Behavior Research Methods
2006, 38 (3), 416-426

Video-understanding framework for automatic behavior recognition

FRANÇOIS BRÉMOND, MONIQUE THONNAT, and MARCOS ZÚÑIGA
INRIA Sophia Antipolis, ORION Group, Sophia Antipolis, France

We propose an activity-monitoring framework based on a platform called VSIP, enabling behavior recognition in different environments. To allow end-users to actively participate in the development of a new application, VSIP separates algorithms from a priori knowledge. To describe how VSIP works, we present a full description of a system developed with this platform for recognizing behaviors, involving either isolated individuals, groups of people, or crowds, in the context of visual monitoring of metro scenes, using multiple cameras. In this work, we also illustrate the capability of the framework to easily combine and tune various recognition methods dedicated to the visual analysis of specific situations (e.g., mono-/multiactors' activities, numerical/symbolic actions, or temporal scenarios). We also present other applications, using this framework, in the context of behavior recognition. VSIP has shown a good performance on human behavior recognition for different problems and configurations, being suitable to fulfill a large variety of requirements.

One of the most challenging problems in the domain of computer vision and artificial intelligence is video understanding. The research in this area concentrates mainly on the development of methods for the analysis of visual data in order to extract and process information about the behavior of physical objects in a scene.

Most approaches in the field of video understanding have incorporated methods for the detection of domain-specific events. Examples of such systems use dynamic time warping for gesture recognition (Bobick & Wilson, 1997) and self-organizing networks for trajectory classification (Owens & Hunter, 2000). The main drawback of these approaches is the usage of techniques specific only to a certain domain, which causes difficulties in applying these techniques to other areas. Therefore, some researchers have adopted a two-steps approach to the problem of video understanding: (1) A visual module is used in order to extract visual cues and primitive events, and (2) This information is used in a second stage for the detection of more complex and abstract behavior patterns (Hu, Tan, Wang, & Maybank, 2004). By dividing the problem into two subproblems we can use simpler and more domain-independent techniques in each step. The first step usually makes extensive usage of stochastic methods for data analysis, and the second step conducts a structural analysis of the symbolic data gathered at the preceding step (see Figure 1). Examples of this two-level architecture can be found in the works of Ivanov and Bobick (2000) and Vu, Brémond, and Thonnat (2003).

Correspondence concerning this article should be addressed to F. Brémond, INRIA Sophia Antipolis, ORION Group, 2004, route des Lucioles, BP93, 06902 Sophia Antipolis Cedex, France (e-mail: francois.bremond@sophia.inria.fr).

This approach is available as a platform for image sequence understanding called VSIP (Video Surveillance Interpretation Platform), which was developed by the research group ORION at INRIA (Institut National de Recherche en Informatique et en Automatique), Sophia Antipolis. VSIP is a generic environment for combining algorithms for processing and analysis of videos that allows one to flexibly combine and exchange various techniques at the different stages of the video-understanding process. Moreover, VSIP is oriented to help developers in describing their own scenarios and in building systems capable of monitoring behaviors, dedicated to specific applications.

At the first level, VSIP extracts primitive geometric features such as areas of motion. On the basis of them, objects are recognized and tracked. At the second level, those events in which the detected objects participate are recognized. For performing this task, a special representation of events is used, which is called *event description language* (Vu et al., 2003). This formalism is based on an ontology for video events presented in Brémond, Maillot, Thonnat, and Vu (2004), which defines concepts and relations between these concepts in the domain of human activity monitoring. The major concepts encompass different object types and the understanding of their behavior from the point of view of the domain expert.

In this article, we will illustrate this framework by presenting a system developed, using the VSIP platform, for recognizing people behaviors, such as fighting or vandalism, occurring in a metro scene, viewed by one or several cameras. This work has been performed in the framework of the European project ADVISOR (www-sop.inria.fr/orion/ADVISOR). Our goal is to recognize in real time behaviors involving either isolated individuals, groups of people, or crowds from real-world video streams coming from metro stations. To reach this goal, we have developed

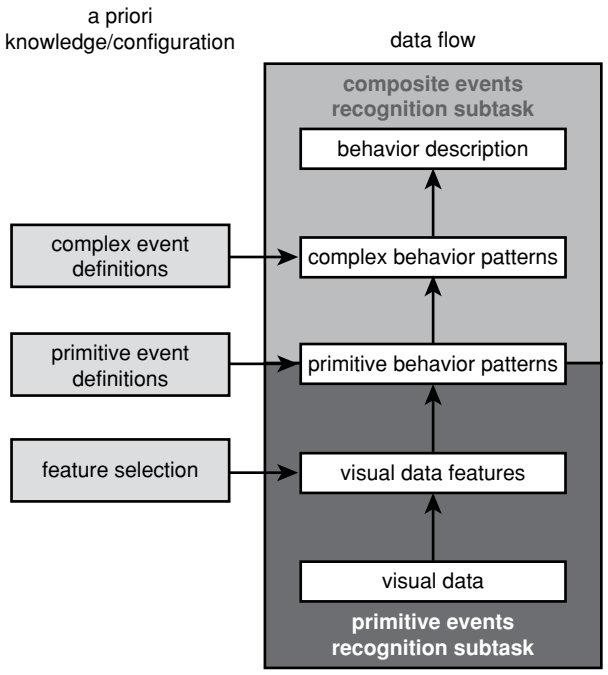

Figure 1. A general architecture of a video-understanding system. The steps depicted in the figure describe the data flow during a video-understanding process.

a system that takes as input video streams coming from cameras and generates annotation about the behaviors recognized in the video streams.

This article is organized as follows. In the second section, we will present the state of the art on behavior recognition. In the third section, we will briefly present the global system and its vision module. Then details of the behavior recognition process will be illustrated with five behavior recognition examples—*fighting*, *blocking*, *vandalism*, *overcrowding,* and *fraud* behaviors—and with an analysis of the obtained results. In the fourth section, we will discuss VSIP's capability for dealing with other applications under various configurations.

Related Work

Since the 1990s, a problem that has been focused on in cognitive vision research has been automatic video interpretation. There are now several research units and companies defining new approaches to the designing of systems that can understand specific scenarios in dynamic scenes. We define a *scenario* as a combination of states, events, or subscenarios. Behaviors are specific scenarios, defined by the users.

Three main categories of approaches are used to recognize scenarios: (1) a probabilistic/neural network combining potentially recognized scenarios, (2) a symbolic network that stores totally recognized scenarios, and (3) a symbolic network that stores partially recognized scenarios. For the computer vision community, a natural approach consists of using a probabilistic/neural network. The nodes of this network usually correspond to scenarios that are recognized at a given instant with a computed likelihood. For example, Howell and Buxton (2002) have proposed an approach to recognizing a scenario that is based on a neural network (time delay radial basis function). Hongeng, Brémond, and Nevatia (2000) have proposed a scenario recognition method that uses concurrence Bayesian threads to estimate the probability of potential scenarios.

For the artificial intelligence community, a natural way to recognize a scenario is to use a symbolic network whose nodes usually correspond to the Boolean recognition of scenarios. For example, Rota and Thonnat (2000) have used a declarative representation of scenarios defined as a set of spatiotemporal and logical constraints. They used a traditional constraint resolution technique to recognize them. To reduce the processing time for the recognition step, they proposed to check the consistency of the constraint network, using the AC4 algorithm. Gerber, Nagel, and Schreiber (2002) have defined a method for recognizing a scenario that is based on a fuzzy temporal logic. The common characteristic of these approaches is that all totally recognized behaviors are stored (recognized in the past; Vu et al., 2003).

Another approach consists of using a symbolic network and storing partially recognized scenarios (to be recognized in the future). For example, Ghallab (1996) has used the terminology chronicle to express a temporal scenario. A chronicle is represented as a set of temporal constraints on time-stamped events. The recognition algorithm keeps and updates partial recognition of scenarios, using the propagation of temporal constraints based on the RETE algorithm. Their applications are dedicated to the control of turbines and telephonic networks. Chleq and Thonnat (1996) have made an adaptation of temporal constraints propagation for video surveillance. In the same period, Pinhanez and Bobick (1998) used Allen's interval algebra to represent scenarios and presented a specific algorithm to reduce its complexity.

All these techniques allow an efficient recognition of scenarios, but there are still some temporal constraints that cannot be processed. For example, most of these approaches require that the scenarios are bounded in time (Ghallab, 1996) or process temporal constraints and atemporal constraints in the same way (Rota & Thonnat, 2000).

Another problem that has captured the attention of researchers recently is the problem of unsupervised behavior learning and recognition, consisting of the capability of a vision interpretation system to learn and detect the frequent scenarios of a scene without requiring the prior definition of behaviors by the user. The unsupervised behavior learning and recognition problem in the field of computer vision has been addressed in only a few works. Most of the approaches are confined to a specific domain and take advantage of domain knowledge in order, for example, to choose a proper model or to select features. One of the most widely used techniques for learning scenarios in an unsupervised manner is the topology

of a Markov model. Brand and Kettnaker (2000) have used an entropy-based function, instead of the maximum-likelihood estimator in the E-step of the EM-algorithm, for learning parameters of hidden Markov models (HMMs). This leads to a concentration of the transitional probabilities on just several states that correspond, in most of the cases, to meaningful events. Another approach is based on variable length Markov models that can express the dependence of a Markov state on more than one previous state (Galata, Cohn, Magee, & Hogg, 2002). Although this method learns good stochastic models of the data, it cannot handle temporal relations. A further, similar technique is based on hierarchical HMMs, whose topology is learned by merging and splitting states (Xie, Chang, Divakaran, & Sun, 2003). The advantage of the techniques above for topology learning of Markov models is that they work in a completely unsupervised way. In addition, they can be used after the learning phase to efficiently recognize the discovered events. On the other hand, these methods deal with simple events and are not capable of creating concept hierarchies, and there is no guaranty that the states of these models correspond to meaningful events.

A different approach for this problem has been proposed by Toshev, Brémond, and Thonnat (2006). In this work, the a priori algorithm from the field of data mining is used to propose a method for the unsupervised learning of behaviors from videos. The developed algorithm processes a set of generic primitive events and outputs the frequent patterns of these primitive events, also interpreted as frequent composite events. In a second step, models of these composite events are automatically generated (i.e., learned) in the event description language defined by Vu et al. (2003), which can be used to recognize the detected composite events in new videos. This application was used for detecting frequent behaviors on a parking lot monitoring system.

This review of the state of the art shows the large diversity of video-understanding techniques in automatic behavior recognition. The challenge is to efficiently combine these techniques so as to address the great diversity of the real world.

Video-Understanding Platform

The video interpretation platform is based on the cooperation of a vision and a behavior recognition module, as shown in Figure 2.

The vision module is composed of three tasks. First, a motion detector and a frame-to-frame tracker generate a graph of mobile objects for each calibrated camera. Second, a combination operation is performed to combine the graphs computed for each camera into a global one. Third, this global graph is used for long-term tracking of individuals, vehicles, groups of people, and crowds as the scene evolves.

For each tracked actor, the behavior recognition module performs three levels of reasoning: states, events, and scenarios.

On top of that, we use 3-D scene models (i.e., geometric models of the empty scene, including the furniture), one for each camera, as a priori contextual knowledge of the observed scene. In a scene model, we define the 3-D positions and dimensions of the static scene objects (e.g., a bench or a ticket vending machine) and the zones of interest (e.g., an entrance zone). Semantic attributes (e.g., fragile) can be associated to the objects or zones of interest, to be used in the behavior recognition process.

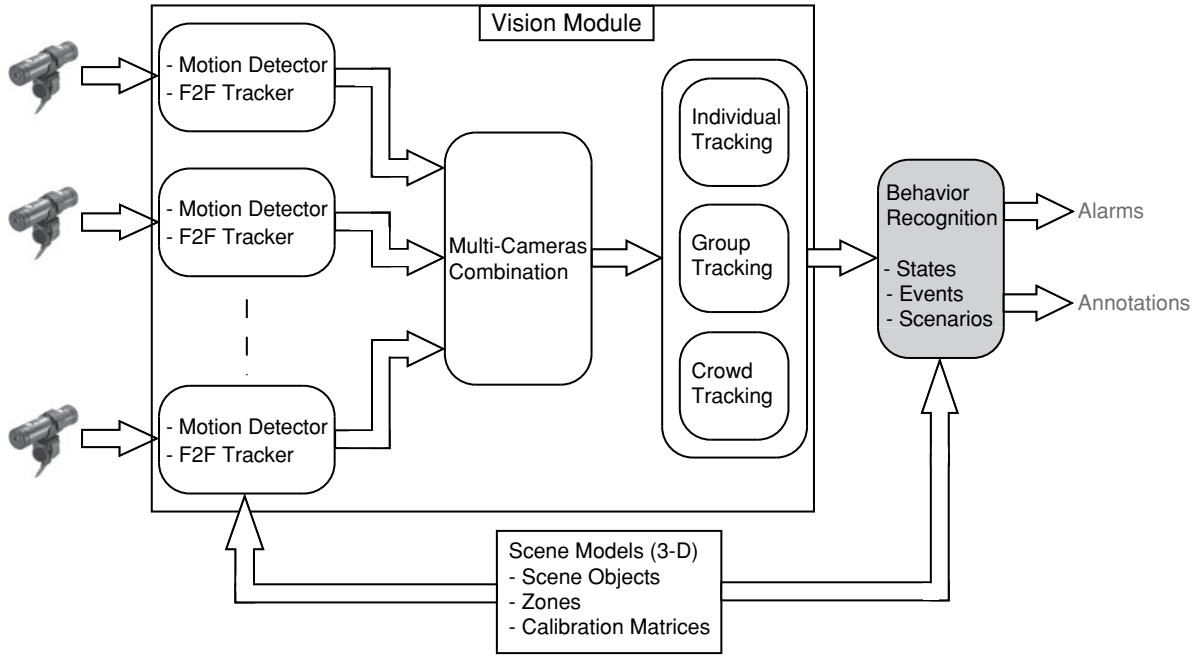

Figure 2. Video interpretation system.

In this article, we will focus on the behavior recognition process, since it is our current focus of interest.[1] The goal of this process is to recognize specific behaviors occurring in an observed scene. A main problem in behavior recognition is the ability to define and reuse methods to recognize specific behaviors, knowing that the perception of behaviors is strongly dependent on the site, the camera view point, and the individuals involved in the behaviors. Our approach consists of defining a formalism allowing us to write and easily reuse all the methods needed for the recognition of behaviors. This formalism is based on three main ideas.

1. The formalism should be flexible enough to allow various types of *operators* to be defined (e.g., a temporal filter or an automaton). We use *operator* as an abstract term to define programs. This term will be defined in the following section.

2. All the needed knowledge for an operator should be explained within the operator, so that it can be easily reused.

3. The description of the operators should be declarative, in order to build an extensible library of operators.

Behavior representation. We call an *actor* of a behavior any scene object involved in the behavior, including static objects (equipment or zones of interest), individuals, groups of people, or crowds. The entities needed to recognize behaviors correspond to different types of concepts, which are the following.

1. The *basic properties.* A basic property is a characteristic of an actor, such as its trajectory or its speed.

2. The *states*. A state describes a situation characterizing one or several actors defined at time *t* (e.g., a group is agitated) or a stable situation defined over a time interval. For the *an individual stays close to the ticket vending machine* state, two actors are involved: an individual and a piece of equipment.

3. The *events*. An event is a change of states at two consecutive times (e.g., a group enters a zone of interest).

4. The *scenarios*. A scenario is a combination of states, events, or subscenarios. Behaviors are specific scenarios (dependent on the application) defined by the users. For example, to monitor metro stations, end-users have defined five targeted behaviors: fraud, fighting, blocking, vandalism, and overcrowding.

To compute all the needed entities for the recognition of behaviors, we use a generic framework based on the definition of *operators*, which are program descriptions containing the following four elements.

1. *Name*. A name indicates the entity to be computed, such as the *an individual is walking* or *the trajectory is straight* state.

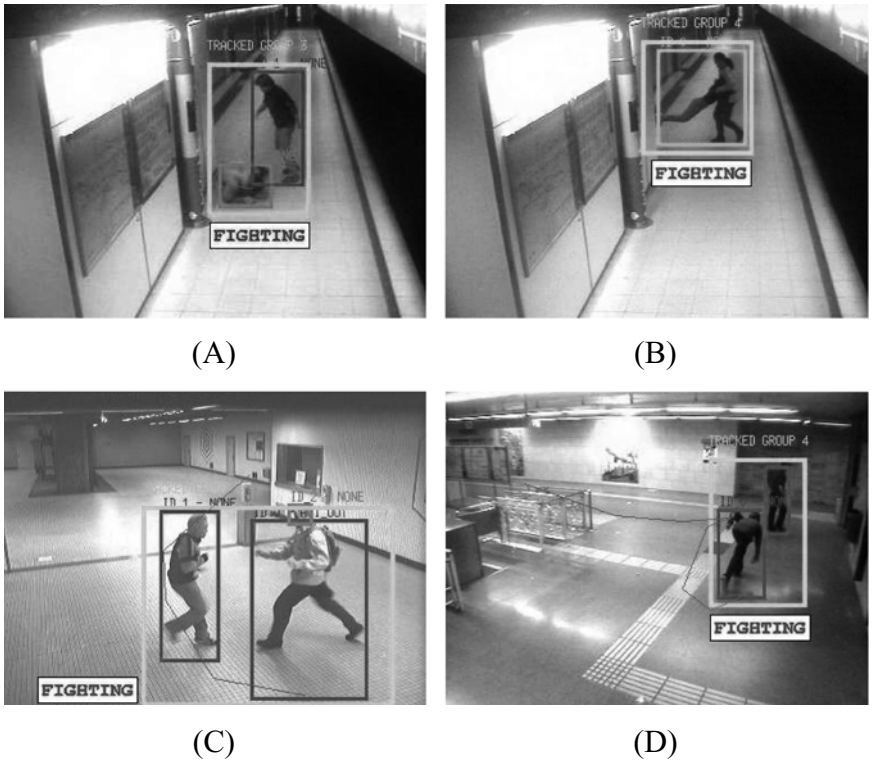

Figure 3. Four methods combined by an *and/or* tree to recognize the *fighting* behavior. Each image illustrates a configuration where one method is more appropriate for recognizing the behavior: (A) a person lying on the floor surrounded by people, (B) a significant variation of the group width, (C) a quick separation of the people inside a group, and (D) a significant variation of the group trajectory.

2. *Input*. Input gives a description of input data. There are two types of input data: basic properties characterizing an actor and subentities computed by other operators.

3. *Body*. The body contains a set of competitive methods to compute the entity. All these methods are able to compute this entity, but they are specialized depending on different configurations. For example, to compute the *fighting* scenario, there are four methods (as shown in Figure 3). For example, one method computes the evolution of the lateral distance between the people inside a group. A second one detects whether someone, surrounded by people, has fallen on the floor.

4. *Output*. The output contains the result of the entity computation accessible by all the other operators. This result corresponds to the value of the entity at the current time. This generic framework, based on the definition of *operators*, provides two advantages: It first enables us to test a set of methods for computing an entity independently of other entities. So we can locally modify the system (the methods for computing an entity) while keeping it globally consistent (without modifying the meaning of the entity). Second, the network of operators for recognizing one scenario is organized as a hierarchy. The bottom of the hierarchy is composed of states, and the top corresponds to the scenario to be recognized. Several intermediate levels, composed of state(s) or event(s), can be defined.

Behavior recognition. We have defined four types of methods, depending on the type of entities.

1. *Basic properties methods*. We use dedicated routines to compute properties characterizing actors, such as trajectory, speed, and direction. For example, we use a polygonal approximation to compute the trajectory of an individual or a group of people.

2. *State methods*. We use numerical methods that include the computation of 3-D distance for states dealing with spatial relations (e.g., *an individual is close to the ticket vending machine*), the evolution of temporal features for states dealing with temporal relations (e.g., *the size of a group of people is constant*), speed for states dealing with spatiotemporal relations (e.g., *an individual is walking*), and the combination of substates computed by other operators. The output of these numerical methods is then classified to obtain a symbolic value.

3. *Event methods*. We compare the status of states at two consecutive instants. The output of an event method is Boolean: The event is either detected or not detected. For example, the *a group of people enters a zone of interest* event is detected when the *a group of people is inside a zone of interest* state changes from false to true.

4. *Scenario methods*. For simple scenarios (composed of only one state), we verify that a state has been detected during a predefined time period, using a temporal filter. For sequential scenarios (composed of a sequence of states), we use finite state automatons. An automaton state corresponds to a state, and a transition to an event. An automaton state also corresponds to an intermediate stage before the complete recognition of the scenario. We have used an automaton to recognize the *blocking* and *fraud* scenarios, as described in Figures 4 and 5.

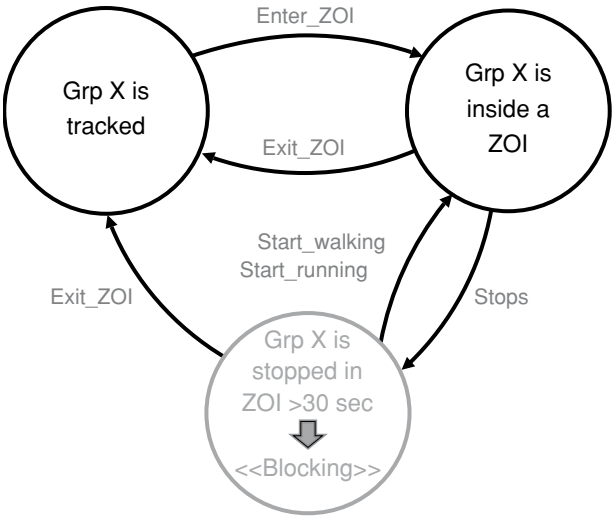

Figure 4. To check whether a group of people is blocking a zone of interest (ZOI), we have defined an automaton with three states: (A) A group is tracked, (B) the group is inside the ZOI, and (C) the group has stopped inside the ZOI for at least 30 sec.

For composed scenarios defining a single unit of movement composed of subscenarios, we use Bayesian networks, as proposed by Hongeng and Nevatia (2001), or *and*/*or* trees of subscenarios, as illustrated in Figure 6. The structure of *and*/*or* trees, even if not continuous, is a good compromise between the usability of knowledge representation by experts and correspondence with observations on videos. A description of Bayesian networks for scenario recognition can be found in Moenne-Locoz, Brémond, and Thonnat (2003). We have defined one Bayesian network for recognizing the *violence* behavior as being composed of two subscenarios: *internal violence* (e.g., erratic motion of people inside a group) and *external violence* (e.g., quick evolution of the trajectory of the group). The structures of Bayesian networks are statistically learned by an offline process, allowing adaptability for different kinds of behaviors but lacking in usage of knowledge from an expert. In contrast, *and*/*or* trees can represent more precise knowledge from experts, but not necessarily in correspondence with the observed videos. Both methods are time demanding, either to collect representative videos or to tune the parameters corresponding to the expert knowledge. Also, both need a learning stage (statistical or manual) to adjust the parameters of the network, using ground truth (videos annotated by operators). Bayesian networks are optimal given ground truth, but *and*/*or* trees are easier to tune and to adapt to new scenes.

For scenarios with multiple actors involved in complex temporal relationships, we use a network of temporal variables representing subscenarios, and we backtrack temporal constraints among the already recognized subscenarios, as proposed by Vu et al. (2003). For users to be able to define the behaviors they want to recognize, an event description language is used as a formalism for describing the events characterizing these behaviors (Vu et al.,

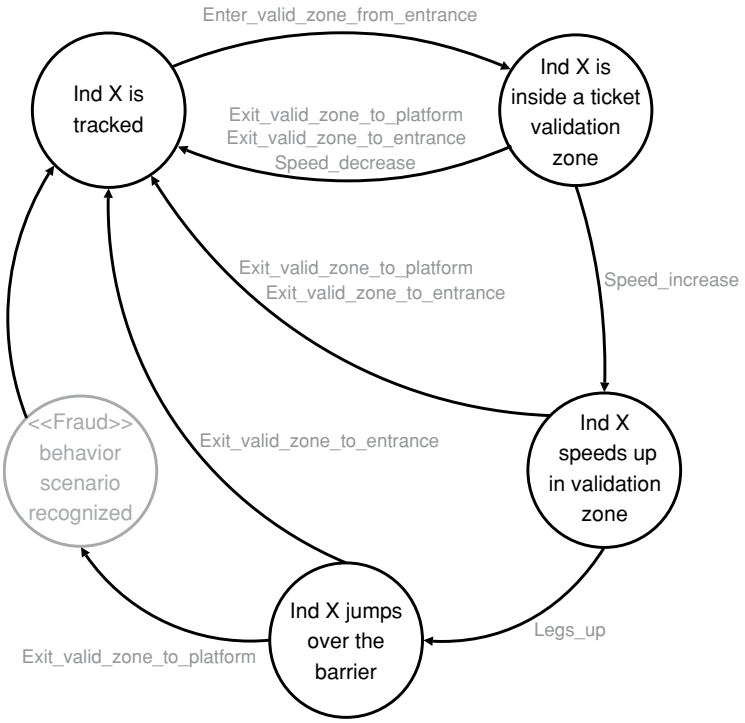

Figure 5. To check whether an individual is jumping over the barrier without validating his ticket, we have defined an automaton with five states: (A) An individual is tracked, (B) the individual is at the beginning of the validation zone, (C) the individual has a high speed, (D) the individual is over the barrier with legs up, and (E) the individual is at the end of the validation zone.

2003). The purpose of this language is to give a formal but also intuitive, easily understandable, and simple tool for describing events. All these features can be achieved by defining events in a hierarchical way and reusing definitions of simple events in more complex ones. A definition of an event consists of the following.

1. *Event name.*
2. *List of physical objects involved in the event*. A *physical object* can be a mobile object or a static one. Typical examples are humans, vehicles, zones, and equipment.
3. *List of components representing subevents that describe simpler activities*.

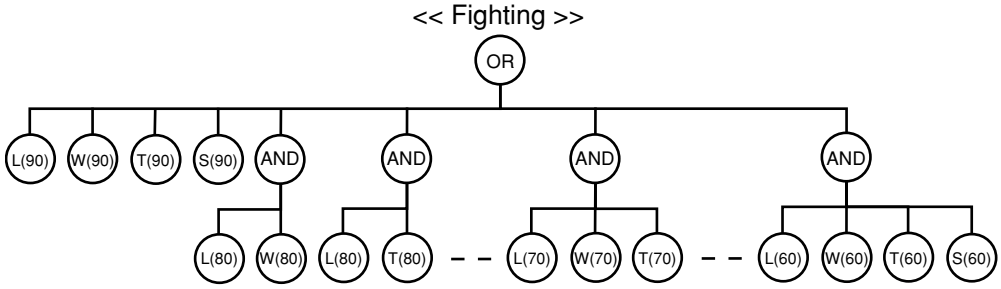

Figure 6. To recognize whether a group of people is fighting, we have defined an *and*/*or* tree composed of four basic scenarios: (L) a person lying on the floor surrounded by people, (W) a significant variation of the group width, (S) a quick separation of people inside the group, and (T) a significant variation of the group trajectory. Given these four basic scenarios, we were able to build an *or* node with all combinations (corresponding to 15 subscenarios) of the basic scenarios. These combinations correspond to *and* nodes with one up to four basic scenarios. The more basic scenarios there are in the *and* nodes, the less strict is the recognition threshold of each basic scenario. For example, when there is only one basic scenario [e.g., L(90)], the threshold is 90, and when there are four basic scenarios, the threshold is 60. To parameterize these thresholds, we have performed a learning stage consisting of a statistical analysis of the recognition of each basic scenario.

4. *List of constraints expressing additional conditions*. The *constraints* can be *spatial* or *temporal*, depending on their meaning. In both cases, we can have *symbolic* or *numeric* form. For example, a *spatial symbolic constraint* is *object inside zone*, and a *spatial numeric constraint* can be defined as follows: *distance*(object$_1$, object$_2$) $\leq$ *threshold*. In the case of a *temporal constraint*, we can also have a *numeric* form, such as *duration*(event) $\leq$ 20[secs], or a symbolic form; event$_1$ *before* event$_2$. Figure 7 depicts an example of a complex scenario—*vandalism*: A person *p* tries to *break up* an equipment *eq*—using the formalism of Vu et al. (2003). This scenario will be recognized if a sequence of five events, described in Figure 8, has been detected.

event (*vandalism*,
physical_objects(**p**: Person, **eq**: Equipment),
components((**state** $\mathbf{e_1}$: **p** ***far_from*** **eq**),
(**state** $\mathbf{e_2}$: **p** ***stays_at*** **eq**),
(**event** $\mathbf{e_3}$: **p** ***moves_away_from*** **eq**),
(**event** $\mathbf{e_4}$: **p** ***moves_close_to*** **eq**),
(**state** $\mathbf{e_5}$: **p** ***stays_at*** **eq**)
),
constraints(($\mathbf{e_1}$ ***before*** $\mathbf{e_2}$) ($\mathbf{e_2}$ ***before*** $\mathbf{e_3}$)
($\mathbf{e_3}$ ***before*** $\mathbf{e_4}$) ($\mathbf{e_4}$ ***before*** $\mathbf{e_5}$)
)
)
)

Figure 7. Definition of the *vandalism* behavior. A sequence of five events, which represent relative positions between a person *p* and an equipment *eq*, must be detected in order to recognize this behavior.

Behavior recognition results. The platform has been tested in different situations and has been validated in the metro-monitoring application. The behavior recognition module runs on a PC Linux and processes four tracking outputs corresponding to 4 cameras with a rate of five images per second. We have tested the whole video interpretation system (including motion detection, tracking, and behavior recognition) on videos coming from 10 cameras at the Barcelona and Brussels metros. We correctly recognized the *fraud* scenario 6/6 (six times out of six) (Figure 9A), the *vandalism* scenario 4/4 (Figure 9B), the *fighting* scenario 20/24 (Figure 3), the *blocking* scenario 13/13 (Figure 9C), and the *overcrowding* scenario 2/2 (Figure 9D). We also tested the system over long sequences (10 h) in order to check the robustness over false alarms. For each behavior, the rate of false alarm is two for fraud, zero for vandalism, four for fighting, one for blocking, and zero for overcrowding.

Moreover, in the framework of the European project ADVISOR, the video interpretation system has been ported on Windows and installed at the Barcelona metro in March 2003 to be evaluated and validated. This evaluation was done by Barcelona and Brussels video surveillance metro operators during 1 week at the Sagrada Familia metro station. Together with this evaluation, a demonstration has been presented to various guests, including the European Commission, project reviewers, and representative of the Brussels and Barcelona metros, to validate the system. The evaluation and the demonstration were conducted using both live and recorded videos: four channels in parallel, composed of three recorded sequences and one live input stream from the main hall of the station. The recorded sequences enabled us to test the system with rare scenarios of interest (e.g., fighting, jumping over the barrier, and vandalism), whereas the live camera allowed us to evaluate the system against scenarios that often happen (e.g., overcrowding) and ones that occurred during the demonstration and also to evaluate the system against false alarms. In total, out of 21 fighting incidents in all the recorded sequences, 20 alarms were correctly generated, giving a very good detection rate of 95%. Out of 9 blocking incidents, 7 alarms were generated, giving a detection rate of 78%. Out of 42 instances of jumping over the barrier, including repeated incidents, the behavior was detected 37 times, giving a success rate of 88%. The two sequences of vandalism were always detected over the six instances of vandalism, giving a perfect detection rate of 100%. Finally, the overcrowding incidents were also consistently detected by the live camera, with some 28 separate events being well detected. The results are summarized in Table 1.

In conclusion, the ADVISOR demonstration has been evaluated positively by end-users and by the European Committee. The algorithm responded successfully to the

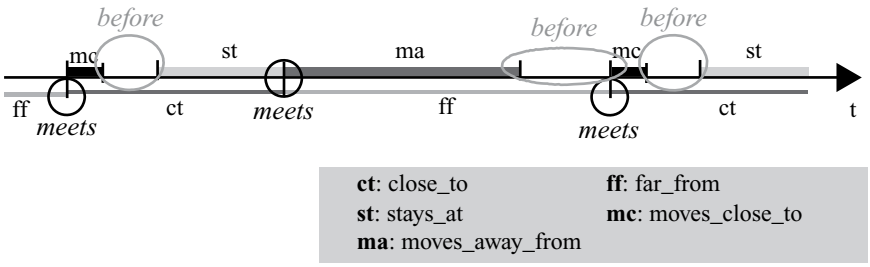

Figure 8. Temporal constraints for the states and events constituting a *vandalism* scenario.

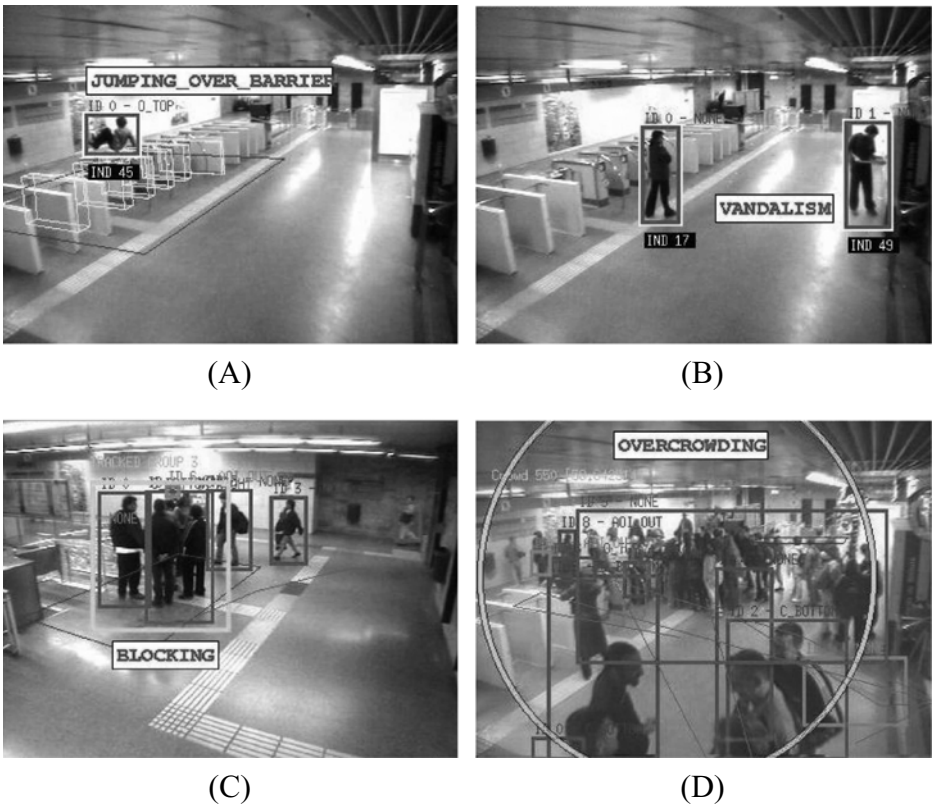

Figure 9. Four behaviors selected by end-users and recognized by the video interpretation system: (A) *fraud*, recognized by an automaton; (B) *vandalism*, recognized by a temporal constraint network; (C) *blocking*, recognized by an automaton; and (D) *overcrowding*, recognized by an *and*/*or* tree.

input data, with high detection rates, with fewer than 5% false alarms, and with all the reports being above 70% accurate.

Discussion

The evaluation of the metro application has validated the ability of VSIP for the recognition of human behaviors. End-users of the application evaluated it positively, because of its high detection rate on different scenarios.

We have been also working closely with end-users on other application domains. For example, with VSIP, we have built six systems, which have been validated by end-users.

1. The activity-monitoring system in metro stations has been validated by metro operators from Barcelona and Brussels.

2. A system for detecting abnormal behaviors inside moving trains (see Figure 10A), able to handle situations

Table 1
Results of the Technical Validation of the Metro-Monitoring System

Scenario Name	Number of Behaviors	Number of Recognized Instances	% of Recognized Instances	Accuracy (%)	Number of False Alerts
Fighting	21	20	95	61	0
Blocking	9	7	78	60	1
Vandalism	2	2	100	71	0
Jumping over the barrier	42	37	88	100	0
Overcrowding	7	7	100	80	0
Total	81	73	90	85	1

Note—For each scenario, we report in particular the percentage of recognized instances of this scenario (fourth column) and the accuracy in time of the recognition (the percentage of the duration of the shown behavior *covered* by the generation of the corresponding alert by the system). This value is an average over all the scenario instances (fifth column).

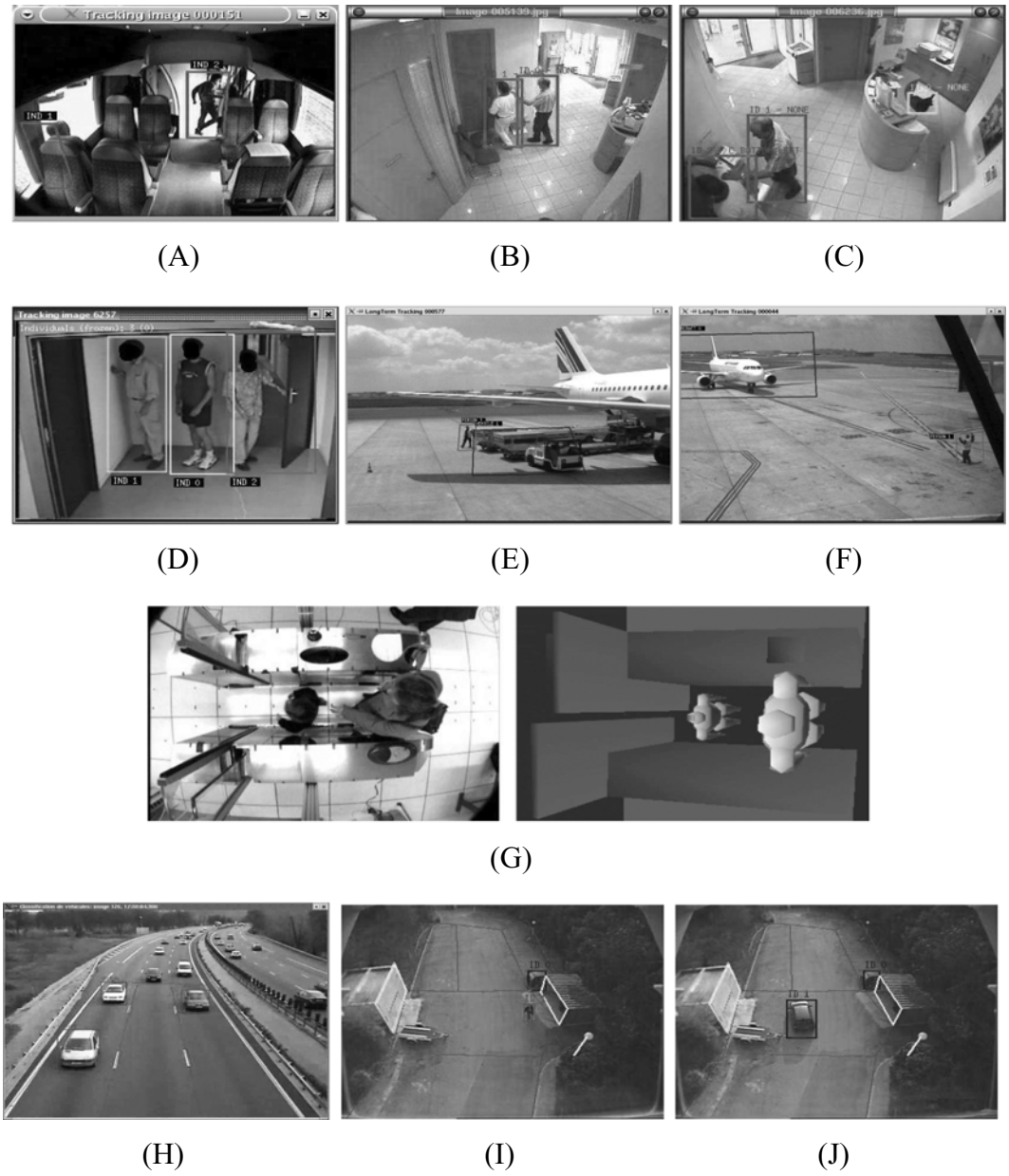

Figure 10. Six other systems derived from VSIP to handle different applications. Image A illustrates a system for unruly behavior detection inside trains. Images B and C, taken with two synchronized cameras with overlapping field of view working in a cooperative way, depict a bank agency monitoring system detecting an abnormal *bank attack* scenario. Image D illustrates a single-camera system for a lock chamber access control application for building entrances. Images E and F show a system for apron monitoring at airports; this system combines a total of eight surveillance cameras with overlapped fields of view. Image G depicts a multisensor system for metro access control; this system uses the information obtained from a static top camera and a set of lateral visual sensors to perform shape recognition of people. The right image corresponds to the understanding of the scene projected in a 3-D virtual representation. Image H illustrates a highway traffic monitoring application. Finally, Images I and J depict a parking lot monitoring system for learning behavior patterns, using an unsupervised technique.

in which people are partially occluded by the train equipment (such as seats), has been validated over three scenarios (graffiti, theft, and begging).

3. A bank agency monitoring system has been installed and validated in four bank agencies around Paris (Georis, Mazière, Brémond, & Thonnat, 2004; see Figures 10B and 10C).

4. A lock chamber access control system for buildings security has been validated with more than 140 recorded videos and with a live prototype (see Figure 10D).

5. An apron-monitoring system has been developed for an airport[2] at which vehicles of various types are evolving in a cluttered scene (see Figures 10E and 10F). The dedicated system has been validated for five servicing scenarios (GPU vehicle arrival, parking aircraft, refueling aircraft, loading/unloading a container, and towing aircraft).

6. A metro access control system has been tested by end-users with more than 200 videos and with a live prototype (Bui Ngoc, Brémond, Thonnat, & Faure, 2005; see Figure 10G).

Some of these applications are illustrated in Figure 10. They present several characteristics that make them interesting for research purposes.

1. The observed scenes vary from large open spaces (like metro halls) to small and closed spaces (corridors and lock chambers).

2. Cameras can have both nonoverlapping (as in metro station and lock chamber systems) and overlapping (metro stations and bank agencies) fields of view.

3. Humans can interact with the equipment (such as ticket vending machines, access control barriers, bank safes, and lock chambers doors) either in simple ways (*open*/*close*) or in more complex ones (such as the interaction occurring during *vandalism against equipment* or *jumping over the barrier* scenarios).

We are currently building various other applications for VSIP. For instance, a system concerning traffic monitoring on the highway has been built in a few weeks to show the adaptability of the platform (see Figure 10H). Other applications are envisaged, such as home care (monitoring of elderly people at home), multimedia (e.g., intelligent camera for video conferencing), and animal behavior recognition (e.g., insect parasitoids attacking their hosts). The VSIP platform has currently been extended to learn behavior models, using unsupervised learning techniques, to be applied to parking lot monitoring (Toshev et al., 2006; see Figures 10I and 10J).

VSIP has shown its ability to automatically recognize and analyze human behaviors. However, some limitations remain. Video-understanding systems are often difficult to configure and install. To have an efficient system handling the variety of the real world, extended validation is needed. Automatic capability to adapt to dynamic environments should be added to the platform, which is a new topic of research. Nevertheless, the diversity of applications for which VSIP has been used shows that this platform is suitable for fulfilling many types of requirements.

Conclusion

We have presented a video-understanding platform to automatically recognize human behaviors involving individuals, groups of people, crowds, and vehicles, by detecting visual invariants. A visual invariant is a visual property or clue that characterizes a behavior.

Different methods have been defined to recognize specific types of behaviors under different scene configurations. To also experiment with various software solutions, all these methods have been integrated into a coherent framework that enables one to modify a given method locally and easily. The VSIP platform has been fully evaluated for several applications. For instance, the system has been tested offline and has been evaluated, demonstrated, and successfully validated in live conditions during 1 week at the Barcelona metro, in March 2003.

This approach can also be applied to the biological domain. For instance, in 2005, we developed a system based on the VSIP platform that detects the behaviors of a trichogramma interacting with butterfly eggs. This application corresponds to a behavioral study whose goal is to understand how a trichogramma introduces its eggs on butterfly eggs, thereby contributing to plague control in agriculture.

Hence, we believe that VSIP shows great potential as a tool for recognition and analysis of human behaviors in very different configurations.

VSIP still presents some limitations when environmental conditions suddenly change or the complexity of a scene increases, which makes necessary the improvement of vision modules to ensure robustness. Moreover, VSIP requires predefinition of events for the detection of behaviors, which can be a very hard task. To cope with this limitation, an unsupervised frequent events detection algorithm (Toshev et al., 2006) has shown encouraging preliminary results. This algorithm is capable of extracting the most significant events in a scene without behavior predefinition by the end-user.

REFERENCES

AVITRACK European Research Project (2006). Available at www.avitrack.net.

Bobick, A. F., & Wilson, A. D. (1997). A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **19**, 1325-1337.

Borg, M., Thirde, D., Ferryman, J., Fusier, F., Brémond, F., & Thonnat, M. (2005). Video event recognition for aircraft activity monitoring. In *Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems* (pp. 1102-1107). Los Alamitos, CA: IEEE Computer Society Press.

Brand, M., & Kettnaker, V. (2000). Discovery and segmentation of activities in video. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **22**, 844-851.

Brémond, F., Maillot, N., Thonnat, M., & Vu, T. V. (2004). *RR5189 Ontologies for video events* (Research Rep. 5189). Sophia Antipolis, France: Orion Team, Institut National de Recherche en Informatique et Automatique (INRIA).

Bui Ngoc, H.-B., Brémond, F., Thonnat, M., & Faure, J.-C. (2005). Shape recognition based on a video and multi-sensor system. In *Proceedings of the IEEE International Conference on Advanced Video and Signal-Based Surveillance* (pp. 230-235). Los Alamitos, CA: IEEE Computer Society Press.

Chleq, N., & Thonnat, M. (1996). Real-time image sequence interpretation for video-surveillance applications. In *Proceedings of the IEEE International Conference on Image Processing* (Vol. 2, pp. 801-804). Los Alamitos, CA: IEEE Computer Society Press.

Cupillard, F., Brémond, F., & Thonnat, M. (2004). Video understanding for metro surveillance. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control* (Vol. 1, pp. 186-191). Los Alamitos, CA: IEEE Computer Society Press.

Galata, A., Cohn, A., Magee, D., & Hogg, D. (2002). Modeling interaction using learnt qualitative spatio-temporal relations and variable length Markov models. In F. van Harmelen (Ed.), *Proceedings of the 15th European Conference on Artificial Intelligence*, (pp. 741-745). Amsterdam: IOS Press.

Georis, B., Mazière, M., Brémond, F., & Thonnat, M. (2004). A video interpretation platform applied to bank agency monitoring. In

Proceedings of the International Conference on Intelligent Distributed Surveillance Systems (pp. 46-50). London: Institution of Electrical Engineers.

Gerber, R., Nagel, H., & Schreiber, H. (2002). Deriving textual descriptions of road traffic queues from video sequences. In *Proceedings of the 15th European Conference on Artificial Intelligence* (pp. 736-740). Amsterdam: IOS Press.

Ghallab, M. (1996). On chronicles: Representation, on-line recognition and learning. In *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning* (pp. 597-606). Cambridge, MA: Kaufmann.

Hongeng, S., Brémond, F., & Nevatia, R. (2000). Representation and optimal recognition of human activities. In *IEEE Proceedings of the International Conference on Computer Vision and Pattern Recognition* (Vol. 1, pp. 818-825). Los Alamitos, CA: IEEE Computer Society Press.

Hongeng, S., & Nevatia, R. (2001). Multi-agent event recognition. In *Proceedings of the 8th International Conference on Computer Vision* (Vol. 2, pp. 84-91). Los Alamitos, CA: IEEE Computer Society Press.

Howell, A., & Buxton, H. (2002). Active vision techniques for visually mediated interaction. *Image & Vision Computing*, **20**, 861-871.

Hu, W., Tan, T., Wang, L., & Maybank, S. (2004). A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man, & Cybernetics: Pt. C. Applications & Reviews*, **34**, 334-352.

Ivanov, Y. A., & Bobick, A. F. (2000). Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **22**, 852-872.

Moenne-Locoz, N., Brémond, F., & Thonnat, M. (2003). Recurrent Bayesian network for the recognition of human behaviors from video. In J. L. Crowley, J. H. Piater, M. Vineze, & L. Paletta (Eds.), *Computer Vision Systems: Proceedings of the 3rd International Conference on Computer vision systems* (pp. 68-77). Berlin: Springer.

Owens, J., & Hunter, A. (2000). Application of the self-organizing map to trajectory classification. In *Proceedings of the IEEE International Workshop on Visual Surveillance* (pp. 77-83). Los Alamitos, CA: IEEE Computer Society Press.

Pinhanez, C., & Bobick, A. (1998). Human action detection using pnf propagation of temporal constraints. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (pp. 898-904). Los Alamitos, CA: IEEE Computer Society Press.

Rota, N., & Thonnat, M. (2000). Activity recognition from video sequences using declarative models. In *Proceedings of the 14th European Conference on Artificial Intelligence* (pp. 673-680). Amsterdam: IOS Press.

Toshev, A., Brémond, F., & Thonnat, M. (2006). An a priori-based method for frequent composite event discovery in videos. In *Proceedings of the IEEE International Conference on Computer Vision Systems* [CD-ROM]. Los Alamitos, CA: IEEE Computer Society Press.

Vu, V., Brémond, F., & Thonnat, M. (2003). Automatic video interpretation: A novel algorithm for temporal scenario recognition. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (pp. 1295-1302). San Francisco: Morgan Kaufmann.

Xie, L., Chang, S.-F., Divakaran, A., & Sun, H. (2003). Unsupervised mining of statistical temporal structures in video. In A. Rosenfeld, D. Doermann, & D. Dementhon (Eds.), *Video mining* (pp. 279-307). Norwell, MA: Kluwer.

NOTES

1. For more details on the vision-processing module, see Cupillard, Brémond, and Thonnat (2004).

2. In the framework of the AVITRACK European Project; see AVITRACK (2006); Borg et al. (2005).

(Manuscript received September 30, 2005;
revision accepted for publication December 22, 2005.)