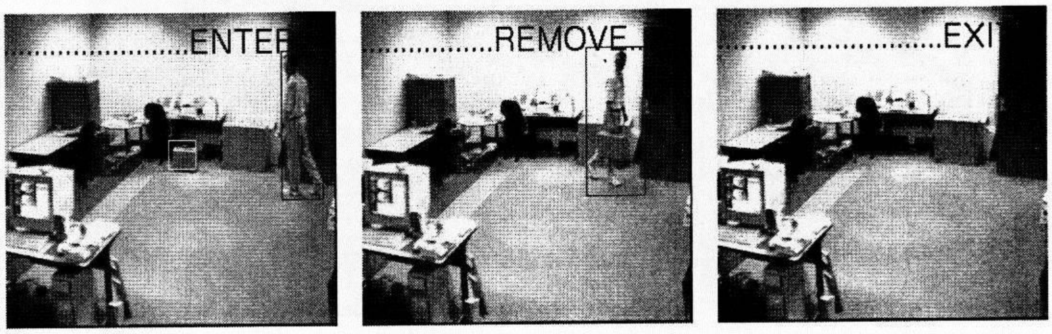


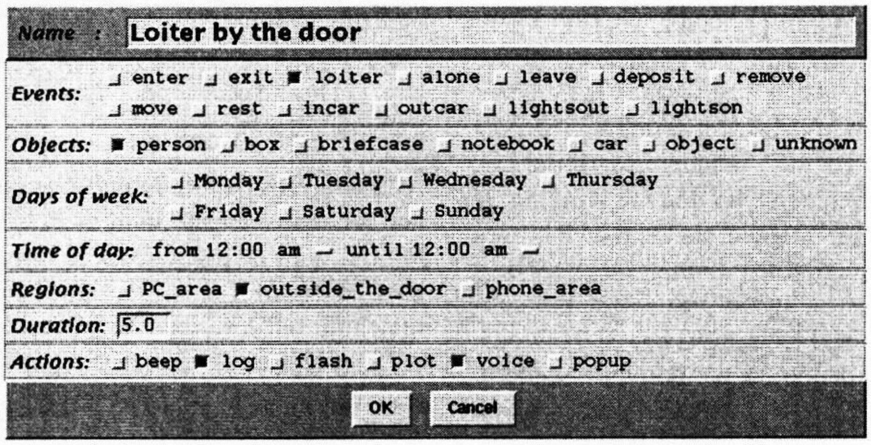
Figure 10: A series of simple events

event is signalled as shown in the image on the right

While the AVS system recognizes numerous events as shown above, the user can select which events are of interest by providing the dialog box interface illustrated in Figure 11. The user selects the event type, object type, time, location, and duration of the event of interest using a mouse. The user can also select an action for the AVS system to take when the event is recognized. This dialog box defines one type of simple event; an arbitrary number of different simple event types can be defined via multiple uses of the dialog box. The illustration in Figure 11 shows a dialog box defining an event called "Loiter by the door" which is triggered when a person loiters in the area near the door for more than 5 seconds.

AVS will generate a voice alarm and write a log entry when the specified event occurs. If the event is only being defined in order to be used as a sub-event in a complex event, the user might not check any action box, and no action will be taken when the event is recognized except to see if it matches the next sub-event in a complex-event activation, or generate a new activation if it matches the first sub-event in a complex event.

After one or more simple events have been defined, the user can define a complex event via the dialog box shown in Figure 12. This dialog box presents two lists: on the left is a scrolling list of all the event types that have been defined thus far, and on the right is a list of the sub-events of the complex event being defined. The sub-event list is initially blank when defining a new complex event. When the user double-clicks with the left mouse button on an item in the event list on the left, it is added as the next item in the sub-event list on the right. When the user double-clicks with the right mouse button on an item in the event list on the left, that item is also added to the sub-event list on the right, but as a negated sub-event. The event name is prefixed with a tilde (~) to indicate that the event is negated.


Name : Loiter by the door

Events: enter exit [✓ loiter] alone leave deposit remove move rest incar outcar lightsout lightson

Objects: [✓ person] box briefcase notebook car object unknown

Days of week: Monday Tuesday Wednesday Thursday Friday Saturday Sunday

Time of day: from 12:00 am until 12:00 am

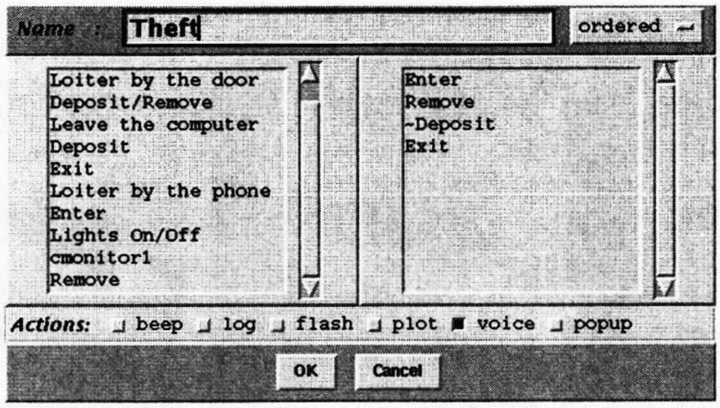
Regions: PC_area [✓ outside_the_door] phone_area

Duration: 5.0

Actions: beep [✓ log] flash plot [✓ voice] popup

OK Cancel

Figure 11: Selecting a type of simple event


Name : Theft | ordered

Loiter by the door
Deposit/Remove
Leave the computer
Deposit
Exit
Loiter by the phone
Enter
Lights On/Off
cmonitor1
Remove

Enter
Remove
~Deposit
Exit

Actions: beep log flash plot [✓ voice] popup

OK Cancel

Figure 12: Defining a complex event

In the upper right corner of the complex-event definition dialog box is an option menu via which the user indicates how the sub-events are to be combined. The default selection is "ordered" to indicate sequential processing of the sub-events. The other options are "all" and "any." If "all" is selected, the complex event will be signalled if all of the sub-events are matched, regardless of order, i.e., the complex event is simply the conjunction of the sub-events. If "any" is selected, the complex event occurs if any of the sub-events occurs, i.e., the complex event is the disjunction of the sub-events. At the bottom of the dialog box, the user can select the action to take when the complex event is recognized. The user can save the entire set of event definitions to a file so that they may be read back in at a later time.

Once a simple or complex event has been defined, the AVS system immediately begins recognition of the new events in real time, and taking the actions specified by the user. The AVS system, augmented as described, provides a functioning realization of the complex-event recognition method.

3. Advanced segmentation and tracking

In security applications, it is often necessary to track the movements of one or more people and objects in a scene monitored by a video camera. In real scenes, the objects move in unpredictable ways, may move close to one another, and may occlude each other. When a person moves, the shape of his or her image changes. These factors make it difficult to track the locations of individual objects throughout a scene containing multiple objects. The tracking capabilities of the original AVS system fail when there is mutual occlusion between the tracked objects. This section describes a new tracking method which overcomes this limitations of the previous tracking method, and maintains the integrity of the tracks of people even when they partially occlude one another.

The segmentation algorithm described here is related to tracking systems such as [Wren et al., 1997, Grimson et al., 1998, Cai et al., 1995] in that it extends the reference image to include a statistical model of the background. Our method further extends the tracking algorithm to reason explicitly about occlusion and maintain object tracks during mutual occlusion events. Unlike the capabilities described in previous sections, the new tracking method does not run in real time, and has not yet been integrated into the AVS system. Optimizations of the new method are expected to enable it to achieve real time operation in the future.

Figure 13 depicts an example scene containing two people. In (a), the two people are standing apart from each other, with Person-1 on the left, and Person-2 on the right. In (b), Person-1 moves to the right so that he is partially occluded by Person-2. Using a conventional technique such as background subtraction, it is difficult to maintain the separate tracks of the two people in the scene, since the images of the two people merge into a single large region.

Figure 14 shows a sequence of frames (in normal English reading order) in which it is particularly difficult to properly maintain the tracks of the two people in the scene. In this sequence, Person-2 moves from right to left and back again, crossing in front of Person-1. There are significant occlusions (e.g., in the third frame shown), and the orientations of both people with respect to the camera change significantly throughout the sequence,

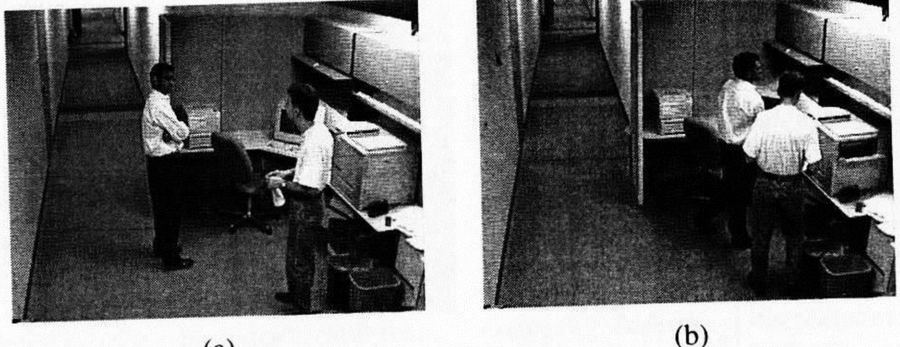

(a) (b)

Figure 13: An example scene containing two people with occlusion

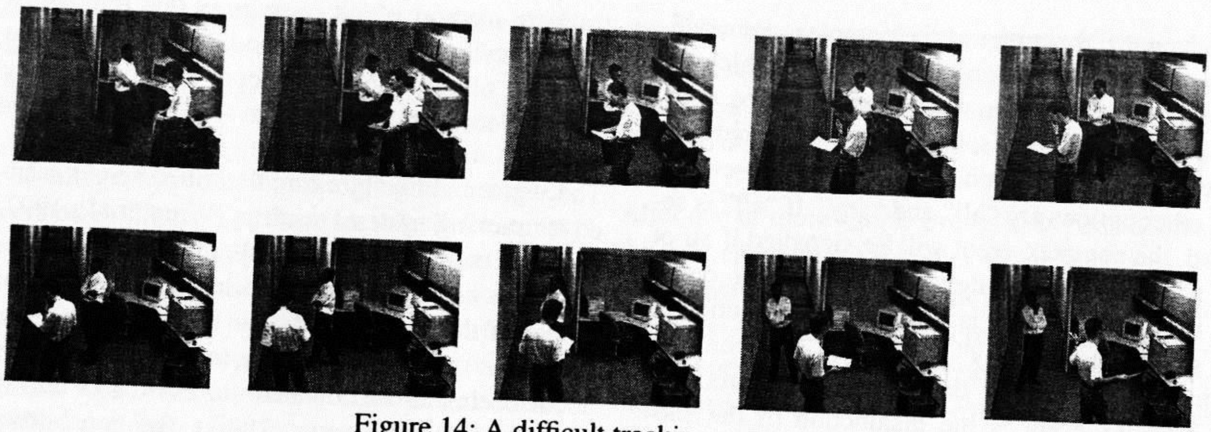


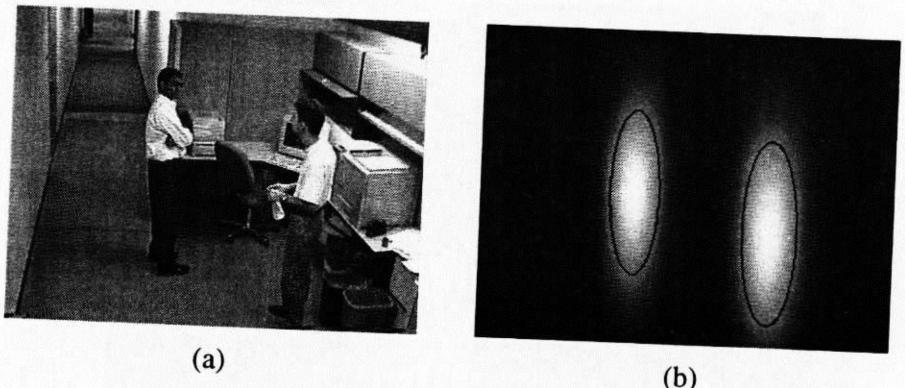
Figure 14: A difficult tracking sequence

making conventional template matching fail on this sequence.

A new tracking method is used to maintain tracks in sequences such as those depicted in Figures 13 and 14. The method maintains an estimate of the size and location of the objects being tracked, and creates an image which approximates the probability that the object intersects that pixel location. Figure 15b shows the probability images for the two person scene of Figure 13a, which is repeated here as 15a. The ellipse on the left indicates the estimated location of Person-1, and the ellipse on the right indicates the estimated location of Person-2. The brightness indicates the probability that the person's image intersects the given pixel, which is highest in the middle of the region, and falls off towards the edge. The black outlines represent the 50% probability contours. The size and shape of the regions are roughly the size and shape of a person standing at that location in the image.

We refer to the "person shaped" probability regions as *probabilistic templates* or simply *p-templates*. The path of the p-template through the scene represents the "track" of a given person which is maintained by the tracking system. P-templates can be used to reason about occlusion in a video sequence. While we only address the issue of p-templates for tracking people that are walking upright, the concept is applicable to tracking any object, e.g., vehicles and crawling people; although the shape of the p-template would need to be adapted to the type of object being tracked.

When the people in the scene overlap, the separate locations of the people can be maintained using the p-templates, and the region of partial occlusion can be detected. Figure 16 shows examples of such a situation. The two ellipses are maintained, even though the people are overlapping. The tracks of the people can be maintained through occlusions by tracking primarily on the basis of non-overlapping areas. This works for both the slight occlusion in Figures 16 (a) and (b), and often even for the very strong occlusions such as in Figures 16 (c) and (d). During the occlusions shown in Figure 14 and again in Figure 16 (c) and (d), the head of Person-1 is tracked, and the lower-body of Person-2 is tracked.



(a) (b)

Figure 15: Probability image for the locations of the people in the scene

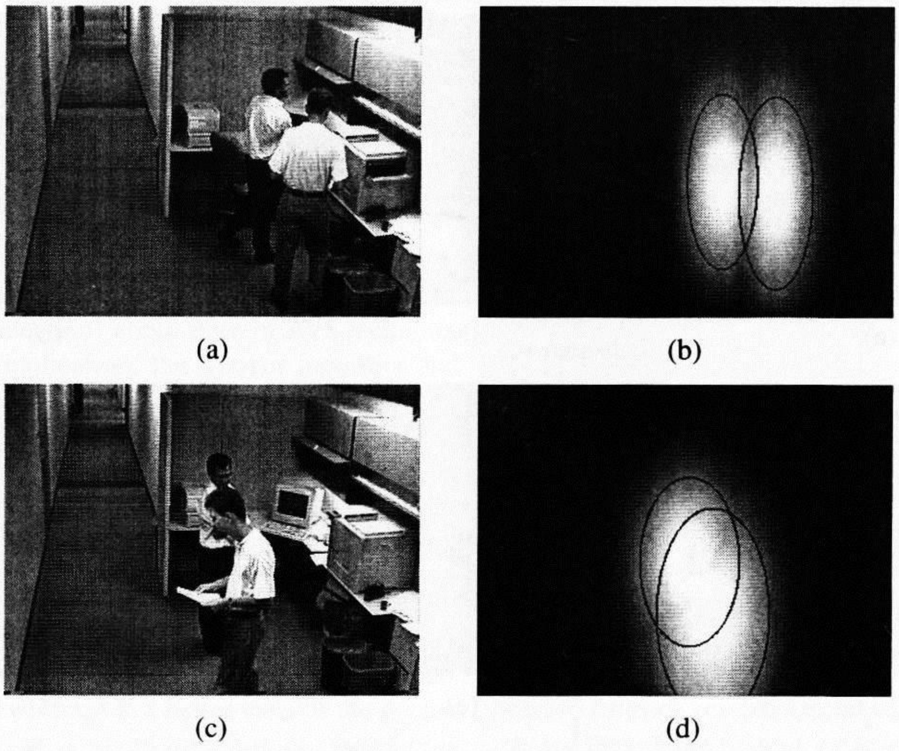


Figure 16: P-template images for partially occluding people

The new method requires a means of instantiating a new p-template when a person enters the scene, and updating the location of the region as the person moves through the scene. First we will describe the update mechanism, assuming that the p-templates have already been instantiated. The instantiation mechanism is described later.

The p-templates described above and depicted in Figures 15 and 16 represent the *prior* probabilities of the person locations, based on looking at the *previous* frame. These priors are then used to compute an estimate of the *posterior* probabilities of the person locations by looking at the new or *current* frame. The computation of the posterior probabilities takes into account both the prior probabilities and the information in the new frame. The posterior probabilities are used to update the locations of the people, and the new locations of the people are then used to compute the priors for the *next* frame.

Our current implementation computes the posteriors using a form of background differencing. Figure 17 shows the posteriors for the p-templates shown in Figure 16. Note that although there is significant overlap in the posterior estimates, especially in Figures 17 (e) and (f), there are significant differences in the brightnesses of the non-occluding areas. In Figure 17 (e), which represents the posteriors for Person-1, the head area of Person-1 is significantly brighter than in Figure 17 (f). Similarly, Figure 17 (f), which represents the posteriors for Person-2, is significantly brighter in the unoccluding area of Person-2's lower body.

Once the posteriors are computed, they are used to estimate the location of the tracked objects. In our implementation of a person tracker, we specifically need to estimate the location of the person's feet in the image, and their height in the image in pixels. Once the location and height are estimated, we can use the image-to-world coordinate transformation technique used in the original AVS system and described in [Olson and Brill, 1997]. That technique, called *quad-mapping*, computes the map locations of objects given the image locations of the bottom of the objects, e.g., in the case of a person, the location of the feet. Furthermore, if the scale of the map is known, the quad-mapping technique will estimate the size of the object, i.e., the height of a person being tracked.

If the lower portion of the p-template is unoccluded, foot locations are estimated directly from the image by looking at the bottom portion of the brightened region. If the upper portion is also unoccluded, the height can similarly obtained directly

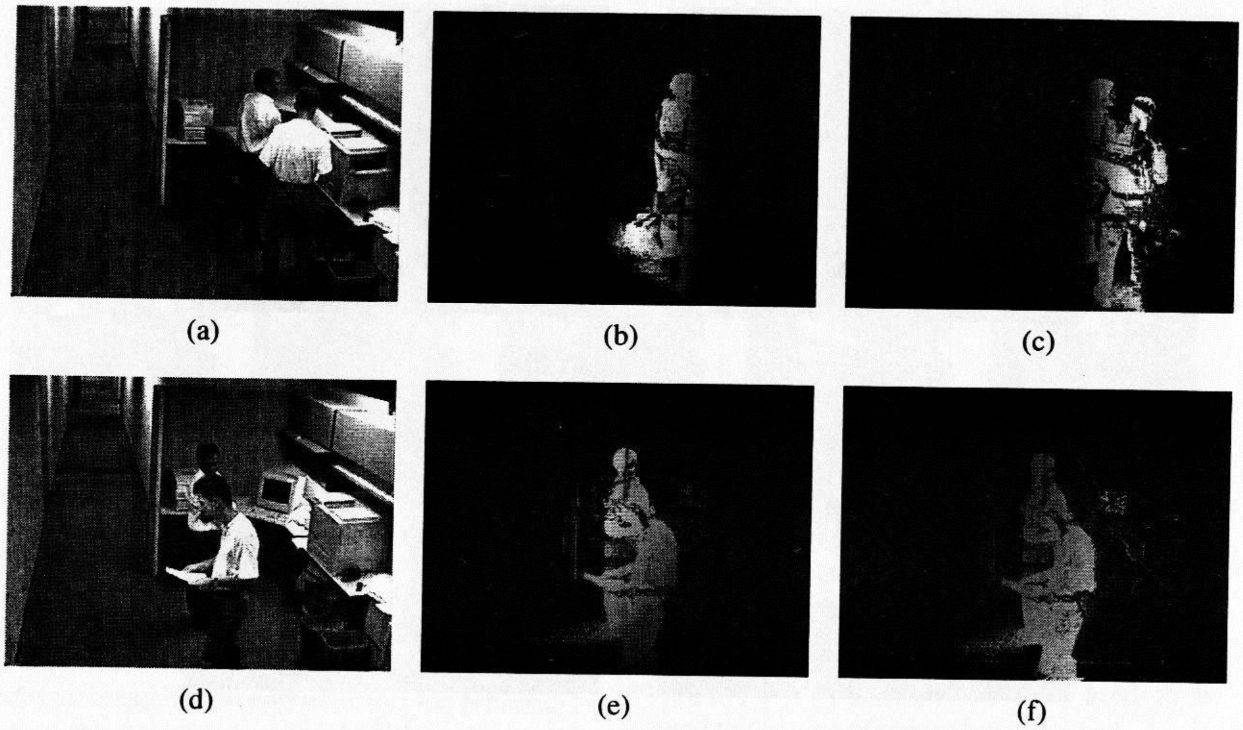


Figure 17: Posterior probability images for partially occluding people

from the image. If the upper part is occluded, but the lower part is not, the foot location is still determined directly from the image, but height is estimated using an estimate of the three-dimensional height of the person. The image height is then obtained by projecting the 3D height back into the image using the quad-mapping technique. If the lower portion is occluded, but the upper part is not, then the upper location is determined directly from the image, and then the 3D height is back-projected into the image to determine the foot location. If both the top and bottom are occluded, the location and height estimates are left unchanged from the previous frame.

Once the foot location and height of the person are computed, it is straightforward to compute the new location of the p-template, which is the Gaussian oval whose location and dimensions are determined by the foot location and image height computed above. The new p-template is then used to find the location of the person in the next frame, and the process repeats while the person remains in the scene.

A new p-template is instantiated whenever a new person enters the scene. Instantiation is best described in a Bayesian probabilistic framework. The p-templates constitute models of the objects in the environment. All of the pixels in the image are the result of a projection of some object in the environment—either from the background, or one of the people in the scene, or something else. The sum of the probabilities that the pixel is either from the background, from a person, or from "something else" must be 1.0. We maintain an "unknown" model to account for the probability that pixels may arise as a result of "something else." We compute the probability that each of the models caused the observed pixel value (where the unknown model is equally likely to produce any pixel value), and then use Bayes' formula to compute the inverse, i.e., the probability that the observed pixel value came from each of the models. When this computation is performed, for some of the pixels, the probability that the pixel came from the unknown model is the highest of all of the model probabilities. This results in a probability image for the unknown model, which represents pixels which probably came from something other than the objects the system knows about. At each frame, the probability image for the unknown model is computed, and this image is examined to see if adding a new person model would account for these unknown pixels. If so, a new person p-template is instantiated at the appropriate location, and the posteriors are recomputed.

Use of the procedure described above to track multiple people maintains tracks through occlusions where our previous technique could not. The robustness to occlusion of the new method enables video monitoring applications to improve tracking reliability in natural environments.

4. Best-view selection performance

Olson and Brill [1997] previously described the "best view selection" application of AVS technology. In this application, the system monitors and records the movements of humans in its field of view. For every person that it sees, it creates a log file that summarizes important information about the person, including a snapshot taken when the person was close to the camera and (if possible) facing it.

As the person is tracked through the scene, the tracker examines each image it captures of that person. If the new image is a better view of the person than the previously saved snapshot, the snapshot is replaced with the new view. In this manner, the system always contains the "best" view seen of the person thus far. When the person leaves the scene, the log entry is saved to a file. Each log entry records the time when the person entered the scene and a list of coordinate pairs showing their position in each video frame. The log entry also contains the "best" snapshot of the person while they were in the scene. Finally, the log entry file contains a pointer to the reference image that was in effect when the snapshot was taken. This information forms an extremely concise description of the person's movements and appearance while they were in the scene. An example of such a record in shown in Figure 18.

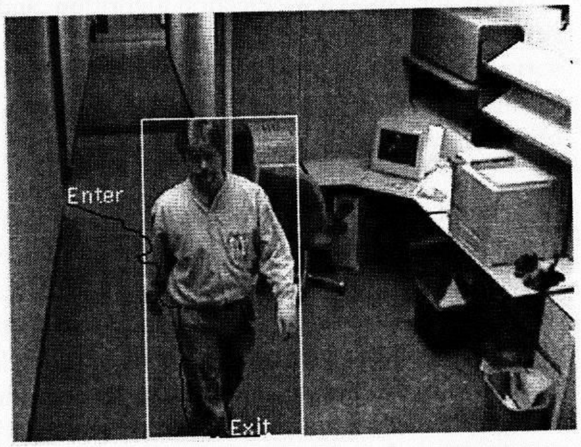


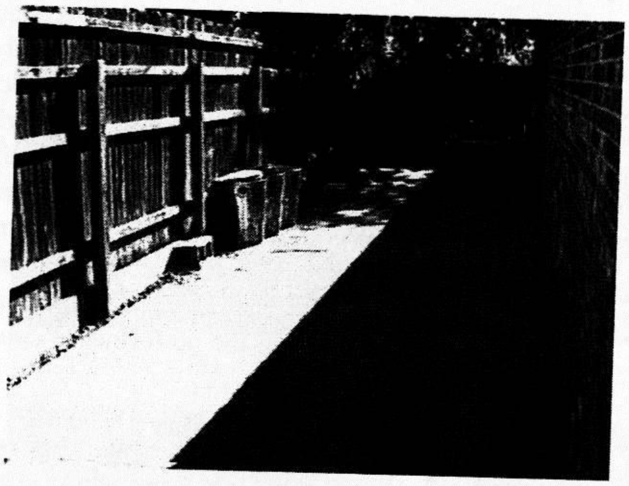
Figure 18: Example best view selection record

In an initial evaluation of this system, the system was installed in an uncontrolled office hallway and run for 118 hours. In this time, the system recorded 965 log entries in 35MB (uncompressed). The resulting records were examined to estimate the system performance, and we estimated 96% detection rate at 6% false alarm rate, with most errors due to segmentation and correspondence failure. However, for this initial experiment, there was no ground truth against which the performance could be measured.

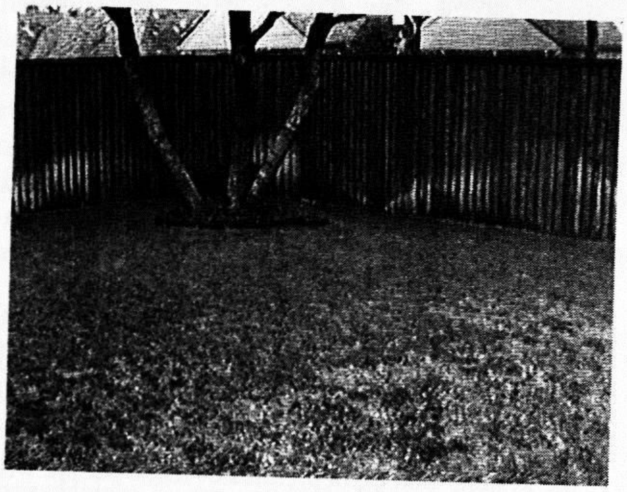
Recently, we have evaluated the system against ground truth observations. The performance of the system was initially evaluated on four hours of indoor video data. The video was manually annotated to obtain ground truth, and the surveillance system was evaluated against this ground truth. For situations in which only one person was in the scene, the system recorded exactly one record for each person, i.e., no person passed undetected though the field of view, and there is exactly one record for each such person. In the indoor condition, we observed a 100% detection rate.

For situations involving more than one person, the system occasionally failed to maintain track through partial occlusions. The result of this is that the system took extra pictures of these people when their track was re-acquired after the occlusion. On other occasions, the system failed to recognize that a motion region contained two people, and so it only took one picture that contained both people. We expect to reduce these errors via the use of the new tracking algorithms described above, once these algorithms are running in real time and are incorporated into the AVS system.

In order to evaluate and improve the system performance in outdoor monitoring environments, we have adopted an iterative research methodology in which we record representative videotape (2-3 hours), ground truth it with respect to the 'person events' that occur in the scene. One 'person event' is defined to be a video sequence in which one person enters monitored area completely, walking upright, and then exits field of view completely. We then run AVS system on the videotape and measure the false positives and negatives on person events. We then improve system as necessary to eliminate errors on video sequence and repeat the process.



(a)



(b)

Figure 19: Outdoor environments

Outdoor environments can be particularly difficult for video monitoring systems that operate based on change detection, due to the outdoor lighting variation. Figure 19 depicts two outdoor environments used to evaluate AVS best-view-selection performance. In Figure 19 (a), there is a strong shadow line running down the center of the field of view, which moves as the sun angle changes. The shadow motion here is sufficient to cause problems for a fixed background subtraction system within 5 minutes. There are also a number of trees in the background which move when the wind blows. Moreover, the shadows of these trees fall directly into the rear of the monitored area, and these shadows move with the wind as well. The shadow of the tree in Figure 19 (b) has a similar behavior. Cloud movement also causes large changes in brightness throughout the images.

Our initial outdoor evaluation was conducted in the environment depicted in Figure 19 (a). We captured two hours of outdoor video with extremely difficult imaging conditions caused by wind blown vegetation and strong shadows, which produced a large amount of "noise" motion. Additionally, the gate at the rear of the scene often blew open and closed. We manually ground-truthed the video to determine that a person entered the scene 20 times during the two hour sequence. The system recorded 16 of these events, for a detection rate of 75%. The undetected people were "lost in the noise." The system also produced 16 false detections in the two hour period, caused by noise from the moving shadows.

We were able to improve on this performance using our iterative research methodology to achieve a 100% detection rate for the 20 events in this two hour sequence. The system still recorded 8 false positives on this sequence. Four of these were caused by the gate blowing open and closed. The other four were cases in which the system lost track of the person in the field of view, and therefore took two pictures of the person, one before losing track, and another after picking up the track again. These cases are therefore more properly referred to as "extra pictures" rather than false positives.

Having achieved improved performance in the environment depicted in Figure 19 (a), we proceeded to test the system in the environment of Figure 19 (b). One three separate days we captured 1-2 hours of video, for a total of 4 hours of test video data in the environment of Figure 19 (b). We ground-truthed this video to determine that it contained 115 person events. The AVS system processed this video using the best-view-selection algorithm, and the results were compared to ground truth. We observed a 100% detection rate and a 2.6% false positive rate as a result of three false positives, all of which were "extra pictures."

In general, system performance was excellent in the indoor condition, with the exception of scenes containing multiple people, which produced extra records. We expect to address the multi-person problem using the p-template technique described in section 3. No person entered the scene without being recorded, even when there were multiple people. The system performance degrades in difficult outdoor lighting conditions, but it has improved significantly in recent work.

5. Conclusion

We have described several improvements in the video monitoring capabilities of the AVS system. Some improvements, such as vehicle event recognition, increase the functionality of the system to enable it to recognize new classes of events. Other improvements, such as the advanced segmentation and tracking, increase the robustness of the system's ability to recognize events in the presence of complications such as occlusion. We will continue to make improvements in the two categories of increased functionality and increased robustness. For the functionality improvements, we expect to recognize new classes of events, especially events regarding vehicles. For the robustness improvements, we are pursuing techniques that enable the system to be robust to lighting variation. As the techniques become more complex, additional effort will be needed to optimize the algorithms for real time operation. Our advanced segmentation and tracking will be the subject of optimization efforts in the near future.

References

[Cai, et al., 1995] Q. Cai, A. Mitiche, and J. K. Aggarwal. Tracking human motion in an indoor environment. In *Proc. of International Conference on Image Processing* (ICIP-95), 1995.

[Davis and Bobick, 1997] J. Davis and A. Bobick. Representation and recognition of human movement using temporal templates. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR-97)*, pp. 928-934, 1997.

[Courtney, 1997] J. D. Courtney. Automatic video indexing via object motion analysis. *Pattern Recognition*, **30**(4), April 1997

[Grimson et al., 1998] W. E. L. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR-98)*, 1998.

[Olson and Brill, 1997] T. J. Olson, and F. Z. Brill. Moving object detection and event recognition algorithms for smart cameras. *Proceedings of the 1997 Image Understanding Workshop*, New Orleans, LA, May 11-14, 1997, p. 159-175.

[Tserng, 1998] C. H. Tserng. Event recognition in scenes involving people and cars. Master's Thesis, MIT, May 1998.

[Wren et al., 1997] A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1997, **19**(7), pp. 780-785.