EURASIP Journal on Applied Signal Processing 2005:14, 2359–2374
© 2005 Alberto Avanzi et al.

Design and Assessment of an Intelligent Activity Monitoring Platform

Alberto Avanzi
i-DTV Group, Bull SA, avenue Jean Jaurès, 78340 Les Clayes-Sous-Bois, France
Email: alberto.avanzi@sophia.inria.fr

François Brémond
ORION Group,INRIA Sophia Antipolis, 2004 route des Lucioles, B.P. 93, 06902 Sophia Antipolis Cedex, France
Email: francois.bremond@sophia.inria.fr

Christophe Tornieri
ORION Group, INRIA Sophia Antipolis, 2004 route des Lucioles, B.P. 93, 06902 Sophia Antipolis Cedex, France
Email: christophe.tornieri@sophia.inria.fr

Monique Thonnat
ORION Group, INRIA Sophia Antipolis, 2004 route des Lucioles, B.P. 93, 06902 Sophia Antipolis Cedex, France
Email: monique.thonnat@sophia.inria.fr

Received 26 January 2004; Revised 25 January 2005

We are interested in designing a reusable and robust activity monitoring platform. We propose three good properties that an activity monitoring platform should have to enable its reusability for different applications and to insure performance quality: (1) modularity and flexibility of the architecture, (2) separation between the algorithms and the a priori knowledge they use, and (3) automatic evaluation of algorithm results. We then propose a development methodology to fulfill the last two properties. The methodology consists in the interaction between end-users and developers during the whole development of a specific monitoring system. To validate our approach, we present a platform used to generate activity monitoring systems dedicated to specific applications, we also describe in details the technical validation and the end-user assessment of an automatic metro monitoring system built with the platform and briefly the validation results for bank agency monitoring and building access control.

Keywords and phrases: intelligent vision platform, video surveillance, autonomous system, evaluation, generic platform.

1. INTRODUCTION

The task of developing algorithms able to recognize human activities in video sequences has been an active field of research for the last ten years. Nevertheless, the lack of genericity and robustness of the proposed solutions is still an open problem. To break down this challenging problem into smaller and easier ones, a possible approach is to limit the field of application to specific activities in well-delimited environments. So the scientific community has led researches on automatic traffic surveillance on highways, on pedestrian and vehicle interaction analysis in parking lots or roundabouts [1], or on human activity monitoring in outdoor (like streets and public places) or indoor (like metro stations, bank agencies, houses) environments [2, 3, 4].

We believe that to obtain a reusable and performant activity monitoring platform, a unique global and sophisticated algorithm is not adapted because it cannot handle the large diversity of real-world applications. However, such a platform can be achieved if many algorithms can be easily combined and integrated to handle such diversity. Therefore, we propose to use software engineering and knowledge engineering techniques to meet these major requirements.

To illustrate what we mean when we speak of "an activity monitoring platform," we first describe the platform we have developed during the last ten years, called video surveillance intelligent platform (VSIP). VSIP is a toolbox helping a developer to build activity monitoring systems (AMSs) dedicated to specific applications.

Then we address three general properties of an activity monitoring platform to insure performance quality and platform reusability. Our goals are (1) to have a platform which allows the building of new activity monitoring systems dedicated to different applications and (2) to insure the quality of the results given by any system built with the platform. While defining and describing each property, we show how they are fulfilled in VSIP.

The first property is *modularity and flexibility of the architecture*. This is a classical software engineering property. To use a platform for deriving new systems for specific applications, it is often necessary to add new algorithms, to remove existing ones, or to replace some of them with others which have the same functionality but are able to cope with more challenging situations. For example, when addressing for the first time an application where the light can be switched on and off, it is necessary to develop an algorithm able to handle instantaneous illumination changes. This algorithm has then to be integrated to the platform in order to be used, without requiring additional development, by any AMS derived from the platform. To allow this kind of "plugging-unplugging" feature, the platform has to be developed keeping in mind a well-defined modular architecture, based, for example, upon clear interfaces between modules (in order to insure information sharing and exchanging between all the system modules). At the same time, modules have to be flexible in order to be reused in different situations. A natural way to obtain flexibility is to outsource parameters (to allow automatic parameter tuning from the highest level). In our platform, we have decided to use the same interface type between modules and the same data organization from the lowest level to the highest one, as detailed in Section 4. Moreover, the data manager we have developed provides the system with feedback channels going from high-level modules towards low-level modules to allow closed-loop configurations, even if these channels are not used by all systems.

A second property is the *separation between the algorithms and the a priori knowledge they use*. Using a priori knowledge is not new but keeping it separate from algorithms enables reusability. Complex systems performing activity monitoring use a huge amount of knowledge of different types. Knowledge is often application dependent and, for the same application, camera dependent, so it should never be embedded into the algorithms. In our case, we have decided to use 3D descriptions of the observed empty scenes as well as predefined scenarios as a priori knowledge available to the system. The 3D descriptions change when the observed scenes change and their separation from the algorithms enables to adapt the system to different video cameras. The predefined scenarios change when the application changes, but thanks to this separation, we can reuse the same algorithm for a different system without modifying it.

A third property is *automatic evaluation*, whose goal is to enable to evaluate the results of the different AMSs built with the platform. This property is important after the integration of a new algorithm or the modification of an existing one. When addressing a new application, it is normal to face new problems which require to handle new situations and to use more a priori knowledge (or to refine the already existing knowledge). The development and the integration into the platform of such new algorithms is made possible by the modularity and the flexibility of the architecture (first property). Thus, the difficulty is to insure that the new algorithm keeps the quality of results previously obtained by the AMS dedicated to other applications. A solution can be the development of an automatic evaluation framework based on ground-truth data, which is able to evaluate the performances of a set of AMSs on a wide set of predefined sequences. Thanks to that, it is possible to evaluate the impact of a new algorithm on the platform, insuring that it globally increases the quality of the results. Moreover, a framework of this type enables to apply statistical learning methods for parameters tuning, useful to find the best parameter set for a given application.

Finally, a development methodology to fulfill the last two properties consists in the *interaction between end-users and developers* (the end users are, e.g., metro security or bank agency operators). This interaction is useful because end-users provide the a priori knowledge (the predefined scenario models) used by the system (second property) and the scenario-level ground-truth used to perform the automatic evaluation (third property). There are also three other important reasons for developers to interact with end-users. The first reason is that end-users, helped by system developers, can find out which are the interesting activities to monitor and how to describe them precisely. The importance of this approach is to avoid to have a system which does not meet user's needs. The second reason is the necessity to often ask professional actors to act a set of scenes showing either normal activities or the activities to monitor. These video sequences are necessary during the development of the system to tune and to test algorithms. Actors are needed because there are often too few recorded sequences showing abnormal activities. Only end-users can explain to actors (1) how to act in a realistic way, (2) which are the activities to monitor, and (3) how to describe them precisely. The third reason is that end-users can perform, at the end of the development, an assessment of the system, measuring its efficiency and evaluating its utility.

In our case, we have been working closely with end-users of different application domains. For example, we have built with VSIP three AMSs which have been validated by end-users: an activity monitoring system in metro stations, a bank agency monitoring system, and a lock chamber access control system for buildings security. These applications present some characteristics which make them interesting for research purposes: the observed scenes vary from large open spaces (like metro halls) to small and closed spaces (corridors and lock chambers); cameras can have both nonoverlapping (like in metro stations and lock chambers systems) and overlapping fields of view (metro stations and bank agencies); humans can interact with the equipment (like ticket vending machines or access control barriers, bank safes and lock chambers doors) either in simple ways (*open/close*) or in more complex ones (as the interaction occurring during *vandalism-against-equipment*

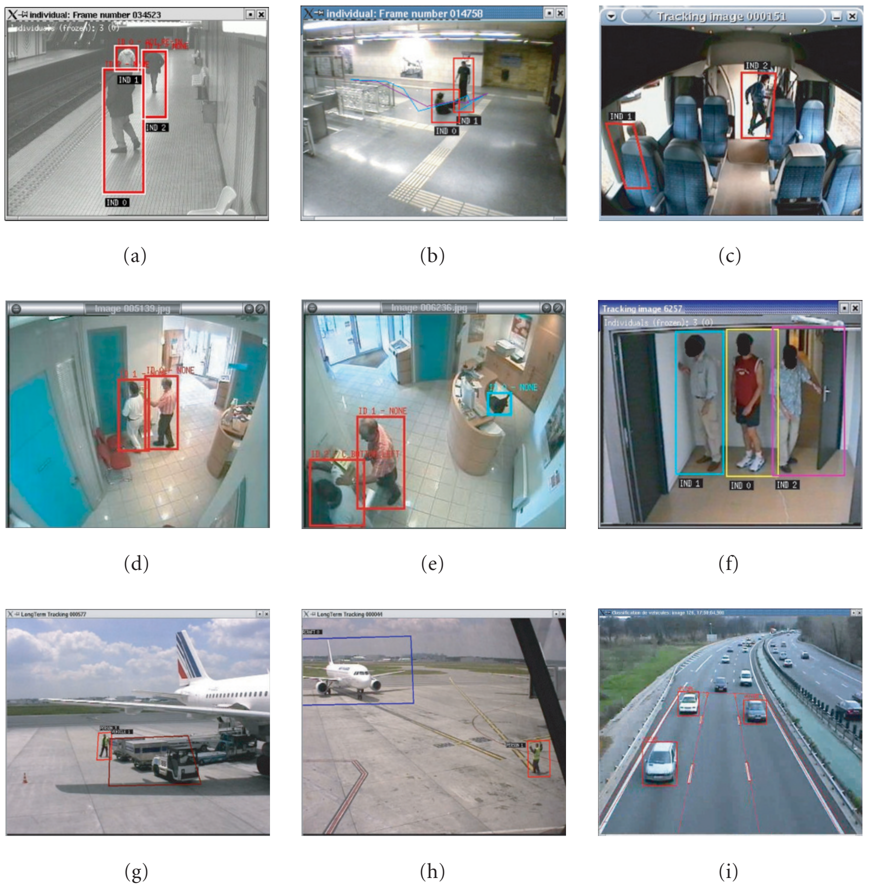


Figure 1: Shows six activity monitoring systems (AMS) derived from VSIP to handle different applications. (a) illustrates a metro monitoring application system running on black and white cameras of the YZER station in Brussels. (b) illustrates the same system but analyzing images from a color surveillance camera of the SAGRADA FAMILIA station in Barcelona. (c) illustrates a system for unruly behaviors detection inside trains. Images (d) and (e), taken with 2 synchronized cameras with overlapping field of view working in a cooperative way, illustrate a bank agency monitoring system detecting an abnormal "bank attack" scenario. (f) illustrates a single-camera system for a lock chamber access control application for building entrances. Images (g) and (h) illustrate an application for aprons monitoring on airports; this application combines a total of 8 surveillance cameras with overlapped fields of view. Finally, (i) illustrates an highway traffic monitoring application.

or *jumping-over-the-barrier* scenarios). All these AMSs have been validated and an end-user assessment has been done or it is scheduled for the beginning of 2005.

We are currently building with VSIP three other applications. All these applications are illustrated on Figure 1. A first application is apron monitoring on an airport[1] where vehicles of various types are evolving in a cluttered scene. The dedicated system has been able to successfully detect at the same time vehicles and people getting in and out on several videos lasting twenty minutes. A second application consists in detecting abnormal behaviors inside moving trains. The dedicated system is able to handle situations in which people are partially occluded by the train equipment like seats. A third application is traffic monitoring on highway; the dedicated system has been built in few weeks to show the adaptability of the platform. These systems are currently under development and validation and end-user assessment will be done in the near future.

The next section presents a state of the art of activity monitoring research done during the last ten years. We then give an overview of the global architecture of VSIP in Section 3. Section 4 presents how we addressed the first property, the modularity and the flexibility of the architecture. Section 5 describes how we have managed to obtain a separation between the algorithms and the a priori knowledge. In Section 6, we address the third property, the automatic evaluation of the results. Section 7 addresses the platform development methodology based on the interaction with end-users. Then we present in Section 8 the validation and the end-user assessment of a system built with the VSIP platform and applied to metro stations. In Section 9, we give the validation results for two other systems, applied to bank agencies and building entrances. Finally in Section 10, we give some concluding remarks and we present ongoing works.

[1]In the framework of AVITRACK European Project, see [5, 6].

2. STATE OF THE ART

Video understanding is now a mature scientific domain which has started in the eighties. Early research on video understanding concentrated on vehicle tracking, because vehicle shapes are relatively easy to model and to detect in videos. These works included the monitoring of vehicles in roundabouts for traffic control (ESPRIT VIEWS—[7]). 3D vehicle modeling [8] was later extended to include models that can be distorted and which are parameterizable and to include appearance modeling to improve robustness.

The last decade has witnessed a more practical and user-centered development of vision and cognitive vision researches. The main achievement has been the development of activity monitoring, usually focusing on the low-level video processing aspect and on people tracking. People tracking is more difficult than vehicle tracking because the human body is nonrigid and people motions have more degrees of freedom and are less predictable than vehicle motions. At present, real-time tracking of people is mainly achieved using appearance-based models. For example, Haritaogou et al. [9] use shape analysis and tracking to locate people and their parts (head, hands, feet, torso) in image sequences. Oliver et al. [10] use Bayesian analysis to identify human interactions using trajectories obtained from a monocular camera. Other examples include the Leeds people tracker [11]. The Leeds people tracker was combined with the Reading vehicle tracker to produce a single 3D integrated tracker for pedestrians and vehicles in the same scene. The visual surveillance and activity monitoring (VSAM) project (from 1997 to 2000) [12] involved twelve research laboratories in the implementation of systems to segment and track people and vehicles in image sequences, locate them in a 3D model of the scene environment using prior camera calibration, and visualize them in a plan view dynamic scene. More recently, the VACE program (video analysis and contents exploitation [13]) and the homeland security ARPA program [14] organize research in USA on video understanding. For VACE, the goal is to recognize events of interest from any type of video sources.

In general, video understanding systems rely on careful camera positioning and a dense camera network. The multicamera tracking of Javed et al. [15] uses multiple views to rebuild the trajectory of people between nonoverlapping cameras, linking the different fields of view being observed. Routes followed by pedestrians through the scene are learnt by observing a large number of motion trajectories and allow to construct a geometric and probabilistic trajectory model for long-term prediction.

Scene modeling is used to increase the reliability of tracking and behavior interpretation. For example, people in the field of view are likely to be on the ground plane, and moving vehicles are likely to be on a road rather than on the pavement.

A new trend in video understanding systems is to use evaluation and program supervision techniques to improve robustness. The creation of PETS [16] enforces the idea that we need evaluation techniques to assess the reliability of existing tracking algorithms. But these workshops are mostly intended to test various algorithms on the same video inputs. Algorithms comparison is mostly qualitative, and quantitative comparison based on precise criteria is missing. Nevertheless, we can mention an interesting theoretical work on performance evaluation which can be found in [17]. It first discusses the importance of testing algorithms on real video sequences, for instance, to test outdoor sequences with various weather conditions. Second, it presents pros and cons of ground-truth techniques and their alternatives. However the repair and tuning stage of these video understanding systems is manually realized. Only few works [18] try to optimize the performance of these systems.

Moreover, few of these systems are able to perform complex reasoning (i.e., spatio-temporal reasoning) and to understand all the interactions between people in real-world applications. In the video understanding domain, two main approaches are used to recognize temporal events from video either based on a probabilistic/neural network or based on a symbolic network. For the computer vision community, a natural approach consists in using a probabilistic/neural network. The nodes of this network correspond usually to events that are recognized at a given instant with a computed probability. For example, Hongeng et al. [19] proposed an event recognition method that uses concurrent Bayesian threads to estimate the likelihood of potential events. For the artificial intelligence community, a natural way to recognize an event is to use a symbolic network whose nodes correspond usually to the symbolic recognition of events. For example, some artificial intelligence researchers used a declarative representation of events defined as a set of spatio-temporal and logical constraints. Some of them used a traditional constraints resolution or temporal constraints propagation [20] techniques to recognize events.

In spite of all these achievements, no activity monitoring systems can be said to be robust or generic enough to be used in a real-world application. An adapted design, development, and evaluation methodology is still needed to achieve a generic intelligent video understanding platform.

This paper proposes four good properties that an activity monitoring platform should have to enable its reusability for different applications. As a concrete expression of these properties, we present a complete platform including human and vehicle detection and tracking, scene modeling, spatio-temporal reasoning capabilities. To underline the reusability of the platform allowed by these four properties, we present validation and end-users assessment results for three systems built with the platform.

3. PLATFORM OVERVIEW

To demonstrate the feasibility of our approach and to illustrate with concrete examples the application of the proposed properties, we present an activity monitoring platform, named VSIP, whose global structure is shown in Figure 2. We use this platform to build activity monitoring systems for specific applications.

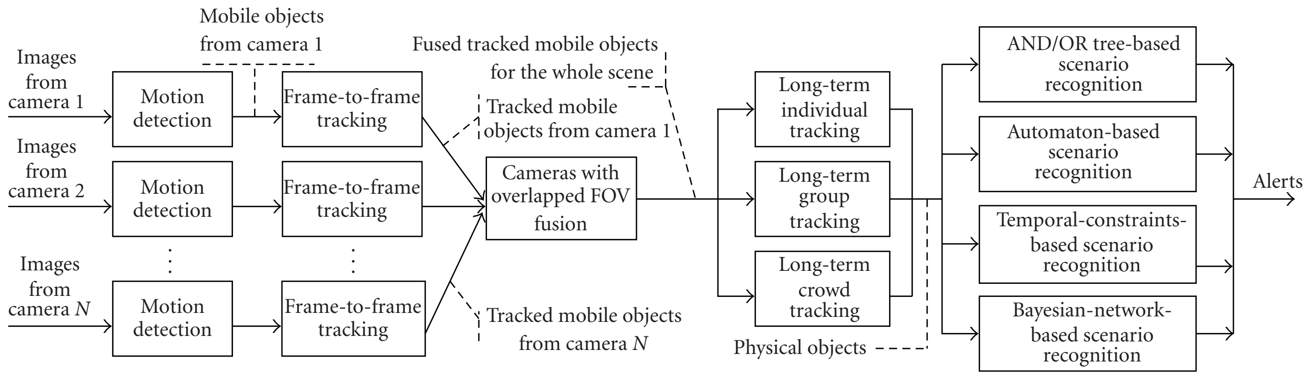

Mobile objects from camera 1
Images from camera 1
Motion detection
Frame-to-frame tracking
Images from camera 2
Motion detection
Frame-to-frame tracking
Images from camera N
Motion detection
Frame-to-frame tracking
Fused tracked mobile objects for the whole scene
Tracked mobile objects from camera 1
Cameras with overlapped FOV fusion
Tracked mobile objects from camera N
Long-term individual tracking
Long-term group tracking
Long-term crowd tracking
Physical objects
AND/OR tree-based scenario recognition
Automaton-based scenario recognition
Temporal-constraints-based scenario recognition
Bayesian-network-based scenario recognition
Alerts

Figure 2: Shows the global structure of the activity monitoring platform. First, a motion detection step followed by a frame-to-frame tracking is made for each camera. Then the tracked mobile objects (objects from i-th camera) coming from different cameras with overlapping fields of view are fused into a unique representation for the whole scene. Depending on the chosen application, a combination of one or more of the available trackers (individuals, groups, and crowd tracker) is used. The results are passed to the behavior recognition algorithms, which combine one or more of the following algorithms, depending on the scenarios to recognize automaton-based, Bayesian-network-based, AND/OR tree-based, and temporal-constraints-based recognition algorithms. Finally, the system generates the alerts corresponding to the predefined recognized scenarios.

The input images are color or black and white, digitized with a variable frame rate (typically between 4 and 25 fps).

The *segmentation* algorithm detects the moving regions by subtracting the current image from the reference image (a background image built with images taken under different lighting conditions). These moving regions, associated with a set of 2D features like density or position are called *blobs*. A *noise tracking* algorithm allows to discriminate blobs between real moving regions and regions of persistent change in the image (like a new poster on the wall or a newspaper on the table). Following the type of the application, a *door detection* algorithm, which allows to handle the opening/closing of doors which have been specified in the 3D description of the scene, can be activated. This algorithm removes the moving pixels corresponding to a door being opened or closed. A set of 3D features like 3D position, width, and height, are computed for each blob. Then the blobs are classified into several predefined classes (like, e.g., person, group, noise, car, truck, aircraft, unknown, etc.) by the *classification algorithm*.

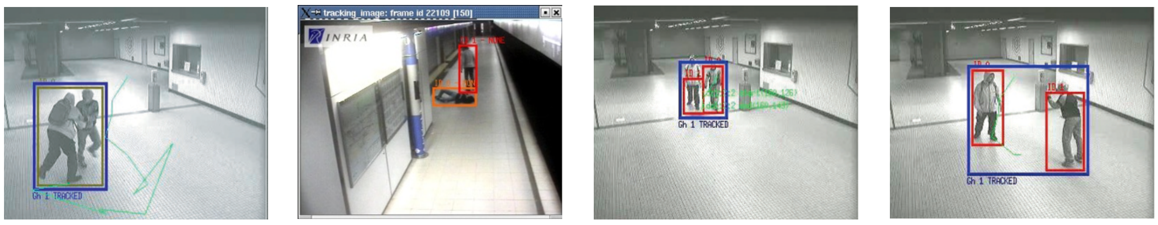
The blobs with their associated class and a set of 3D features are called *mobile objects*. A *split and merge* algorithm corrects some detection errors like a person separated into two different mobile objects. A *3D repositioning* algorithm corrects the 3D position of the mobile objects classified as person that have been located at a wrong place (such as outside the boundary of the observed scene or behind a wall). This happens when the bottom part of the person is not correctly detected (e.g., the legs can be occluded by an object or badly segmented). If useful for the application, a *chair management* algorithm can be activated, which helps differentiating a mobile object corresponding to a chair from a mobile object corresponding to a person. A *background-updating* algorithm uses the discrimination between real mobile objects and regions of persistent change in the image (discrimination done by the noise tracking algorithm) to update the reference image by integrating the environment changes [21].

The set of the previously described algorithms is generally called "motion detection module." The output of this module is, for each frame, the list of the mobile objects (with their 3D features and their class).

The motion detection module is followed by the *frame-to-frame tracking module*. The goal of this module is to link from frame to frame all mobile objects computed by the motion detection module. The output of the frame-to-frame tracking module is a graph containing the detected mobile objects updated over time and a set of links between blobs detected at time t and blobs at time $t-1$. A mobile object with temporal links towards mobile objects of the previous frame is called a *tracked mobile object*. This graph provides all the possible trajectories of a mobile object and it constitutes the input for the following long-term tracking module.

The lists of mobile objects coming from different cameras with overlapped fields of view are then fused together by a *fusion algorithm* to give a unique representation of the mobile objects. The algorithm uses combination matrices (combining several compatibility criteria) to establish the good association between the different views of a same mobile object observed by different cameras. A mobile object detected by a camera may be fused with one or more mobile objects seen by other cameras, or can be simply kept alone or destroyed if classified as noise. After fusion, the resulting *fused tracked mobile objects* combine all the temporal links of mobile objects which have been fused together. The 3D features of the resulting fused objects are the weighted mean of the 3D features of the original mobile objects. Weights are computed in function of the distances of the original mobile objects from the corresponding camera. In this way, the resulting 3D features are more accurate than the original.

Depending on the scenarios to recognize, one or more *long-term trackers* can be used. All of them rely on the same idea. They first compute a set of paths representing the possible trajectories of the *physical objects* to track (isolated



(a) (b) (c) (d)

Figure 3: Shows 3 examples of visual invariants used to recognize a *fighting* scenario: (a) an erratic trajectory (shown in green on the image) of the group of fighters; (b) one of the fighters laying on the ground; (c) and (d) important relative dynamics inside the group (measured as distance variation over time of the people composing the group).

individuals, groups of people, crowd, cars, trucks, airplanes, etc.). Then they track the physical objects with a predefined delay T to compare the evolution of the different paths. The trackers choose, at each frame, the best path to update the physical object characteristics [22].

The fused physical objects are then processed by the *behavior recognition algorithms* to recognize the predefined scenarios. Depending on the type of scenarios to recognize, different behavior recognition algorithms (based on automatons, Bayesian networks, AND/OR trees and temporal constraints) can be used. These algorithms use the concepts of "state," "event," and "scenario." A state is a spatio-temporal property valid at a given instant or stable on a time interval. An event is a change of state. A scenario is any combination of states and events. The scenarios corresponding to a sequence of events are represented as automatons where events correspond to a transition (a change of state) within the automaton. When the correct chain of events occurs, the scenario is said to be recognized. For scenarios dealing with uncertainty, Bayesian networks can be used. For scenarios with a large variety of visual invariants (e.g., *fighting*), AND/OR trees can be used [23]. Visual invariants are visual features which characterize a given scenario independently of the scene and of used algorithm. For example, for a *fighting* scenario, some visual invariants are an erratic trajectory of the group of fighters, or one person lying down on the ground, or important relative dynamics inside the group as shown in Figure 3.

For scenarios with multiple physical objects involved in complex temporal relationships, we use a recognition algorithm based on a constraint network whose nodes correspond to subscenarios and whose edges correspond to temporal constraints. Temporal constraints are propagated inside the network to avoid an exponential combination of the recognized subscenarios. The scenarios are modeled in terms of "physical objects" (people or static scene objects or zones of interest, etc.), "components" (which can be primitive states, composite states, primitive events, or composite events) and "constraints" between the physical objects and/or the components (constraints can be temporal, spatial, or logical). For each frame, scenarios are recognized incrementally, starting from the simplest ones (e.g., "an individual is close_to") up to the more complex. The temporal constraints are checked at each frame. This algorithm uses a declarative language to specify scenarios (see, e.g., Figure 4 and [20]). An ontology for video events (see [24]) has been developed in the framework of ARDA workshop on video event.

```
composite-event(vandalism_against_ticket_machine_one_man,
  physical-objects((p: Person), (eq1: Ticket_Vending_Machine),
                   (z1: Ticket_Vending_Machine_Zone) )
  components( (c1: primitive-event Enters_zone(p, z1))
              (c2: primitive-event Move_close_to(p, eq1))
              (c3: composite-event Stays_at(p, eq1))
              (c4: primitive-event Goes_away_from(p, eq1))
              (c5: primitive-event Move_close_to(p, eq1))
              (c6: composite-event Stays_at(p, eq1)))
  constraints( (c1; c2; c3; c4; c5; c6) ) ) // Sequence
```

Figure 4: The description of a vandalism scenario using our declarative language. It describes the degradation of a piece of equipment by an individual: first, the person moves close to the equipment. He/she stays close to the equipment then he/she moves away from the equipment to avoid being seen. He/she then goes back close to the equipment and so forth. The terms corresponding to the video event ontology are in bold.

The whole processing chain can be processed for two cameras in real time on one of-the-shell PC.

4. MODULARITY AND FLEXIBILITY OF THE ARCHITECTURE

In software engineering, a classical property for a platform is its modularity and flexibility. We agree that this property has to be considered during the whole development of an automatic interpretation platform in order to insure the reusability of the algorithms.

For a platform, *modularity* is the property of being composed by subunits (modules) each of them achieving a particular and well-defined task. Modularity enables to create systems that can be adapted to various applications (metro surveillance, bank agency surveillance, people counting, etc.) in various environments (e.g., different metro stations). Indeed, systems can be composed combining carefully the modules corresponding to the particular requirements of the applications. Nevertheless, a problem is still open: the management of the data exchanges between modules. Our solution to this issue is based on the notion of *shared data* *manager*. A shared data manager is a data structure where modules read and write input/output data. As we have seen, a *module* represents a platform functionality: in our case, for example, video acquisition functionality, segmentation functionality, or frame-to-frame tracking. *Input/output* are all the data exchanged between modules. For example, the acquisition module does not take any input data and outputs an image. The shared data manager can be thought as a module which performs "data management and distribution" task, following the modularity philosophy. The shared data manager manages the way data are exchanged between modules. A module is only connected to the shared data manager. To put some data in the shared data manager, the module calls the appropriate method of the shared data manager. The module is not aware of how and when data will be used. Separating data management from module functionality allows, for instance, an application to be distributed on different machines changing only shared data-manager implementation. This organization enables to provide a homogeneous vision of the platform. Thus, building an application is a systematic process that consists in creating a shared data manager and selecting one or several modules to connect with it. If an additional development is needed (e.g., because addressing for the first time an outdoor application), it is limited to the new encountered problem (e.g., "illumination changes due to weather conditions") without affecting the other modules of the platform. Thanks to the shared data manager, we have the possibility to reuse the same algorithms with different architectures (e.g., distributed or multithreaded architectures or code embedded into cameras). To develop an activity monitoring system on a distributed architecture, a shared data manager has to be created on each computer. The role of the data managers is to automatically maintain and update the shared data. The distribution has no other effects on the platform. Moreover, data types which are handled by the platform have precise and clear definitions; a piece of information is unique and has the same meaning over the whole platform. For example, a *blob* is a connected set of pixels detected by the segmentation (they can be moving or stationary) with an associated set of 2D descriptors like size, position, and density. A *mobile object* is defined as a set of blobs "merged together" because it globally corresponds to the perception of a physical object on the image. It is characterized by a class (like person or airplane) and by a set of 3D features (position and size). A *tracked mobile object* is a mobile object with (potentially) one or more temporal links to mobile object(s) of the previous frame.

We call *flexibility* the property of having a set of tunable parameters. This property implies the possibility to configure algorithms and to define different scenarios without changing source code. To fulfill this property, we have decided to make all the internal parameters of every module tunable. To do that, parameter values are defined in separate files outside the code (i.e., outsourcing of parameters) using a description formalism as proposed in [25]. These files are handled by the shared data manager as regular input/output data. These parameters can be changed during processing enabling parameters optimization as explained in Section 6.

Thanks to the shared data manager and the outsourcing of parameters, we achieved to have a platform architecture which fulfills modularity and flexibility properties.

5. SEPARATION BETWEEN ALGORITHMS AND A PRIORI KNOWLEDGE

This section focuses on the second property, separation between algorithm and a priori knowledge. VSIP platform uses a large number of a priori knowledge for two main reasons. First, it is often useful to design specific routines using additional a priori knowledge for correcting imprecise and uncertain data. For instance, we correct 3D wrong positions due to partial occlusion in a cluttered environment by adding a correcting step which uses the information coming from the 3D scene description (position and dimensions of context objects which can occlude people and the type of occlusion they can cause). Thanks to this approach, we manage to recognize the *jumping over the validation barrier* scenario which needs to compute precisely the 3D position of people behind the barrier. Second, by providing an algorithm with knowledge, it is possible to reduce the processing time. For example, on a sidewalk where only pedestrian can be observed, we will not try to classify mobile objects as vehicles.

The a priori knowledge is composed of two different types of information: *image acquisition knowledge* and *a priori models*. The first one is composed by the following information.

(i) *Camera calibration parameters* are used to compute the real position in the 3D scene of the 2D objects detected on the image.
(ii) *Hardware information* contains the features of each equipment (frame rate of the camera, network configuration, data compression rate, etc.).
(iii) *Reference images* are a set of predefined images representing the appearance (night or day) of the empty scene (scene without mobile objects).

Models are of 2 types.

(i) *3D scene model* contains the 3D geometry of the scene observed by the camera and the objects present in the scene. These physical objects are of two types: contextual objects found in the empty scene (trash cans, benches, stamping machines, zone of interest, etc.) and mobile objects which can evolve in the scene (persons, group of people, airplane, train, etc.). A semantic information is associated to each object, like "occluding" for an object which can occlude people and "on top" or "on bottom" to specify the type of the occlusion, or "in/out" for a zone corresponding to an entry or an exit.
(ii) *Scenario models library*. This information is independent of the camera. It consists in a set of predefined scenarios to recognize. These scenarios are described using a special declarative user-oriented language (see Section 7).

If the use of a priori knowledge enables to better and more efficiently solve the interpretation problems, only the separation between knowledge and algorithms enables algorithms to be independent of the application and to be reusable in other situations. All a priori knowledge in VSIP is stored in specific configuration files independently of the code. For example, all cameras observing the same scene are processed by computers having the same 3D scene model. Also, applying an AMS to a new scene requires only to change the 3D scene model. We have also proposed an adapted formalism to describe each type of knowledge. For example, the scenario models are described using a special declarative user-oriented language, as shown in Section 7.

Because of this knowledge organization, we have managed to separate a priori knowledge from the algorithms.

6. AUTOMATIC EVALUATION

When facing new applications, it is often necessary to add to the platform new algorithms able to handle situations encountered for the first time. For example, for bank agency monitoring application [26], we have developed a *chair management* module, which has been integrated to VSIP and is currently used by an AMS for indoor applications dealing with chairs.

Our experience in building AMS has shown that usually to handle real-world diversity, a reusable platform should contain a combination of simple algorithms dedicated to each type of situations rather than containing a very sophisticated algorithm handling all situations. Robustness in activity monitoring is then achieved when many algorithms can be easily combined in the same platform.

Once validated on a specific application, these new algorithms have to be integrated to the platform. To preserve the reusability of the platform and its robustness with respect to the whole set of applications, two problems arise.

(i) It is necessary to insure that the new algorithms do not lower the quality of the results obtained by other AMS built with the platform. In other words, it is important to be able to measure the impact of new algorithms on the quality of the results obtained by all AMS on a predefined set of sequences representative of the applications.

(ii) We have to be able to find the good set of parameters which guarantees, for each new application, the best quality of results. Sometimes it happens that after the introduction of a new algorithm, the initial set of parameters does not give satisfactory results anymore. Thus we have to be able to recompute them for each application (one set for each application) in an automatic way.

To find a solution for both problems we have developed an evaluation framework. This framework is based upon the following.

```
<annotation_activity id = “31” priority = “2”
                     class = “alarm”
                     sub_class = “jumping_over_barrier”>
   <list_video_frames best_camera_area = “HALL01”
                      best_camera_id = “C11”>
      <video_frame id = “42535” camera_area = “HALL01”
                   camera_id = “C11”>
      </video_frame>
   </list_video_frames>
   <time start_time_hour = “2” start_time_min = “21”
         start_time_sec = “46” start_time_ms = “799”>
   </time>
   <list_activity_physical_objects>
      <physical_object id = “104” role = “source”>
      </physical_object>
      <physical_object id = “16” role = “stat_reference”>
      </physical_object>
   </list_activity_physical_objects>
</annotation_activity>
```

Figure 5: XML annotation of a video: the recognized scenario is “jumping over the barrier.” It implies two physical objects, one person (ID 104) and one validation barrier (ID 16). The scenario is best viewed on camera C11.

(i) A set of ground-truth sequences for each given application. With the term “ground-truth” we describe a set of sequences for which a human operator has given the “best results” (truth) that a system would have given if it had worked perfectly. Ground-truth can be specified at each different step of the platform: motion detection, frame-to-frame tracking, fusion, long-term tracking, and scenario recognition. For example, at motion detection level, ground-truth means to draw for each image a bounding box surrounding each mobile object evolving in the scene, labeling it with its type. At tracking levels, it means to correctly track by hand the mobile objects even when the mobile object is partially or totally occluded. Finally, at scenario level, it means to recognize by inspection the scenarios depicted by the image sequences.

(ii) A clear definition of the high-level data types used by the platform as an interface between modules (details in Section 4), and an XML format for each of these data types allowing their manipulation even outside VSIP. For example, Figure 5 shows the XML format used to represent the annotation data types which is generated when a scenario is recognized.

When a new algorithm is added to the platform, the set of ground-truth sequences are run automatically by each AMS. The evaluation results are compared with those obtained before the integration of the new algorithm. In case of lower-quality results, a first possibility is to recompute the set of parameters separately for each AMS. The framework allows to apply a statistical learning technique for parameter tuning. For all ground-truth sequences, the algorithm is run with a modified set of parameters. If the results improve, the parameters are validated, if not, they are modified using

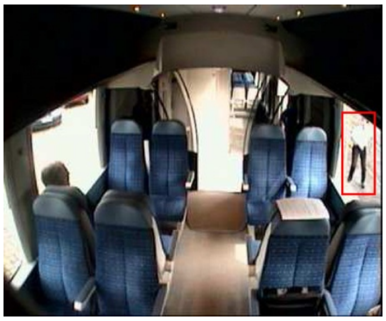


FIGURE 6: Shows a person (on the right) who is seen through a window in the wall. In this case, the repositioning algorithm has to take into account the particular situation and to avoid repositioning the person inside the train when it is outside.

an algorithm that explores heuristically the N-dimensional space of parameters (N being the number of parameters).

If the improved set of parameters still gives worse results than the one used before the introduction of the new algorithm, then this algorithm is said to be not generic enough to be used for all applications. The next step is to understand precisely why the new algorithm fails and under which hypotheses it can be used.

For example, the repositioning algorithms (developed for the bank agency monitoring system) were designed to correct the position of individuals when they are wrongly detected behind a wall (in situations where legs are not detected). The algorithm gives incorrect results in train surveillance system because it wrongly correct the position of people who are behind walls containing a window (see Figure 6). Thus two algorithms have to be developed to handle scenes containing both walls with or without windows.

Today, this choice of algorithm is made manually for VSIP when building an AMS. We are currently working on extending the evaluation framework to determine automatically in which situations an algorithm can be used.

7. INTERACTION BETWEEN THE END-USERS AND THE DEVELOPERS

As we have seen in Section 1, the interaction between the end-users and the developers is a development methodology useful to fulfill the separation between the platform and the a priori knowledge (as described in Section 5) and to perform automatic evaluation of the results (as described in Section 6).

Moreover, system design is often driven by technical limitations rather than user requirements. The proposed approach consists in integrating not only user needs but also user knowledge in the development process in order to address real-life problems. This integration has three main interests. The first one is to provide a system well adapted to end-users needs. The second one is to provide a framework to assess the usefulness of the system on video sequences representing real-life situations. The last interest is the possibility to improve efficiency and robustness of the system by using user knowledge. For example, detecting a pickpocket theft is impossible but with the help of users, we found some typical precursor events (e.g., blocking a passenger in an exit zone) which are easier to recognize. Collaboration with users is an incremental process during which different types of knowledge are taken into account. The collaboration is composed of three phases.

In the first phase, end-users motivations are collected to define goals and their priorities. In the case of metro stations, three goals were specified by the users: traffic free flow, passenger and employee security, and equipment protection. Based on these goals and on the importance given to each situation (frequency, gravity in terms of physical loss, and costs), several scenarios are chosen. In Barcelona (Spain) metro, for example, one of the major issues is fraud. The Barcelona metro stations are not equipped with efficient devices to control platform access: there are only simple barriers easy to stride. Thus, metro managers decided that it would be interesting that the video surveillance system automatically detects people jumping over the barriers. In Brussels (Belgium) metro, fraud is not relevant because there is no validation barrier. However, access blocking is a real problem for different reasons pointed out by metro managers. The first one is the degradation of the traffic free flow. The second one concerns accidents that may occur when one or several individuals are blocking the escalators. In this case, people may pack and fall. The last one is less obvious and concerns pickpocket activities. Actually, while a few individuals are blocking a passenger in an exit, an accomplice can take advantage of the situation to rob the passenger.

In a second phase, users which have a visual or ground experience (e.g., video surveillance operators, security agents) specify precisely the course of each scenario: how the individuals present in the scene behave before, during, and after the event. Users may also provide visual invariants which are characteristics of each behavior wherever it occurs as shown in Figure 3. Based on the detailed description of the course of each scenario and on visual invariants, a set of video sequences representing abnormal and closely related but normal situations is recorded with the help of actors if necessary. Using an XML language, each video is then annotated by end-users with the scenarios they represent.

A video annotation describes three pieces of information as shown in Figure 5: on which camera/frame we can see the scenario (tagged "video frame"), when the scenario occurs (tagged "time"), and who is involved in the scenario (tagged "physical object"). As the formalism is the same for both information (end-user description of scenarios models and VSIP output), we are able first to make sure that recognized scenarios match user descriptions and second to automatically evaluate system efficiency by comparing user annotations and system results.

The third phase corresponds to scenario modeling and recognition. It is a sensitive step because scenario models must be understood on the same way by the users and the system. Actually, users may want to easily modify or extend scenarios. The usual approach is to hard code each scenario in the system. For example, given that in any application we are interested in recognizing a limited number of scenarios, it is often easier for developers to hard code the scenario recognition routines (automaton, AND/OR trees, etc.) instead of developing a more reusable and complex algorithm able to generate automatically the routines corresponding to a textual description of a scenario. But this approach is not satisfactory because it heavily limits the reusability of the developed routines and prevents nondevelopers users from being able to modify or extend by themselves the set of scenarios that can be recognized by the AMS. Our proposed approach introduces a new scenario representation language based on a video event ontology (see [24]). An ontology is the set of all the concepts and the relations between concepts shared by the community in a given domain. The ontology first facilitates the communication between the domain experts (end-users) and the developers. The ontology makes the video understanding systems user centered and enables the end-users to fully understand the terms used to describe scenarios models without being concerned by the low-level processing of the system. Moreover, the ontology is useful to evaluate the AMS and to understand exactly what type of events a particular system can recognize. This ontology is also useful for developers of AMS to share and reuse scenario models dedicated to the recognition of a specific event.

This video event ontology has been built in the framework of ARDA workshops. It insures that the terms are shared by several laboratories specialized in video analysis. Events are decomposed in different abstract levels and in a hierarchical structure with the aim to make the model generic and applicable to a wide range of applications. There are two main types of concepts to be represented: physical objects of the observed scene and video events occurring in the scene. A physical object can be a contextual object (e.g., a desk, a door) or a mobile object detected by a vision routine (e.g., a person, a car). A video event can be a primitive state, a composite state, a primitive event, or a composite event. Primitive states are atoms used to build other concepts of the knowledge base of an AMS. A composed concept (i.e., a composite state or a composite event) is represented by a combination of its subconcepts (called components) and an optional set of events that cannot occur during the recognition of this concept.

The language based on this ontology enables to describe in an intuitive and declarative way all the knowledge necessary to recognize scenarios (see Figure 4).

Furthermore, to improve the incremental development process, we develop a visualization tool that generates 3D animations and video sequences from scenario models. These sequences are useful both for users and for developers. Users can visually check that the scenario model corresponds to the scenarios they want to specify. Developers have a tool to generate test sequences for debugging their code. The use of the video event ontology of an adapted language for scenario modeling and of the visualization tool has made this collaboration efficient by keeping the knowledge coherent and accessible to all participants (end-users and developers).

8. END-USER ASSESSMENT AND VALIDATION FOR METRO STATIONS MONITORING APPLICATION

Using the VSIP platform, we have built several AMS for different applications, as described in Section 1 and illustrated in Figure 1: a bank agency surveillance application, a metro activity monitoring system, a lock chambers access control application, and so forth. In this section, we present the results of the end-user assessment and the technical validation of the metro activity monitoring system installed in Sagrada Familia station of the Barcelona metro at the end of the European ADVISOR Project (March 2003). In Section 9, we present the corresponding validation results for two other applications (bank agency monitoring and lock chambers access control) for which the end-users assessment is scheduled for the beginning of 2005.

The AMS built for metro monitoring was the activity monitoring kernel of the final demonstrator of the ADVISOR Project. Besides the AMS, the demonstrator includes:

(i) a capture system which digitizes the images coming from live cameras and plays back recorded sequences;
(ii) a crowd monitoring system, delivering additional information about crowd (like direction of crowd motion flow);
(iii) an archive system, which records the input video sequences together with annotations describing the recognized scenario, if any. A second functionality of the archive is the possibility to act like a playback system allowing the easy search and retrieval of specified sequences and/or recognized scenarios;
(iv) a human-computer interface allowing the operators to visualize the results, to control system parameters, and to access the archive system.

The demonstrator has been presented to security operators from STIB (Brussels, Belgium) and TMB (Barcelona, Spain), two metro companies, followed by a tutorial explaining how to use it.

The end-user assessment and the technical validation were conducted using both live and recorded data. Four closed-circuit cameras at Sagrada Familia were connected to the AMS system, providing live data from the metro station. In addition, four prerecorded sequences were also fed into the system. These sequences are composed by the following.

(i) Scenes played by actors containing the various human behaviors to recognize. These sequences were intended to demonstrate the capability of the system to recognize predefined scenarios, such as *fighting*, that were unlikely to occur in live during the evaluation and the validation.
(ii) Normal scenes coming from recording made by security operators and showing normal behaviors. These sequences were intended to demonstrate the robustness of the system with respect to false alerts (i.e., alerts generated even if no predefined scenario is happening in the video).

8.1. End-user assessment

The end-user assessment consists of end-users (video surveillance operators) to establish how useful the system is. During the end-user assessment, the end-users were asked to use the system, as part of their regular surveillance task for a few hours a day during a week and evaluate its performance and usefulness. The results were documented by the completion of a comprehensive questionnaire that pointed out the following remarks.

The operators found that the AMS worked correctly and recognized with enough precision the predefined scenarios (*fighting*, *blocking*, *overcrowding*, *jumping over the barrier*, and *vandalism against equipment*).

The scenarios corresponded to the following situations.

(i) *Blocking* occurs when a group of at least 2 people is stopped in a predefined zone for at least 4 seconds and can potentially block the path of other people.
(ii) *Fighting* occurs when a group of people (at least 2 persons) is pushing, kicking, or grasping each other for at least 2 seconds.
(iii) *Overcrowding* occurs when the density of the people in an image is greater than a specified threshold.
(iv) *Jumping over the barrier* occurs when a person jumps over a specified ticket validation barrier.
(v) *Vandalism against equipment* occurs when an individual is damaging a piece of equipment in the image.

False alerts happened rarely and were not a problem because the operators had the time to acknowledge or to reject the generated alert. Operators pointed out that some efforts should be made on the system ergonomics, like easing the acknowledgment of an alert or automating the replay on the screen of the videos corresponding to a recognized scenario. They concluded stating that the AMS system was a real help to the surveillance task, and that should be used by metro companies to ease the security operator work.

8.2. Technical validation

The technical validation consists of technical people to determine whether the system recognizes the specified scenarios. A technical validation of the AMS system was performed at Sagrada Familia metro station. For the validation task, the system was tested using four input channels in parallel, the four channels being composed of three recorded sequences and one live input stream. The validation of the scenario recognition involved playing the sequences through the system and reporting the resulting alerts generated by the AMS. The sequences used for validation were annotated with ground-truth corresponding to the type and the occurrence time of the scenarios. The results obtained when the sequence was played through the system were then compared with the ground-truth. If the system generated the correct scenario recognition, then an estimate of the accuracy of the recognition was obtained. This was achieved by measuring the overlapping length between the observed scenario (ground-truth) and the occurrence of the scenario recognized by the AMS. So, for example, if the AMS reported a sequence as showing *fighting* for 45 seconds, when the ground-truth shows that 60 seconds of *fighting* occurred, then a score of 75% was awarded. The score also included true negative periods of the sequence, that is, if nothing happens and no alerts are generated, then the sequence is considered as correctly recognized. A delay of 5 seconds between the beginning of the scenario and the ground-truth is permitted in the measurement as this is the necessary delay for the scenario recognition algorithm to start the recognition of the scenarios.

The live channel was validated visually by the evaluators and was used mainly to check the rate of false alarms.

The results of the validation are presented and analyzed in Sections 8.2.1 and 8.2.2.

8.2.1. Ground-truth of the validation data

The sequences used in the validation of the AMS are composed of 29 different subsequences containing behaviors played by actors (corresponding to *fighting*, *blocking*, *overcrowding*, *Jumping-over-the-barrier,* and *vandalism-against-equipment* scenarios) and 3 long subsequences showing people with no behavior of interest. The 32 subsequences were duplicated several times at different places into the video test sequences, giving a total of 81 occurrences of scenarios supposed to be recognized by the AMS, and 22 occurrences of "normal" activities (supposed to generate no alerts).

We define as "ground-truth" the set of three information: the type, the starting time, and the duration of the scenarios recognized by a competent authority (technical people different from end-users and system developers). The ground-truth data is created by visual inspection. That is, the competent authority examines the sequences and decides which behaviors have occurred. This process is subjective: a scenario classified as *overcrowding* by an operator A could be considered as "normal" by a different operator B. This fact has no major consequences, because even in the case of end-users (that means people who have to use the system and judge its utility), the definition could change from a person to another one.

8.2.2. Scenario recognition validation results

Overall, the system was validated for over four hours using three recorded videos and one live camera, giving a total of more than 16 hours of validation. Table 1 details the results of the validation process.

The results of the validation for the *fighting* scenario show a success rate of 95% and the reports were found to be 61% accurate in the timing and duration of the alert report. Note that the accuracy is subject to the human interpretation of when fighting begins, which is not always clear. For example, two people might begin fighting by pushing each other, so it is unclear if fighting has begun at that point or when they actually start coming to blows.

The *blocking* scenario was detected giving detection rate of 78% with an average accuracy of 60%. One false *blocking* report was generated during the validation, when there was only one person standing by the exit barriers. At least two

Table 1: Shows the results of the technical validation of the AMS. 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 (that means what percentage of the duration of the shown behavior is "covered" by the generation of the corresponding alert by the system. This value is an average over all the scenario instances) (fifth column).

Scenario name	Number of behaviors	Number of recognized instances	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 o.t.b.	42	37	88%	100%	0
Overcrowding	7	7	100%	80%	0
Total	81	73	90%	85%	1

people are required to be blocking a predefined area to constitute a blocking event.

The *vandalism-against-equipment* scenario contains an actor repeatedly going to a piece of equipment and attempting to break it open. As people approach, he moves away from the equipment and returns to it later. The system recognizes this as one long act of vandalism rather than several individual acts and, therefore, has been scored as such. The main problem of this scenario was not to loose the tracks of the people when they cross other people during the scenario. This scenario gives a success rate of 100% with an accuracy of 71%.

The *jumping-over-the-barrier* (o.t.b.) scenario gives a success rate of 88%. The main difficulty of this scenario was to handle occultation and the ability to correctly compute the position of people relative to the validation machine (in front of/behind).

The *overcrowding* scenario shows a success rate of 100%, with an overall accuracy of 80%. The ground-truth of an *overcrowding* alert is also somewhat subjective since it is not exactly obvious at which point the scene becomes overcrowded.

The AMS was provided with a live feed from the Sagrada Familia station. The camera was situated in the main hall and overlooked the escalator from one of the platforms. Therefore, during busy periods, a large number of people disembark from the train, go up by the escalator, and enter the field of view of the camera. The relatively high density of people caused the AMS system to trigger an *overcrowding* alert. This is demonstrated by the fact that many such alerts were triggered on the busy Friday afternoon, whereas only two were generated on the much quieter Saturday morning. Thus, the high number of overcrowding alerts suggests that it would be interesting to synchronize the *overcrowding* scenario detection with the train arrival, to avoid the generation of an alert if the crowd is only disembarking from the train. Therefore, the *overcrowding* alerts have been scored as being correct because they were generated by a relatively high density of people emerging from the escalator after getting off a train. No other behaviors—except the *blocking* false alert detailed previously—were observed during the validation on this live channel.

Both validation and assessment scored the monitoring system as satisfactory. The next step is to test its performances and usability on larger camera networks and during longer periods of time.

9. SOME OTHER VALIDATION RESULTS

9.1. Bank agency monitoring system

As for the previous application, many discussions with domain experts have been needed in order to define scenarios, corresponding to interesting human behaviors, which have to be recognized in bank agencies. A bank scenario can be modeled in two parts: the attack precursor part (i.e., the robber approach) and the attack part.

Today, classical bank agencies gradually evolve towards agencies with one or several counters without money, ATM (automatic teller machine), safe room, and offices for commercial employees. The safe room is then the more significant zone inside the bank agency since all the available money is stored inside. As a consequence, all irregular behaviors or bank protocol infringement (involving either robbers or maintenance and cleaning employees) must be detected nearby the safe entrance. The protocol can be different for each bank. For instance, one of these rules is that only one person can enter the safe room at a time. In this case, the system must raise an alert when more than one person is inside the safe room. For bank experts, this part of the scenario (people number inside the safe) must be recognized with a very high confidence.

Moreover, it is interesting to recognize a robber approaching the safe entrance. Modeling all bank-attack precursors is a difficult task due to their large number and variety. We list here some examples.

(i) Employee attack: frequent, often stealthy, rapid, and hardly observable even for human beings. The bank employee is threatened but it is generally difficult to see the difference with a classical customer request.

Table 2: Validation results for a live installation of the bank agency monitoring system.

Scenario	Number of instances	True positive	False negative	False positive
With 3 persons	16	93.75%	6.25%	0%
With 2 persons	10	100%	0%	0%

(ii) Safe attack: they are not frequent. Bank employees and customers are threatened. People are shocked and things can take a bad turn.

(iii) Aggressive attack: bank employees and customers are threatened. The robber has lost his/her self-control, money is not the main motivation, and the robbery usually leads to a drama.

This scenario part is optional for bank-attack detection but important in order to anticipate potential actions and prevent any drama. Therefore, we have modeled a large set of scenarios to take into account the variety of bank robberies.

The behavior recognition assessment has been realized in live condition inside a bank agency during one hour, together with end-users. The assessment was based on the following scenarios.

(i) Scenarios with 2 persons: the bank employee is behind the counter. The robber enters the bank agency, goes to the counter, and threatens the employee. Both people go to the safe and the safe gate is opened.

(ii) Scenarios with 3 persons: the bank employee is behind the counter. A customer enters the bank agency, goes to the counter, and stays in front of it. After that, the robber enters the bank, joins the customer, and threatens the employee and the customer. The employee and the robber go to the safe and the safe gate is opened. The customer stays behind the counter or leaves the agency.

A true positive corresponds to an alert raise when a real bank attack happens (simulated by actors), a false negative is the miss of an alert raise when a real bank attack happens, and a false positive is an alert raised when no real bank attack happens. The *bank_attack* scenario with 3 persons was played 16 times. We obtained 93.75% of true positive, 6.25% of false negative, and 0% of false positive. The scenario with 2 persons was played more than 10 times and we obtained 100% of true positive. These results are summarized on Table 2.

The main reason why we obtained good true positive percentage is first that scenarios were precisely modeled thanks to the interaction with domain experts through an incremental process. The second reason of this success is the cooperation of two cameras to monitor the agency enabling to obtain better results due to the redundancy of information.

A second end-user assessment and validation phase will be held on a different bank agency with other scenarios at the beginning of 2005.

9.2. Lock chamber access monitoring system

Buildings with lock chambers at entrances are often faced with the problem of controlling how many people enter or exit the building. Sometimes these chambers are activated with a personal pass which allows the passage of the owner only. Nothing (but a human operator or a CCTV camera) can prevent the owner of a pass to let a second person to enter at the same time. Another motivation of this application is to be able to know exactly the number of people inside the building in case of fire alarms.

We built with the VSIP platform a lock chamber access monitoring system which is able to count the number of people passing through a general lock chamber defined as a closed space. The AMS can monitor the trajectories of people (where they come from and where they go); this feature is particularly useful in case of lock chambers with several access points.

This application uses automaton-based scenario recognition algorithms to monitor the trajectories of people and to count them. The limited field of view of cameras (e.g., see Figure 1f) and the high number of people that can be present at the same time in the field of view make this application challenging.

We have validated the lock chamber access AMS in two different cases. For the first one, a camera monitors a small lock chamber with two transparent doors at the opposite sides. For the second one, a camera monitors a larger lock chamber with 6 entrances on the four different sides, five of the entrance points are provided with doors.

For each sequence showing one or several persons passing from one entrance to another, we classify the result in four different classes.

(i) Good detection: the entrance and the exit points of each person passing through the lock chamber have been correctly detected for all persons.

(ii) Bad detection: entrance or exit points (or both) when one or more persons are incorrectly detected, or the number of persons detected is wrong.

(iii) Misdetection: someone is passing through the lock chamber but the system does not detect the person.

(iv) False alarm: a person is detected as passing from an entrance to an exit when there is nobody in the field of view of the camera.

Table 3 summarizes the validation results obtained by our AMS in both cases. Percentages are computed using the

Table 3: Validation results for a lock chamber access monitoring system.

Type of sequence	Number of instances	Good detections	Bad detections	Misdetections	False alarms
Small lock chamber, 1 person passing alone	17	94.10%	5.90%	0.00%	0.00%
Small lock chamber, 2 or more people passing together	25	96.00%	4.00%	0.00%	0.00%
Large lock chamber, 1 person passing alone	72	94.50%	5.50%	0.00%	2.00%
Large lock chamber, 2 or more people passing together	28	92.90%	7.10%	0.00%	2.00%

formula

$$\text{FAP} = \frac{\text{FA}}{\text{GD} + \text{BD} + \text{WD}}, \quad (1)$$

where FAP stands for "false alarm percentage" and FA, GD, BD, and WD are the total number of false alarms, good detections, bad detections, and misdetections over all the instances. Analog formulas are used for good detection, bad detection, and wrong detection percentages. The sequence used for the validation are all-day-life sequence, showing normal passage of people in small and large lock chambers as it happens during normal work activities (lock chambers are located in a company).

We are currently extending the validation of this application using a larger set of sequences and a live end-user assessment of this AMS is scheduled for the beginning of 2005.

10. CONCLUSION

Our goal is to obtain a reusable and performant activity monitoring platform (called VSIP). To achieve this goal, we believe that a unique global and sophisticated algorithm is not adapted because it cannot handle the large diversity of real-world applications. However, such a platform can be achieved if it can easily combine and integrate many algorithms. Therefore, we have presented three properties that an activity monitoring platform should have to enable its reusability for different applications and to insure performance quality. We have defined these properties as follows: modularity and flexibility, separation between algorithm code and a priori knowledge, and automatic evaluation. We have then proposed a development methodology to fulfill the last two properties and which consists in the interaction between end-users and developers during the whole development of a new activity monitoring system for a specific application.

We have then explained how we managed to develop VSIP following the given properties. We have shown how a shared data manager, the outsourcing of parameters, and the use of clear definitions of data structure enable to achieve modularity and flexibility. We have explained how the knowledge organization through description files and a language dedicated to the description of scenarios permit to obtain a clear separation between algorithms and a priori knowledge provided to the platform. We have shown that automatic evaluation allows developers to insure that new algorithms fulfill their specifications and keep platform performance over a set of selected applications. The evaluation framework allows also to apply learning techniques to tune the parameters of an AMS dedicated to a specific application. We have underlined that the interaction between end-users and developers was possible thanks to the definition of a video events ontology, an adapted language for scenario modeling and a tool to visualize the specified scenario models.

To illustrate the feasibility of our approach, we have presented VSIP, an activity monitoring platform fulfilling the three properties. This platform has been used to build activity monitoring systems dedicated to different applications taking advantage of a deep interaction with end-users. We have described three systems which have been validated and three other systems currently under development and whose validation will be completed in the near future.

The activity monitoring platform still presents some limitations, the most important being the difficulty, when adding a new algorithm to the platform to understand which are the algorithm weaknesses and how to fix them. So we are currently developing tools to extend the evaluation framework. The goal is to help developers to automatically analyze algorithm shortcomings in order to understand precisely under which hypothesis they can be used.

REFERENCES

[1] Y. Ivanov, C. Stauffer, A. Bobick, and W. E. L. Grimson, "Video surveillance of interactions," in *Proc. 2nd IEEE International Workshop on Visual Surveillance (VS '99)*, pp. 82–89, Fort Collins, Colo, USA, June 1999.

[2] J. Menendez and S. A. Velastin, "A method for obtaining neural network training sets in video sequences," in *Proc. 3rd IEEE International Workshop on Visual Surveillance (VS '00)*, pp. 69–75, Dublin, Ireland, July 2000.

[3] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts and shadows in video streams," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no. 10, pp. 1337–1342, 2003.

[4] R. Cucchiara, C. Grana, A. Prati, G. Tardini, and R. Vezzani, "Using computer vision techniques for dangerous situation detection in domotic applications," in *Proc. International Conference on Intelligent Distributed Surveillance Systems (IDSS '04)*, pp. 1–5, London, UK, February 2004.

[5] AVITRACK European Research Project http://www.avitrack.net.

[6] M. Borg, D. Thirde, J. Ferryman, F. Fusier, F. Brémond, and M. Thonnat, "An integrated vision system for aircraft activity monitoring," in *Proc. 6th IEEE Workshop on Performance Evaluation of Tracking and Surveillance (PETS '05)*, Breckenridge, Colo, USA, January 2005.

[7] H. Buxton and S. Gong, "Visual surveillance in a dynamic and uncertain world," *Artificial Intelligence*, vol. 78, no. 1-2, pp. 431–459, 1995.

[8] H.-H. Nagel, "Image sequence evaluation: 30 years and still going strong," in *Invited Lecture in Proc. 15th International Conference on Pattern Recognition (ICPR '00)*, vol. 1, pp. 1149–1158, Barcelona, Spain, September 2000.

[9] I. Haritaogou, D. Harwood, and L. S. Davis, "W4: real-time surveillance of people and their activities," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 809–830, 2000.

[10] N. M. Oliver, B. Rosario, and A. P. Pentland, "A Bayesian computer vision system for modelling human interactions," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 831–843, 2000.

[11] N. Johnson and D. C. Hogg, "Learning the distribution of object trajectories for event recognition," *Image and Vision Computing*, vol. 14, no. 8, pp. 609–615, 1996.

[12] R. Collins, A. Lipton, T. Kanade, et al., "A system for video surveillance and monitoring: VSAM final report," Tech. Rep. CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pa, USA, May 2000.

[13] ARDA VACE, http://www.ic-arda.org/InfoExploit/vace/.

[14] HomelSecurity ARPA (Advanced Research Projects Agency), http://www.hsarpabaa.com.

[15] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, "Tracking across multiple cameras with disjoint views," in *Proc. 9th IEEE International Conference on Computer Vision (ICCV '03)*, Nice, France, October 2003.

[16] IEEE Computer Society, Ed., *IEEE International Series of Workshops on Performance Evaluation of Tracking and Surveillance (PETS)*, IEEE Computer Society, 2002, http://visualsurveillance.org.

[17] T. Ellis, "Performance metrics and methods for tracking in surveillance," in *Proc. 3rd IEEE Workshop on Performance Evaluation of Tracking and Surveillance (PETS '02)*, Copenhagen, Denmark, June 2002.

[18] B. Georis, F. Brémond, M. Thonnat, and B. Macq, "Use of an evaluation and diagnosis method to improve tracking performances," in *Proc. 3rd IASTED International Conference on Visualization, Imaging and Image Processing (VIIP '03)*, Benalmádena, Spain, September 2003.

[19] S. Hongeng, F. Brémond, and R. Nevatia, "Representation and optimal recognition of human activities," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00)*, vol. 1, pp. 818–825, Hilton Head Island, SC, USA, June 2000.

[20] T-V. Vu, F. Brémond, and M. Thonnat, "Automatic video interpretation: a novel algorithm for temporal scenario recognition," in *Proc. 18th International Joint Conference on Artificial Intelligence (IJCAI '03)*, Acapulco, Mexico, August 2003.

[21] C. Tornieri, F. Brémond, and M. Thonnat, "Updating of the reference image for visual surveillance systems," in *Proc. International Conference on Intelligent Distributed Surveillance Systems (IDSS '03)*, London, UK, February 2003.

[22] A. Avanzi, F. Brémond, and M. Thonnat, "Tracking multiple individuals for video communication," in *Proc. International Conference on Image Processing (ICIP '01)*, Thessaloniki, Greece, October 2001.

[23] F. Cupillard, F. Brémond, and M. Thonnat, "Behaviour recognition for individuals, groups of people and crowd," in *Proc. International Conference on Intelligent Distributed Surveillance Systems (IDSS '03)*, London, UK, February 2003.

[24] F. Brémond, N. Maillot, M. Thonnat, and T. Van Vu, "Rr5189 - ontologies for video events," Tech. Rep., Orion Team, Institut National de Recherche en Informatique et Automatique (INRIA), Sophia Antipolis, France, May 2004.

[25] S. Moisan and M. Thonnat, "What can program supervision do for program reuse," *IEE Proceedings - Software*, vol. 147, no. 5, pp. 179–185, 2000.

[26] B. Georis, M. Mazière, F. Brémond, and M. Thonnat, "A video interpretation platform applied to bank agency monitoring," in *Proc. International Conference on Intelligent Distributed Surveillance Systems (IDSS '04)*, pp. 46–50, London, UK, February 2004.

Alberto Avanzi graduated as an engineer in electronics and telecommunications from Supélec (École Supérieure d'Électricité) in 2000, and as an engineer in electronics from Politecnico of Milan in 2001. He spent 9 months at INRIA Sophia Antipolis developing a long-term human tracking algorithm for video sequences. He then spent 9 months as a Consultant in computer science for the Politecnico of Milan and as a Professor in electronics. Since 2001, he has been part of Bull i-DTV Team as a Software Engineer but he has worked in the ORION Team in the framework of a joint venture between Bull and INRIA. From 2001 to 2003, he was deeply involved in the annotated digital video for intelligent surveillance and optimized retrieval (ADVISOR) European Project. He is now involved at the same time in the research work at ORION Team and in the industrialization of ORION code for Bull i-DTV Team. He is the author or coauthor of some scientific papers published in international journals or conferences in video understanding.

François Brémond is a Researcher in the ORION Team at INRIA Sophia Antipolis. He obtained his M.S. degree in 1992 at ENS Lyon. He has conducted research works in video understanding since 1993 both at Sophia Antipolis and at University of Southern California (USC), La. In 1997, he obtained his Ph.D. degree at INRIA in video understanding and pursued his research work as a postdoctorate student at USC on the interpretation of videos taken from unmanned airborne vehicle (UAV) in DARPA Project visual surveillance and activity monitoring (VSAM). He designs and develops generic systems for dynamic scene interpretation. The targeted class of applications is the automatic interpretation of indoor and outdoor partially structured scenes observed in particular with monocular color cameras. These systems detect and track mobile objects, which can be either human beings or vehicles, and recognize their behaviors. He is particularly interested in filling the gap between sensor information (pixel level) and behaviour recognition (semantic level). He is the author or coauthor of more than 30 scientific papers published in international journals or conferences in video understanding. He has cosupervised several Ph.D. theses. He has participated in several European projects and industrial research contracts.

Christophe Tornieri graduated as an engineer in computer sciences from École Supérieure en Sciences Informatiques (ESSI) in 2002. Since 2002, he has worked in the ORION Team at INRIA Sophia Antipolis, on automatic human behavior interpretation on video sequences. He has been particularly interested in problems related to illumination changes and context object detection. He has been deeply involved in the design and the implementation of the current video interpretation platform of ORION Team. From 2002 to 2003, he participated in the annotated digital video for intelligent surveillance and optimized retrieval (ADVISOR) European Project. Since 2003, he has worked on video interpretation algorithms embedded in trains. He is the author or coauthor of some scientific papers published in international journals or conferences in video understanding.

Monique Thonnat received in 1982 her Ph.D. degree in optics and signal processing from University of Marseille III. Her Ph.D. was prepared in the Spatial Astronomical Laboratory of CNRS. In 1983, she joined INRIA in Sophia Antipolis as full-time Research Scientist. She became a Senior Scientist in 1991 and in 1995, she created the ORION Project, a multidisciplinary research team at the frontier of computer vision, knowledge-based systems, and software engineering. She is the author or coauthor of more than 100 scientific papers published in international journals or conferences. During 3 years (from 1979 to 1982), she worked on image processing techniques for astronomy. Then, in 1983, she worked on pattern recognition and artificial intelligence techniques for complex object recognition and on computer vision for the automatic interpretation of 3D stereo data. Her more recent research activities involve the conception of new techniques for the reuse of programs (or program supervision) and on image understanding techniques for the interpretation of video sequences. She has supervised 20 Ph.D. theses (14 completed, 6 ongoing). She is directly involved in the application of her research in the industrial domain; in particular, in the framework of 6 European projects.