Event Detection and Analysis from Video Streams

Gérard Medioni Ram Nevatia Isaac Cohen
University of Southern California
Institute for Robotics and Intelligent Systems
Los Angeles CA 90089-0273
{medioni|nevatia|icohen}@iris.usc.edu
http://iris.usc.edu/Outlines/vsam-project.html *

Abstract

We present a system for event detection and analysis from video streams. Our approach is based on a *detection* and *tracking* module which extracts moving objects trajectories from a video stream. These trajectories, together with a rough description of the scene, are then used by the *behavior inference* module in order to recognize and classify object motion. The hierarchical tasks are performed on a buffered set of frames in order to provide accurate results by taking into account the temporal coherence of moving objects.

1 Introduction

Video streams collected by stationary platforms, mobile ground vehicles, or Unmanned Air Vehicles (UAVs), present several challenges for continuous surveillance and monitoring in battlefield and urban environments. While the multiplicity of these sensors permits close surveillance and monitoring, one cannot rely on purely manual, human resources for image analyses. Indeed, unaided humans may have difficulty remaining focused on the tasks. Typically, long periods may pass before any event of interest takes place; it is easy for human attention to wander in such a situation, and significant events may be missed. We present steps toward a completely automated process to indicate possibly significant events to an operator.

The key task of video surveillance and monitoring (VSAM) is to observe moving vehicles and humans, and to infer whether their actions pose a threat that should be signaled to the human monitor.

This event detection and analysis is based on the detection of moving objects, the estimation of their speed and trajectory and inference of their behavior (within the constraints of available resolution and time).

Motion detection is made difficult as both the observer and some elements of the scene may be moving. To handle this situation, we first compensate in the image for the motion field induced by the observer, which manifests itself globally. The results of this egomotion estimation are used to register frames, to detect independently moving objects, and to track them.

Motion by itself, however, is not a sufficient indication of a threatening or otherwise interesting activity. In most natural scenes, there is a significant amount of moving objects, and it is the analysis of their trajectories and interaction with the features of the scene which allows us to classify and recognize interesting events. This scenario recognition, or behavior inference module, is based on a crude model of the site being observed. This image to model correspondence helps in relating the observed motion to relevant features on the ground. Certain speeds and trajectories in certain contexts indicate a threatening behavior. More complex behaviors are analyzed by observing actions of groups of vehicles, rather than just single vehicles.

In the following sections, we first give an overview of our approach, then describe in more detail the detection, tracking and behavior analysis modules. Besides demonstrating the system on real data streams, we characterize the performance of each module.

*This research is supported by the Defense Advanced Research Projects Agency (DARPA) under contract DAAB007-97-C-J023, monitored by US Army, Fort Monmouth, NJ.

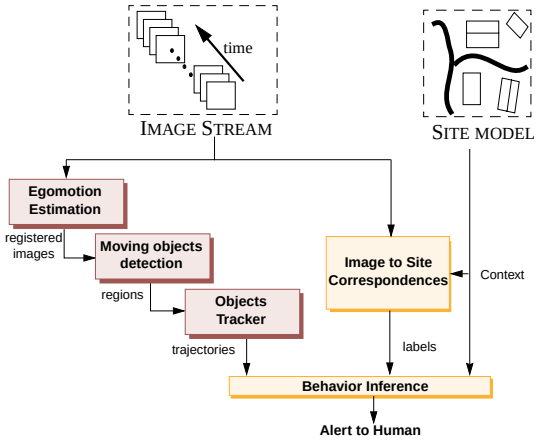


Figure 1: *Overview of the system*

2 Overview

Our approach is based on two modules. The first one achieves detection and tracking of moving objects from the video stream collected by the UAV, while the second takes the inferred trajectory of each object, and user provided contextual information, to recognize the behavior of the moving objects among a large number of potential scenarios. The most important features of each module are briefly described in the following.

Tracking the detected objects over the image sequence amounts to matching these different regions in order to determine the trajectories of the objects. This matching can be done using objects templates, color or texture. For generality purposes, we propose instead to infer, from the detected regions, a 2D dynamic template of the object [Cohen and Medioni, 1998]. A dynamic template is extracted by using the *temporal coherence* of the object over a number of frames (typically 3 to 5 frames). Temporal integration of the detected objects over a number of frames is used to characterize the moving objects by computing their motion, direction and trajectory. A graph is used to represent moving regions and the way they relate to each other. Each node is a region and each edge represents a possible match between two regions in two different frames. We assign to each edge a cost, which is the likelihood that the regions indeed correspond to the same object. This formalism can handle a large number of situations such as stop and go motion, and allows us to characterize the trajectories of moving objects as an optimal path along each graph's connected component.

The behavior inference module is based on the definition of a set of scenarios. These scenarios are defined by a combination of spatial and temporal properties. The behavior module uses a set of temporal properties such as distance of the moving object (the car) to a set of reference locations (road block, buildings, ...). These reference locations describe the most significant objects in the scene. Currently, these objects are defined manually, and consist of polygonal outline of the objects in the scene. Each moving region in the scene, based on its trajectory, generates a set of hypotheses that are used to compute a set of properties. These scalar values are used for identifying the behavior of the object as one of the available scenarios [Brémond and Medioni, 1998]. Moreover, the system relies on a recursive definition of scenarios, which has the advantage of being generic, since any recognized scenario can be one of the available scenarios or any combination of these. Recognition is then achieved by computing a likelihood degree, and each transition from a subscenario to the next is computed through a finite state automaton.

The detection/tracking and behavior inference modules are implemented to process the frames are as they are acquired by the platform. Indeed, the detection of moving regions is done based on the current and the previous frame. However, to achieve robustness, the tracking and the behavior inference are done on a buffered set of frames. This buffer is typically made of the last 5 to 10 frames, and is used to build the graph representation of the moving objects and infer accurately a template of the moving object and its trajectory. This data is then used by the behavior inference module for scenario-recognition. The advantage of such an approach is that the two modules (tracking and behavior inference) and the acquisition proceed in a concurrent way. We are thus able to characterize the behavior of the moving objects while the action is taking place (with a fixed delay of a few frames).

3 Detection and Tracking of Moving Objects

The first module detects and tracks independently moving objects. This is accomplished in three steps:

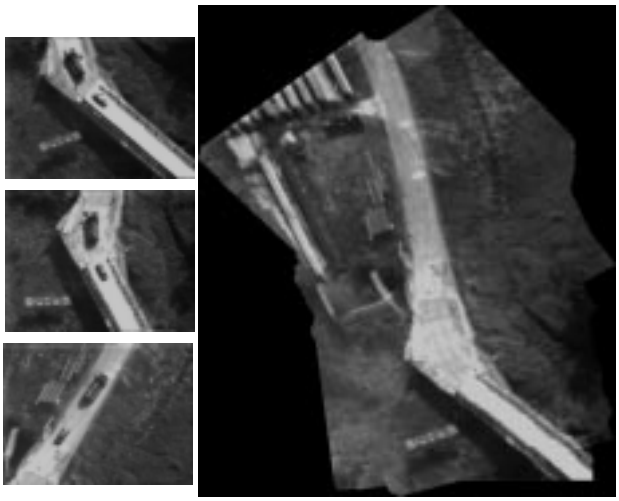


Figure 2: *Mosaic obtained by the hierarchical feature based approach*

- compensation of the image flow induced by the motion of observation platform (also called image *stabilization*)
- *detection* of moving regions in each frame,
- *tracking* of moving regions in time.

In the following section, we briefly describe the method used for detecting and tracking moving objects; for more details, we refer the reader to [Cohen and Medioni, 1998].

3.1 Egomotion Estimation

The ego-motion estimation is based on the camera model which relates 3D points to their projection in the image plane. The framework we use models the image induced flow instead of the 3D parameters of the general perspective transform [Irani *et al.*, 1995; Morimoto and Chellappa, 1997; Szeliski and Shum, 1997]. The parameters are estimated by tracking a small set of feature points in the sequence. Furthermore, a spatial hierarchy in the form of a pyramid is used to track selected feature points. The pyramid consists of at least three levels, and an iterative affine parameter estimation produces accurate results. Figure 2 illustrates a byproduct of the processing, a mosaic obtained from a video stream.

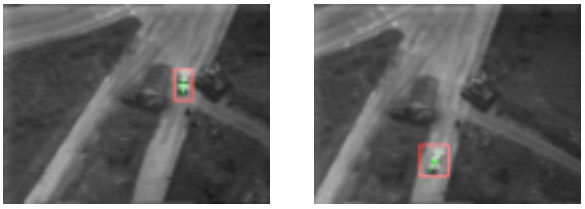


Figure 3: *Tracking a car going through a road block. The rectangle corresponds to the bounding box of the moving object.*

3.2 Detection of Moving Objects

Stabilization allows us to create a synthetic image sequence in which we have cancelled the motion field induced by the displacement of the observer. This image sequence is more suitable for moving objects detection, since classical methods like temporal gradients [Halevi and Weinshall, 1997], accumulated gradients [Davis and Bobick, 1997] and optical flow [Irani *et al.*, 1992; Cohen and Herlin, 1996] can be used to detect image variations.

These detected variations are due to 3D structures, not properly handled by the affine model, or to the moving objects in the scene. These regions are usually small and cannot be used to infer the 3D geometry of the moving object. However, we can use the redundancy of the information and its temporal coherence in order to locate and track moving objects in the scene. Our method is based on normal flow computation and a 2D multiframe motion analysis for locating and tracking moving objects.

3.3 Tracking Moving Regions

The detected moving regions represent the essential information used by a the video surveillance system. In order to characterize the events or actions taking place in the scene monitored by the video camera, additional information needs to be inferred from objects trajectory and their relationship to the monitored scene.

Inferring objects trajectory requires to match regions detected in two (or more) consecutive frames. In airborne video imagery, the matching has to deal with false detections due to errors in image stabilization, and with objects in the scene which stop and resume moving, or may become partially occluded. Therefore matching the different detected regions in order to de-

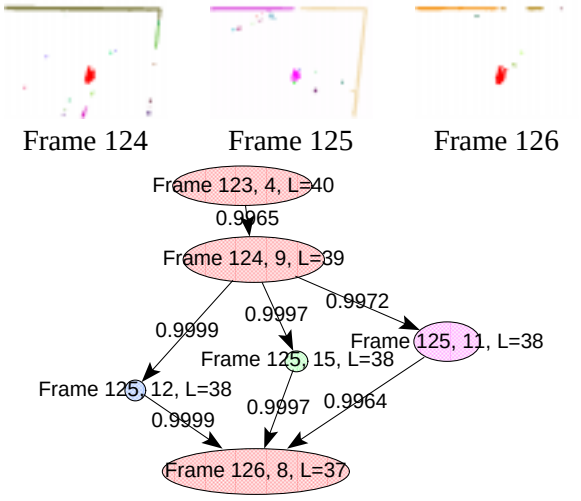


Figure 4: *Detected regions and associated graph*

rive a path describing their trajectory requires a good representation of the detected regions and a similarity measure which can be used for properly matching these regions. Moreover, a reliable approach for tracking moving object must manage false alerts, while still characterizing the moving objects.

3.4 Extraction of Object Trajectories

Using a graph representation of moving objects, and an appropriate weighting of the nodes and vertices of the graph allows us to characterize objects trajectories as an optimal path along each graph's connected component [Cohen and Medioni, 1998]. Figure 4 depicts the graph representation used in our approach along with some of the properties associated to each node and edge. This approach allows us to automatically extract the trajectories of all moving objects as new frames are acquired. Each newly detected region is matched to the previous ones and is considered as a potential goal node. On the other hand, each graph node without a parent is considered a potential source node. The source and the goal nodes define the origin and the end of each extracted path.

Defining a criterion to characterize an optimal path is equivalent to associating to each edge of the graph a cost. Each edge of the graph corresponds to a match between two moving regions. We also have to take into account the location of each node, since nodes describing the same object are likely to be close one to another.

Figure 5: *Trajectory of the car and the locations where it stopped*

The extraction of the optimal path is done by starting at graph's nodes without parent, and expanding the node with maximal cost. A result of this approach is illustrated in Figure 5, where a trajectory of the car is shown.

3.5 Experimental Results and Evaluation

This method was tested on a set of video streams acquired by Predator UAV and VSAM airborne platforms. These video streams represent a variety of scenes involving human activity (see figures 5 and 6.), and were used to evaluate the performance of our detection and tracking algorithm. Although it is difficult to quantitatively evaluate this type of system, we summarize in table 1 some values having a clear decisive relevance to the video surveillance.

Table 1 shows a summary of the output of the detection and tracking module. The numerical values represent the outputs obtained at different stages of the processing. The "Moving Objects" column represents the true number of objects moving in the video stream, and was provided by the user. The next two columns represent the output of the detection and tracking submodules respectively. As we can see, the number of regions detected is fairly large compared to the number of moving objects. These numbers correspond to the number of regions where the normal flow field was larger than a given threshold (10^{-5}, in all the experiments). The detection column gives the distribution's plot of the number of these regions over the processed

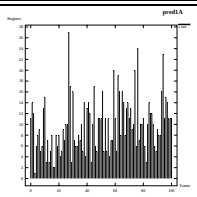
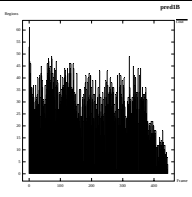
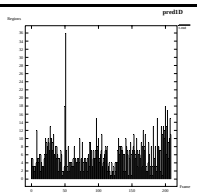
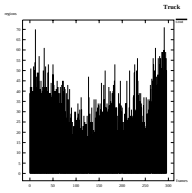
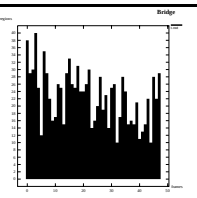
video stream	Moving Objects	Detection	Tracking		Metrics	
		detected regions	Regions	Trajectories	DR	FAR
	1	 $\bar{x} = 9, \sigma = 5$	1	1	1.	0.
	2	 $\bar{x} = 29, \sigma = 15$	3	3	1.	0.2
	4	 $\bar{x} = 6, \sigma = 3$	4	5	1.	0.
	2	 $\bar{x} = 34, \sigma = 11$	10	5	1.	0.8
	7	 $\bar{x} = 22, \sigma = 8$	15	12	1.	0.53

Table 1: *Quantitative analysis of the detection/tracking modules*

sequence. Also, the associated mean and variance are given as indicative values. The temporal integration of these regions, over a set of frames, allows us to reduce this number of regions (given in the fourth column) and discard the *false detections*, since regions due to noise are not temporally coherent. However, some inaccuracies of the egomotion model, or the presence of a parallax can cause some regions to have a coherent temporal signature. Finally, the column "trajectories", represents the number of trajectories considered as valid (*i.e.* coherent temporal regions detected for more than 10 frames), which represents the latency time used in the tracking. In some situations, this number is larger than the number of moving objects in the stream. This is due to object trajectories being fragmented into several paths, and to failures in matching similar regions representing the same object. The remaining trajectories are due to regions with good temporal coherence which do not correspond to moving objects, and these are, typically, regions due to strong parallax.

We have defined two metrics for characterizing the *Detection Rate* (DR) and the *False Alarm Rate* (FAR) of the system. These rates, used to quantify the output of our system, are based on:

- TP (true positive): detected regions that correspond to moving object,

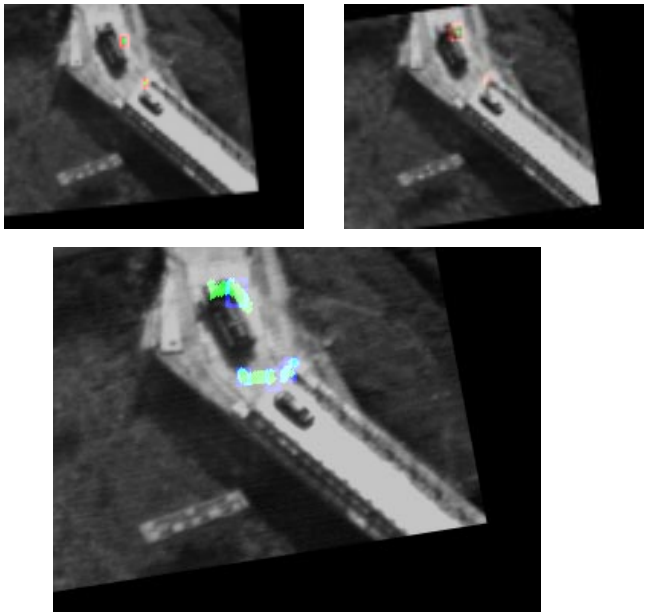


Figure 6: *Tracking small objects (people) on a bridge)*

- FP (false positive): detected regions that are not a moving object,
- FN (false negative): moving objects not detected.

and combined to define:

$$DR = \frac{TP}{TP + FN}$$

and

$$FAR = \frac{FP}{TP + FP}$$

These detection rates are reported in table 1. As the number of moving objects is small, these measurements may have large variances. This table shows that the large number of moving objects generated by the detection submodule is reduced by the tracking submodule, leading to a perfect detection rate in all examples. The large FAR in the last two experiments is due to 3D structures. We are in the process of adding a filtering step to differentiate motion from parallax (and infer the 3D at the same time). This should drastically reduce the FAR.

More details and a complete discussion can be found in [Cohen and Medioni, 1998].

4 Behavior Inference

Given object trajectories and associated contextual information, the task of the interpretation system is to recognize scenarios relative to the behaviors of mobile objects. In our case, the mobile objects correspond either to humans or to vehicles, and the scenarios describe human activities.

We have developed a *behavior analysis* module which identifies the tracked moving regions as mobile objects and interprets the scenarios that are relative to their behaviors. This interpretation step is done on a delayed buffered set of frames, typically ten frames, in order to take as input only moving regions tracked with enough reliability. The scenario recognition module uses two kinds of context (defined as *a priori* information on the scene environment and acquired before the processing of the scene): spatial context; and mission context. The spatial context corresponds to a map of the scene. The mission context contains the specific methods to recognize the type of scenarios.

We call *scenario* any activity related to humans. For the mobile objects involved in a scenario, we have defined two roles: *source* or *reference*. The source object is a mobile object that performs the action associated with the scenario. The reference object is the reference of the action. The reference object can be either a mobile object or a static object belonging to the scene context. For example, in the activity "*the car* goes toward *the checkpoint*", "*the car*" is the source object and "*the checkpoint*" is the reference object.

We describe briefly the behavior analysis module in the following section. A more complete description of this work can be found in [Brémond and Medioni, 1998].

4.1 Scenarios Describing Human Activities

4.1.1 Mobile Object Properties

The image processing module detects the moving regions and computes several **attributes** from them. Currently we use eight attributes: height, width, speed, motion direction and the distance to a reference object. The tracking module then tracks the detected regions. Because of detection errors, as shown on figure 7, a moving region can either correspond to noise, to a part of a mobile object (e.g. the arm of a person), to one

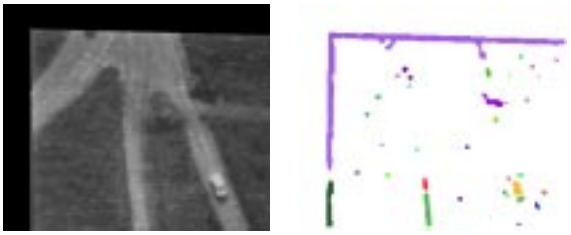


Figure 7: *Moving regions detected by the first module. Most detected regions correspond to noise*

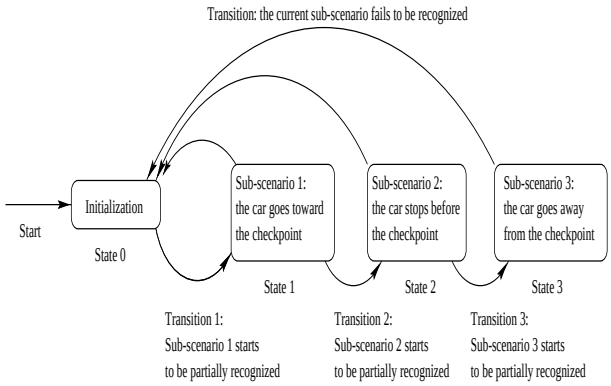


Figure 8: *Automaton for the scenario "the car avoids the checkpoint"*

mobile object (e.g. a person) or to a group of mobile objects (e.g. a crowd). The scenario recognition module then generates hypotheses to consider the tracked moving regions as mobile objects composed of one or several regions. Finally, we compute the properties of mobile objects and analyze the scenarios relative to the behavior of mobile objects based on their properties. The mobile object properties have a numerical value and they are supposed to be instantaneous. They are computed on a short interval of time (typically 4 or 5 frames for video surveillance application) in order to avoid unstable values. The mobile object properties are intended to be generic (i.e. independent of applications) and re-usable for different application types. Thus we use these properties as basic elements to recognize scenarios.

4.1.2 Scenario Modeling

We recursively recognize a **scenario** based on the properties of the mobile objects involved in the scenario. At level 0, a scenario is recognized directly from the associated mobile object properties. At level *n*, a scenario is recognized through a combination of sub-scenarios recognized at level *n-1*. There are two types of combinations: *single state*; and *multi-state*. First, a scenario can correspond to a single state constraint on a set of sub-scenarios and on mobile object properties. Second, a scenario can correspond to a multi-state constraint representing a temporal sequence of sub-scenarios. For example, the scenario *"the car goes toward the checkpoint"* represents a single state combination of three properties: the distance between the car and the checkpoint, the direction of the car and its speed. If the distance decreases, the direction is toward the checkpoint and the car slows down, then the scenario is said to be recognized. On the other hand, the scenario *"the car avoids the checkpoint"* represents a multi-state constraint of three sub-scenarios: *"the car goes toward the checkpoint"*, *"the car stops before the checkpoint"* and *"the car goes away from the checkpoint"*. The scenario is recognized when its three sub-scenarios are consecutively recognized.

In the case of a multi-state combination, the recursive definition of scenarios allows us to easily change the scenario duration and the temporal segmentation. For example, a scenario at an upper level can always be defined to add new sub-scenarios. For the scenario *"to park a car"*, we can define at an upper level a scenario to describe several attempts to park the car before succeeding.

To describe a scenario, we use an eight part model:

- scenario name,
- involved mobile objects,
- combination type (*single state* or *multi-state*),
- list of sub-scenarios,
- scenario recognition value,
- set of methods to compute the recognition value,
- scenario likelihood degree and
- set of methods to compute likelihood degree.

Thus scenarios have a symbolic value, and we use them to recognize activities on long image sequences. They are intended to describe various human activities, and adapt the recognition to different types of applications.

4.2 Scenario Recognition Methods

We use two main methods to recognize scenarios, depending on the type of the combination of sub-scenarios.

4.2.1 Single State Combination of Events

If the case of a scenario represented through a single state constraint, a value is associated to quantify the verification of the constraint. In this case, the main point of the scenario recognition is to combine the values of the associate sub-scenarios and properties in order to compute the scenario recognition value. However, the hard part of the method is to handle the uncertainty of these values, mainly due to the inaccuracy of mobile object properties, or to detection errors. To tackle this problem, we have defined a likelihood degree for scenarios. This is a numerical value indicating how reliable is the computation of the scenario recognition value.

We use two kinds of methods to compute the likelihood degree. Most of the time, and in particular for long scenarios, we use the temporal coherence: during the processing, if the new scenario value is coherent with the old ones, we increase the likelihood degree of the scenario. For example, when computing the scenario "the car slows down" if we notice that the car speed decreases effectively at each new frame arrival, we incrementally increase the scenario likelihood degree. Second, to handle specific cases we sometimes compute the likelihood degree using possibilistic logic [Brémond and Thonnat, 1996]. When a scenario is based on several mobile object attributes, and when we cannot wait for the arrival of new information, we diagnose thanks to possibilistic logic, whether the scenario value computation is reliable or not at a given time point.

4.2.2 Multi-State Combination of Events

If the scenario represents a temporal sequence of sub-scenarios, then it is recognized through an automaton, whose states correspond to the sub-scenarios. The scenario recognition value is the current state of recognition. The likelihood degree and the automaton transitions are computed through the likelihood degree of its sub-scenarios. The scenario is recognized when all its sub-scenarios are consecutively recognized, and if its likelihood degree is high enough. For example, we have built a scenario that recognizes whether or not a car is avoiding a checkpoint. The recognition automaton is described on figure 8. The scenario is composed of three sub-scenarios: *"the car goes toward the checkpoint"*, *"the car stops before the checkpoint"* and *"the car goes away from the checkpoint"*. The scenario is recognized when the three sub-scenarios are consecutively recognized and if the likelihood degree of the main scenario is high enough. Thus likelihood degrees are propagated from bottom to top through the whole scenario recognition module. When a recognized scenario is interesting for a given application, then an alarm is triggered.

We do not use spatial or temporal logic as it is the case in several interpretation systems, because these logics cannot be used in all surveillance applications. In an idealistic situation, no diagnosis stage is necessary, and properties computed on mobile objects are true or false at one location and at one time-point. For example, this idealistic situation occurs when using an optical barrier as sensor. The instant when the optical barrier detects an event is a symbolic information that can be manipulated in the framework of a temporal logic. However this is not the case when using a camera, and the diagnosis stage is then often necessary. For example, to recognize the scenario *"the car has stopped for more than 3 minutes"* we need at least a diagnosis stage to establish when the scenario starts. Thus the manipulation of this information is uncertain and less informative. Therefore, to represent the temporal aspect of scenarios, we just use the notion of a temporal sequence. The use of an automaton is then sufficient to recognize a scenario corresponding to a multi-state combination of sub-scenarios.

The single state constraint verification and the automaton are the two main methods that enable us to recognize all the scenarios described by the proposed scenario model.

4.3 Experimental Results

One of our target applications is video-surveillance of streams obtained by the Predator UAV. We choose as instance of scenario the monitoring of checkpoints (i.e. road blocks). As an example to validate the scenario recognition module, this section describes two

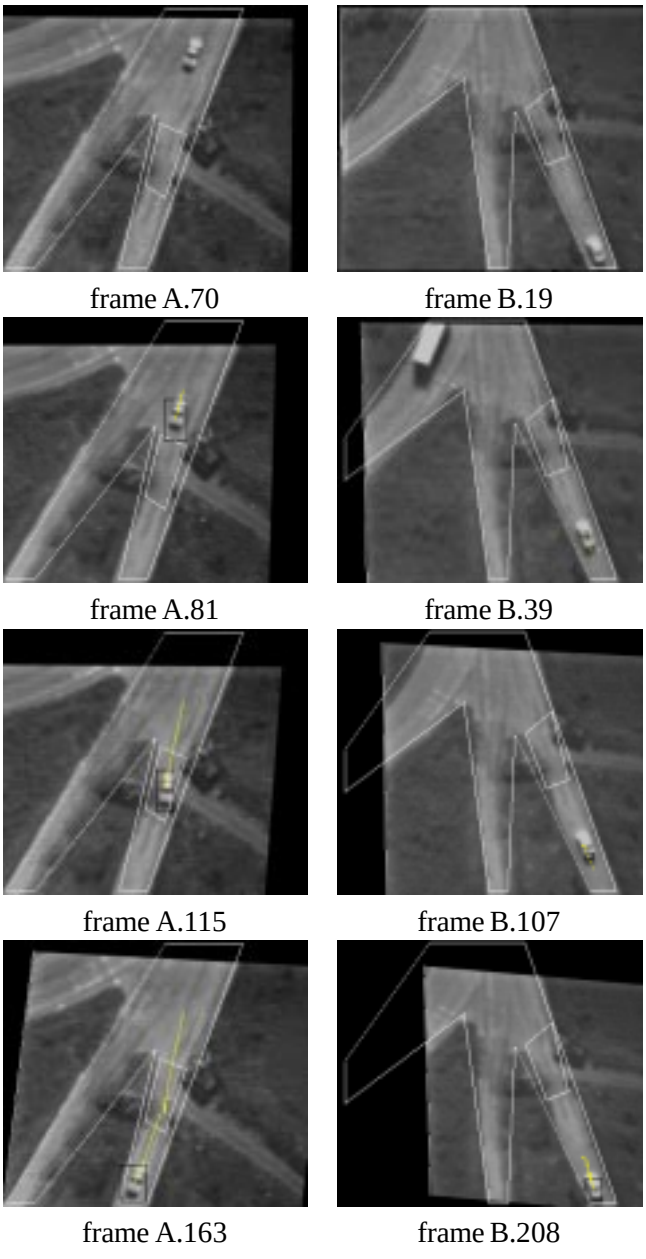


Figure 9: *Car behaviors related to the monitoring of a checkpoint*

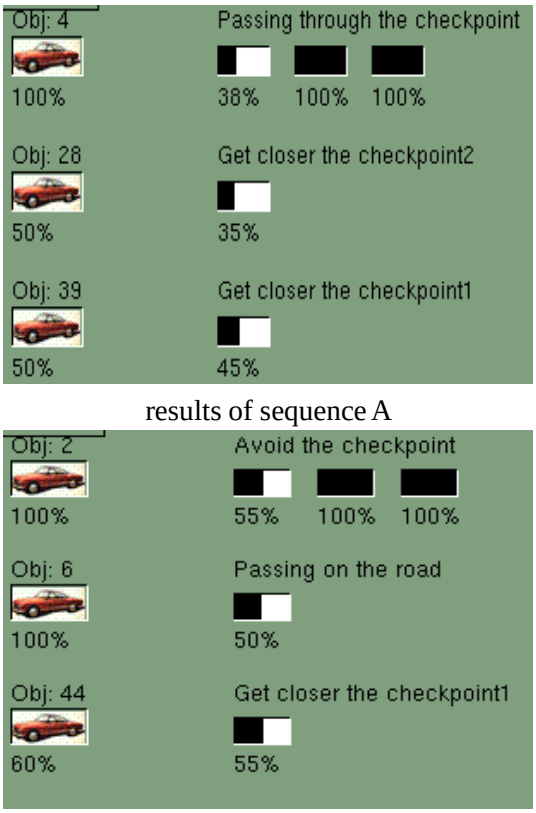


Figure 10: *Scenarios for each tracked mobile object with the highest likelihood degree*

image sequences depicting car behaviors related to the monitoring of a checkpoint. The image sequences are shown on figure 9. They have been taken from an airborne platform and acquired at 15Hz. In sequence A we can see *"a car passing through the checkpoint"* called scenario 1 (this is defined as a normal behavior) and in sequence B we can see *"a car avoiding the checkpoint"* called scenario 2 (this is defined as an abnormal behavior). We have drawn two polygons to delimit two contextual zones : the road and the checkpoint. The rectangles correspond to the bounding boxes of the moving regions associated with the detection of the car and the lines correspond to the car trajectory computed by the system.

As shown in figure 10, the likelihood of scenario 2 *"the car avoids the checkpoint"* is high enough, for this scenario to be recognized. However, scenario 1 *"the car passes through the checkpoint"* fails to be recog-

nized because the car did not stop at the checkpoint as required by sub-scenario 1.2. An abnormal behavior has been recognized with enough confidence to trigger an alarm. This example shows how to recognize a scenario composed of a multi-state combination of sub-scenarios. To obtain a good evaluation of the system, we need further tests on several image sequences. One of the most difficult issues in this type of recognition is the ability to define generic properties and scenarios that can be applied on a large number of sequences.

5 Conclusion and Future Work

We have addressed several problems related to the analysis of a video stream. The method is based on the integration of detection and tracking module and a behavior inference module, and can handle noisy data and inaccurate detections. Further research and development is planned in order to improve the stabilization by using other parametric model such as quadratic, or homographic projections. These enhancements will improve the quality of the detection algorithm. Also, a special consideration will be given to the quantification of false or non-detection rates. This will allow us to infer a confidence measure for the obtained results.

Improving the reliability of the behavior inference module will increase its stability with regard to false and non-detection of the moving objects. Handling these types of situations, which are likely to occur in processing real video streams, requires an automatic tuning of the scenario recognition methods. Further research and development is planned to model and recognize more complex scenarios. We are currently working on these issues for building a robust system and its validation through intensive testing.

Acknowledgments

We would like to thank Sarnoff/CMU/Night Vision Labs for providing some of the airborne video streams from the VSAM airborne platform used in this paper as illustrations.

References

[Brémond and Medioni, 1998] F. Brémond and G. Medioni. Scenario recognition in airborne video imagery. In *DARPA Image Understanding Workshop*, 1998.

[Brémond and Thonnat, 1996] F. Brémond and M. Thonnat. Interprétation de séquences d'images et incertitude. In *Proc. of the Rencontres sur la Logique Floue et ses Applications (LFA)*, Nancy, December 1996.

[Cohen and Herlin, 1996] I. Cohen and I. Herlin. Optical flow and phase portrait methods for environmental satellite image sequences. In *ECCV*, pages 141–150, Cambridge, April 1996.

[Cohen and Medioni, 1998] I. Cohen and G. Medioni. Detecting and tracking moving objects in video from an airborne observer. In *DARPA Image Understanding Workshop*, 1998.

[Davis and Bobick, 1997] J. W. Davis and A. F. Bobick. The representation and recognition of human movement using temporal templates. In *CVPR*, pages 928–934, Puerto-Rico, June 1997. IEEE.

[Halevi and Weinshall, 1997] G. Halevi and D. Weinshall. Motion disturbance: Detection and tracking of multi-body non rigid motion. In *CVPR*, pages 897–902, Puerto-Rico, June 1997. IEEE.

[Irani *et al.*, 1992] M. Irani, B. Rousso, and S. Peleg. Detecting and tracking multiple moving objects using temporal integration. In *ECCV*, pages 282–287, May 1992.

[Irani *et al.*, 1995] M. Irani, P. Anandan, and S. Hsu. Mosaic based representation of video sequences and their applications. In *ICCV*, pages 605–611, Cambridge, Massachusetts, June 1995. IEEE.

[Morimoto and Chellappa, 1997] C. Morimoto and R. Chellappa. Fast 3D stabilization and mosaic construction. In *CVPR*, pages 660–665, Puerto-Rico, June 1997. IEEE.

[Szeliski and Shum, 1997] R. Szeliski and H.Y. Shum. Creating full view panoramic image mosaics and environment maps. In *SIGRAPH97*, Los-Angeles, August 1997. ACM.