Scene Understanding: perception, multi-sensor fusion, spatio-temporal reasoning and activity recognition

François Bremond

HABILITATION A DIRIGER DES RECHERCHES

Presented at the
University of Nice – Sophia Antipolis

Scene Understanding: perception, multi-sensor fusion, spatio-temporal reasoning and activity recognition.

By

François Bremond

Defended on the 2nd of July 2007 with the jury composed of

Michel Barlaud,	Professor at the University of Nice Sophia Antipolis	President
Rita Cucchiara,	Professor at the University of Modena, Italy	Reviewer
Graeme A. Jones,	Professor at the University of Kingston, U.K.	Reviewer
Bernd Neumann,	Professor at the University of Hamburg, Germany	Reviewer
Malik Ghallab,	Research Director CNRS	Supervisor
Monique Thonnat,	Research Director INRIA	Supervisor

Acknowledgments

I would like to thank Monique Thonnat the head of ORION team for her support, competence and help during all these years to accomplish this HDR. She gave me the opportunity to make this work happen and gave me many ideas and advices. We had a lot of debates and enriching discussions about issues in video understanding. I really appreciated her deep sense of human relationships, her perseverance and her friendly never-ending encouragements all the time.
I would like to give a special thanks to the professors (Michel Barlaud, Rita Cucchiara, Graeme A. Jones, Bernd Neumann and Malik Ghallab) who participated in the Habilitation jury. Despite their busy schedules and agendas, they still made time to give me their great support and valuable comments.

I would also like to thank all the members of the ORION team: B. Boulay, B. Bui, E. Corvee, M.B. Kaaniche, L. Le-Thi, R. Ma, V. Martin, S. Moisan, A. T. Nghiem, J.L. Patino-Vilchis, T. Raty, A. Ressouche, J.P. Rigault, V. Valentin, V.T. Vu, M. Yoshimura, N. Zouba, M. Zuniga, and people who I have worked in the past: Cathy and J.C. Ossola, R. Vincent, H. Tolba, N. Chleq, N. Mayot, F. Fusier, G. Davini, B. Georis, F. Cupillard, A. Avanzi, C. Tornieri, F. Duchene and M. Maziere. I also greatly appreciated the team assistance Catherine Martin for her friendly help in everyday tasks.

Now, I would like to thank all my friends from INRIA, Agnes and other personal friends Gael, Fred, Virginie, and also friends from my environment association (TRAVISA, ADEV) and my Buddhist friends. In particular, I am deeply grateful to Daisaku Ikeda, my Buddhist mentor who taught me the importance of self-control and human relationships in life, and how to enjoy life even in difficult situation. On the human level, I constantly get great encouragements reading his books and practising Buddhism.

Finally, I am grateful to my family, especially to my wife, my two boys, my brothers and my parents for their love, and patience. They have always supported my choices and gave me courage and taught me perseverance. Without them this work would have never been finished.

Thanks to all those people who I might have forgotten to name in this section.

Document organization

This document is a summary of my research activities since October 1997, the date of my PhD defense. This document constitutes the dissertation required to show my capability to supervise young researchers such as PhD students. My research activities are organized in the five following axes:

- Perception for scene understanding.
- Maintaining 3D coherency throughout time (physical world).
- Event recognition (semantic world).
- Performance evaluation and learning (autonomous systems).
- Knowledge acquisition (interactive systems).

The chapter 4 presents globally these research themes.

The chapter 5 describes the work done to conceive a vision platform in order to easily build efficient modules for scene interpretation.

The chapter 6 presents the work on mobile object tracking and on information fusion.

The chapter 7 describes the work on the representation and recognition of events using different types of formalism.

The chapter 8 presents the work on performance evaluation for video interpretation algorithms and preliminary work on learning techniques to tune program parameters.

The chapter 9 presents the work on knowledge acquisition to help the users to describe their scenarios of interest.

The chapter 10 is a conclusion of this dissertation proposing several perspectives and answering long term issues in scene interpretation.

Table of Content

Chapter 4

Research Themes

Scene understanding is the process, often real time, of perceiving, analysing and elaborating an interpretation of a 3D dynamic scene observed through a network of sensors. This process consists mainly in matching signal information coming from sensors observing the scene with models which humans are using to understand the scene. Based on that, scene understanding is both adding and extracting semantic from the sensor data characterizing a scene. This scene can contain a number of physical objects of various types (e.g. people, vehicle) interacting with each others or with their environment (e.g. equipment) more or less structured. The scene can last few instants (e.g. the fall of a person) or few months (e.g. the depression of a person), can be limited to a laboratory slide observed through a microscope or go beyond the size of a city. Sensors include usually cameras (e.g. omni directional, infrared), but also may include microphones and other sensors (e.g. optical cells, contact sensors, physiological sensors, radars, smoke detectors).

Scene understanding is influenced by cognitive vision and it requires at least the melding of three areas: computer vision, cognition and software engineering. Scene understanding can achieve four levels of generic computer vision functionality of detection, localisation, recognition and understanding. But scene understanding systems go beyond the detection of visual features such as corners, edges and moving regions to extract information related to the physical world which is meaningful for human operators. Its requirement is also to achieve more robust, resilient, adaptable computer vision functionalities by endowing them with a cognitive faculty: the ability to learn, adapt, weigh alternative solutions, and develop new strategies for analysis and interpretation. The key characteristic of a scene understanding system is its capacity to exhibit robust performance even in circumstances that were not

foreseen when it was designed (G. H. Granlund). Furthermore, a scene understanding system should be able to anticipate events and adapt its operation accordingly. Ideally, a scene understanding system should be able to adapt to novel variations of the current environment to generalize to new context and application domains and interpret the intent of underlying behaviours to predict future configurations of the environment, and to communicate an understanding of the scene to other systems, including humans. Related but different domains are robotic, where systems can interfere and modify their environment, and multi-media document analysis (e.g. video retrieval), where limited contextual information is available.

4.1 Objectives

Despite few success stories, such as traffic monitoring (e.g. Citylog), swimming pool monitoring (e.g. Poseidon) and intrusion detection (e.g. ObjectVideo), scene understanding systems remain brittle and can function only under restrictive conditions (e.g. during day rather than night, diffuse lighting conditions, no shadows), having poor performance over time, they are hardly modifiable, containing little a priori knowledge on their environment. Moreover, these systems are very specific and needs to be redeveloped from scratch for other applications. To answer these issues, most researchers have tried to develop new vision algorithms with focused functionalities, robust enough to handle real life conditions. Up to now no vision algorithms were able to address the large varieties of conditions characterising real world scenes, in terms of sensors conditions, hardware requirements, lighting conditions, physical object varieties, application objectives...

My goal is to design a framework for the easy generation of autonomous and effective scene understanding systems. This objective is very ambitious; however the current state-of-the-art techniques in cognitive vision have lead to partial solutions [Cohn et al., 2006], [Dee and Hogg, 2005], [Needham et al., 2005], [Nevatia et al., 2004], [Remagnino et al., 2006], [Crowley, 2006b], [Jodogne and Piater, 2005] and [Xiang and Gong, 2006b]. I believe that to reach this goal, a holistic approach is needed where the main scene understanding process relies on the maintenance of the coherency of the representation of the global 3D scene throughout time. This approach which can be called *4D semantic interpretation*, is driven by models and invariants characterising the scene and its dynamics. Scene understanding is a complex process where information is abstracted through four levels; signal (e.g. pixel, sound), perceptual features, physical objects, and events. The signal level is characterized by strong noise, ambiguous, corrupted and missing data. The whole process of scene understanding consists in filtering this information to bring forth pertinent insight of the scene and its dynamics. To fulfil this objective, models and invariants are the crucial points to characterise knowledge and insure its consistency at the four abstraction levels. For instance, I have defined formalisms to model the empty scene of the surrounding (e.g. its geometric), the sensors (e.g. calibration matrices of the cameras), the physical objects expected in the scene (e.g. 3D model of human being), and the scenarios of interest for users (e.g. abnormal events). The invariants (called also regularities) are general rules characterising the scene dynamics. For instance, the intensity of a pixel can change significantly only in two cases: change of lighting condition (e.g. shadow) or change due to a physical object (e.g. occlusion). A second rule for example, verifies that physical objects cannot disappear in the middle of the scene.

There is still a open issue consists in determining whether these models and invariants are given a priori or are learned. The whole challenge consists in organising all these knowledge in order to capitalise experience, share it with others and updating it along experimentation. To face this challenge, tools in knowledge engineering such as ontology, are needed.

4.2 Summary of research work done

To concretize this approach my research activities have been organised within the following five axes. For each axis, I summarize the main scientific challenges I have addressed.

- **Perception for scene understanding (perceptual world)**. A first axis is to collect and develop vision algorithms to handle all the varieties of real world conditions. The goal of all these algorithms is to detect and classify the physical objects which are defined as interesting by the users. A first difficulty consists in developing robust segmentation algorithms for detecting the physical objects of interest. The most common algorithms estimate the motion within the videos. These algorithms are based on the hypothesis that the objects of interest are related to what is moving in the video, which can be inferred by detecting signal changes. Unfortunately, these algorithms have the tendency to detect a lot of noise (e.g. due to light changes) together with the objects of interest. A second difficulty consists in extracting meaningful features characterising the objects of interest. Most of algorithms compute features relatively to the trajectory of the physical objects. Robust descriptors characterising the shape of physical objects still need to be developed. A third difficulty is to establish under which hypotheses the algorithms are valid, and to understand their limits. In the same way, algorithms processing other media and modalities (e.g. audio, contact, radar) need more development to complement the information extracted from video streams. A still open issue is to establish the precision and likelihood of these processes.

- **Maintenance of the 3D coherency throughout time (physical world)**. A second axis consists in combining all the information coming from the different sensors observing the detected physical objects and in tracking these objects throughout time. Despite all the works done in this domain within the last 20 years, fusion and tracking algorithms remain brittle. To guarantee the coherency of these tracked objects, spatio-temporal reasoning is required. Modelling the uncertainty of these processes is also an open issue. Another question that we need to answer is at which level this information should be combined. Information fusion at the signal level can provide more precise information, but information fusion at higher levels is more reliable and easier to realise. In any case, a precise formalism is needed to combine uncertain information coming from heterogeneous sensors.

- **Event recognition (semantic world)**. At the event level, the computation of relationships between physical objects constitutes a third axis. The real challenge is to explore efficiently all the possible spatio-temporal relationships of these objects that may correspond to events (called also actions, situations, activities, behaviors, scenarios, scripts and chronicles). The varieties of these events, called generally video events, are huge and depend on their spatial and temporal granularities, on the number of the physical objects involved in the events, and on the event complexity (number of components constituting the event and the type of temporal relationship). So the challenge is to explore this large event space without getting lost in combinatorial searches.

- **Evaluation, control and learning (autonomous systems).** To be able to improve scene understanding systems, we need at one point to evaluate their performance. The classical methodology for performance evaluation consists in using reference data (called ground truth). However, generating ground truth is tiresome and error prone. Therefore, an issue is to perform the performance evaluation stage using unsupervised techniques. Once evaluation is possible, a real challenge consists in optimising the scene understanding system using machine learning techniques in order to find the best combination of programs, the best set of program parameters with the best control strategies to obtain an efficient and effective real-time process. The difficulty is three fold. First, programs depend on environmental conditions and the program optimisation process has to be dynamic to take into account of environmental changes and available resources. Second, all these programs are interlinked with each others, so the modification of one program parameter can mess the functioning of all other programs. Finally, the knowledge on these programs is not formalised and usually, even the developers cannot tell what will be the program output under even specific conditions. Another way to improve system performance is to add higher reasoning. Scene understanding is essentially a bottom-up approach consisting in abstracting information coming from signal (i.e. approach guided by data). However, in some cases, a top-down approach (i.e. approach guided by models) can improve lower process performance by providing a more global knowledge of the observed scene or by optimising available resources. For instance, the global coherency of the 4D world can help to decide whether some moving regions correspond to noise or to physical objects of interest. So, the fourth axis in my research consists in exploring program supervision (including evaluation) and machine learning techniques for the easy generation of effective real-time scene understanding systems.

- **Communication, Visualisation and Knowledge Acquisition (interactive systems).** Even when the correct interpretation of the scene has been performed, the scene understanding system still has to communicate its understanding to the users. So user interactions constitutes a fifth axis. There are at least three types of users: program developers, experts of the application domain and end-users. The first challenge is to enable program developers to understand all specific components and in the same time, the global architecture of the scene understanding system, so that they can adapt efficiently their programs and configure and install the system on a site. To

reach this goal, formalism is required to express program knowledge. Second, if we want an effective system, the a priori knowledge needs to be formalised to enable the domain experts to describe their methodology for analysing the scene. For instance, a tool and a dedicated ontology have to be provided to assist the experts in defining the scenarios that the system has to recognize. To help this process, a graphical tool can be designed to generate and visualise 3D virtual animations illustrating these scenarios. In complement of these tools, clustering techniques can be used to mine the frequent activities (i.e. event patterns or time series) occurring in the scene. Moreover, if we want the system to be used, special care needs to be brought to display what has been understood to the final users. An ergonomic interface on an adapted media (e.g. immersive reality or PDA personal digital assistant), a convenient representation of the scene (e.g. virtual 3D scene, augmented reality), and an intuitive vocabulary are the necessary devices to be provided to the end-users. Besides these devices, the system needs to take into account feedback information from the end-users to be able to adapt its performance to the specific goals of each user.

All along these years, for each axis, I have tried to establish the scientific and technological foundation for a the Scene Understanding approach through the design of systems dedicated to more than 20 applications (e.g. Visual Surveillance, Activities Monitoring, Ambient Intelligence, Perceptual User Interface, Health Care, and Animal Behavior Analysis), in direct contact with users ranging from end-users (e.g. human operators, managers, domain experts), to integrators, hardware and software providers. I believe that applications are a key point in conceiving effective scene understanding systems for three reasons: first they enable to answer real challenges, second they are the necessary conditions to enable experts of the application domain to provide the precise knowledge on the scene and finally they are the main way with the help of end-users to evaluate the performance of the final system. These real world systems could not have been conceived, only with the help of 7 PhD students and 9 research engineers that I have supervised.

This document aims at summarising these experiences (i.e. partial solutions) and exploring the still open issues.

4.3 Plan of the dissertation

The following of the manuscript described successively the research that I have lead along these five axes.

More precisely, the chapter 5 entitled “perception for scene understanding” describes the work I have done to conceive a vision platform (called VSIP) to build efficient components for scene understanding.

The chapter 6, named “maintaining 3D coherency throughout time”, presents work done on object tracking and information fusion. In particular, two types of fusion will be detailed: (1) video camera coupled with other sensors (contact sensors and optical cells), (2) multiple cameras with overlapping field of view. In chapter 6, we have tried to underline the invariants and rules necessary to maintain the 3D coherency throughout time.

The chapter 7 named “event recognition”, describes works on event representation and event recognition, using different types of formalism: finite state automata, HMM, Bayesian networks, and temporal constraint networks.

The chapter 8 entitled “evaluation and learning”, presents works on performance evaluation of video understanding algorithms and early work done on the configuration of video processing programs. In this chapter, we describe a first attempt on learning the parameters of video processing programs.

The chapter 9 named “knowledge acquisition” presents work on knowledge modelling techniques for helping end-users to describe their scenarios of interest. We explain how these techniques can be guided by a dedicated ontology. In particular, we describe a tool for the automatic generation of simulated videos from end-users specifications. We also present early works for learning automatically scenarios models from videos.

In this document, references [Bremond x] correspond to the papers I wrote (or I participated in the writing) describing my work and are listed in the CV part, whereas other references [author x] are listed at the end of the document.

Chapter 5

Perception for scene understanding

This chapter aims at explaining the choice and methodology taken to build a generic scene understanding platform called VSIP (Video Surveillance Intelligent platform). This platform or tool box, aims at the easy generation of dedicated scene understanding systems. It has been designed in three stages during a period of 12 years.

First, from 1994 to 1997, based on the state of the art, we had specified a general model of the video understanding process [Bremond 1]. This model contains the main video understanding functionalities which can be summarized in general with the following tasks: (1) object detection and classification, (2) object tracking and (3) event recognition. We have proposed to use a global representation of what has been understood in the observed scene, based on a 3D geometric representation of the scene, including contextual knowledge [Bremond 4, 14 and 16]. Meanwhile from 1994 to 2000, we had gained some experiences in developing perception algorithms through European projects in Video Surveillance in parking, subways, super-markets (e.g. PASSWORDS, AVSPV) [Bremond 2, 5, 16, 17, 18, 19 and 20], and through a DARPA project in Unman Arial Vehicles surveillance (VSAM, Video Surveillance and Activity Monitoring) [Bremond 6, 8, 21, 22, 23 and 24].

Second, from 2000 to 2004, we had collected a set of perception algorithms corresponding to the basic functionalities in video understanding, in an exhaustive and structured way. This work had been done thanks to the European project ADVISOR on subway monitoring [Bremond 25, 26, 30, 31 and 39] and to an industrial project on bank agency monitoring [Bremond 13, 27, 29, 31 and 38].

Third, from 2004 to 2006, we had complemented these programs with specific algorithms dedicated to particular applications (more than 10) [Bremond 36, 37, 38, 40, 41 42, and 44]. For instance, we have developed a specific algorithm for door detection to determine when a door is open, and to differentiate the door from people motion [Bremond 32]. Through out the PhD thesis of Bernard Boulay [Bremond 10 and 36] we have also proposed an algorithm to recognise posture of people evolving in a scene observed by one video camera. During this period, we had tried to handle varieties of real world scenes, while improving the robustness of the generic algorithms for the most common scene understanding tasks.

Since most of the work has been done on Videos and the requirements of classical real world application, we have focused most of the perception algorithms on video processing. Thus, this dissertation document will study mostly vision algorithms and we will give only early results on perception from other sensors. To complement these results, the following articles can be read [Dong and Pentland 2006, Cucchiara et al 2006, Foresti and Regazzoni 2002, Al-Hames 2005] For simple sensors (optical cells and contact sensors), we have mainly developed filtering techniques to get rid of outliers in order to get a coherent signal. For instance, we have designed a system to recognise precisely the shape of people travelling through a passage observed by a combination of cameras (a camera observing the passage from the top and 5 lateral cameras) coupled with LEDs (to sharpen people detection) and optical cells [Bremond 43]. For complex sensors (microphones), we have relied on processing techniques developed by other teams through collaborations. For instance, we were able to extract audio and video information from on board train scenes to detect abnormal events such as “vandalism against a window” [Bremond 51].

Thus, algorithms processing other media (e.g. audio, contact) are needed to complement the information extracted from video streams. A still open issue is to establish the precision and likelihood of these processes in their contributions to extract meaningful content on the observed scene. As far as I am concerned, the main issue with knowledge extraction from other sensors is more an information fusion issue, which will be discussed in the next chapter.

All these experiments in scene perception have underlined two recurrent challenges. First, the brittleness of segmentation algorithms brings uncertainty and errors through the whole understanding process. Image segmentation is a classical but hard problem in perception, which is becoming tractable in video understanding, relaying on the assumption that the interesting objects are the physical objects in motion in the scene. Although good segmentation results can be obtained in specific situations (e.g. stable lighting conditions), segmentation algorithms have the tendency to detect a lot of noise (e.g. lack of contrast, ambiguous, corrupted and missing data) together with the objects of interest. Second, the difficulty of extracting meaningful features prevents from obtaining precise descriptions of the physical objects of interest. Most of algorithms compute features relatively to the trajectory of the physical objects, but remain weak to extract descriptors characterising the shape of the physical objects. Therefore, most of research works in video understanding have tried to address these issues. Since usually these two challenges need first to be addressed to pursue higher reasoning, this chapter 5 will focus on explaining the reasons of these difficulties and on proposing few solutions.

To detail the choices taken for building the VSIP Platform, this chapter is organised as follows. Section 1 studies the state of the art in video understanding. Section 2 enumerates the large varieties of conditions that we have to face when developing scene understanding systems. Section 3 describes briefly the main VSIP algorithms, which have been detailed in collaboration with Benoit Georis during his PhD on program supervision for video understanding. Section 4 details few advanced algorithms for handling specific problems in scene understanding. Conclusion reviews advantages and limitations of the VSIP platform.

5.1 State-of-the art in Video Understanding

This section aims at presenting different video processing techniques which have been designed under different assumptions and for different purposes. Each technique alone is not sufficient to address the variety of all possible situations and to be elected as an acceptable solution regarding the complexity of the video understanding problem. In the meantime, integrated approaches are most of the time designed for particular applications and are not flexible enough to be reused for other applications. In consequence, this description stresses (1) the large variety of video processing programs and (2) the need of having a library of programs combined with an external control component which would be in charge of selecting, tuning and combining programs in function of the application requirements.

Video understanding is a process which consists in the recognition of events (either predefined by an end-user or learnt by a system) in a given application domain (e.g., human activities) starting from a pixel analysis up to a symbolic description of what is happening in the scene viewed by cameras. This process implies to use several consecutive techniques to reach this objective. There is a huge literature describing such video processing techniques and several surveys list and categorize these techniques in a more or less exhaustive way. For instance, in [Cedras 1995], the authors present a survey on motion-based recognition techniques ranging from human tracking and recognition to lip-reading and gesture interpretation. In [Aggarwal 1997] the authors classify the techniques for human motion analysis in three parts: motion analysis, tracking moving humans from a single or from multiple views and recognition of human activities. In [Gavrila 1999], the author classifies the techniques for analysing human movements in three categories: 2D model-free approaches, 2D model-based approaches and 3D approaches. In [Pentland 2000], the author proposes an application-centered classification, such as surveillance, monitoring, smart rooms or perceptual user interfaces. Finally, two recent surveys [Wang et al., 2003] [Moeslund et al., 2006], include the latest research (from 1997 to 2005) according to a similar classification than the one proposed in [Aggarwal 1997].

However, our objective here is not to realize a survey but instead to present relevant techniques which fall into one of the two general categories composing a video understanding process (the structure is identical as the one in [Aggarwal 1997] but not necessarily restricted to human motion analysis): (1) object detection and classification and (2) spatio-temporal analysis. This structure has the advantage of providing a global view of the video understanding process even if the categories are interrelated and may overlap. For instance, a technique based on a Kalman filter usually realizes at the same time both object detection and tracking [Piater 2001]. Along with a brief description of these techniques, we give indications concerning the assumptions which have guided their design and which influence their conditions of use. So, this state of the art focuses more on video processing techniques rather than on event recognition techniques, to highlight the large variety of video processing programs, which are greatly dependent on the processed videos. In chapter 7, a study on event recognition techniques is presented to emphasise new issues in scene understanding.

Object detection and classification

Several approaches have already been investigated in order to detect and classify objects of interest in a scene. Concerning the detection task, most of the techniques use motion information to detect the regions which correspond to objects of interest. They are presented in the following.

Background subtraction

Although many algorithms have been designed for image segmentation [Herbulot et al., 2006], background subtraction is probably the most widely used technique for detecting interesting objects from images coming from a static camera [Lim et al 2005, Monnet et al 2003, Cavallaro et al 2005, Salvador et al 2004,]. Each particular technique depends on the type of background modelling which is used. The simplest technique consists in taking a reference image which is temporally averaged over time using either the median value (e.g., [Yang 1992]) or the mean value (e.g., [Marcenaro 2000]). This reference image is subtracted from the current image and the resulting difference image is threshold in order to obtain moving regions. This technique may perform well for a static background but fails as soon as the background is more complex and dynamic (e.g., lighting variations, slow moving objects). In addition, the use of a unique threshold value for the entire image is another limitation. In consequence, several alternatives have been proposed.

The *Pfinder* system [Wren1997] uses a multi-statistical model of colour and shape for the tracked objects and a single Gaussian per pixel for the background model. After an initialization period with an empty scene, this system reports fairly good results for an indoor scene with few persons; however, no results are reported for outdoor scenes. On the opposite, the technique presented in [Stauffer1999] is intended to be applied on outdoor scenes and is especially good at handling moving elements of the scene such as swaying trees. They model the values of a particular pixel as a mixture of Gaussians. However, a main problem is that this online model suffers from slow learning at the beginning, especially in busy environments. In order to overcome this problem, an extension of this method is proposed in [Kaewtrakulpong and Bowden, 2001]. Nevertheless, both methods are not well appropriate to provide an accurate segmentation for low-contrasted images due to the Gaussian smoothing. Another solution is the *W4* (Who-When-Where-What) system [Haritaoglu et al., 2000] which models the background with three values for each pixel: the minimum intensity, maximum intensity values and the maximum intensity difference between consecutive frames over a training period. However, the authors recognize that shadows or sudden illumination changes (e.g., due to a cloud) cannot be handled with this technique.

Concerning problems of low-contrasted targets and sudden changes, they are treated in [Elgammal et al., 2000] where a non-parametric model of the background is able to quickly adapt itself to changes and to better discriminate low-contrasted targets. Concerning the shadow issue, a model of background with three Gaussians is proposed in [Friedman and Russell, 1997] for traffic monitoring applications. A Gaussian models each of the three parts

of the scene: the road, the vehicles and the shadow. Another interesting work [Cucchiara et al., 2003] proposes to add several criteria on pixel values in the HSV (Hue-Saturation-Value) colour space during the background subtraction step in order to suppress pixels corresponding to shadows.

Frame differencing

A frame differencing technique can be viewed as a background subtraction technique where the reference image is replaced at each time step by the previous frame [Lipton et al., 1998]. This technique thus provides the fastest adaptation to environment changes. However, depending on the frame rate which is used and the speed of objects, holes may appear inside the detected moving regions (e.g., for slow moving objects). A first improvement is reported in [Bergen1 et al., 1992] where the authors use three-frame differencing to detect moving regions. A second improvement is presented in the VSAM (Visual Surveillance And Monitoring) system [VSAM 1998], [Collins et. al., 2000] where three-frame differencing is combined to background subtraction to produce a robust detection of moving regions. This later technique is shown to be fast and efficient, but the report fails to present a detailed performance evaluation.

Optical flow

The most classical approach to detect objects in a video consists in motion estimation [Zhai and Shah, 2006], [Velastin et al., 2006], [Velastin et al., 2005] and [Desurmont et al., 2006]. In particular, the optical flow technique typically computes the motion field from image intensities in order to isolate regions having a coherent motion [Barron et al., 1994], [Fejes and Davis, 1997], [Nagel, 1998], [Middendorf and Nagel, 2000]. For instance, this technique can discriminate an individual walking at counter flow in a crowded scene. Another advantage of this technique is that it can be used even in presence of camera motion. A drawback of this technique is that the computation of derivatives for each pixel is often required, thus making the method computationally expensive. However, alternatives to this pixel-based approach exist. For instance, in [Coimbra et al., 2003], the authors propose to use MPEG-2 motion vectors as a basis for obtaining the motion field. Then, they apply specific rules and filters to obtain a smooth motion field (e.g., they use the gradient magnitude present in AC coefficients of the discrete cosine transform as a confidence measure of motion vectors). In consequence, this alternative is able to work in real-time conditions.

Feature detection

Instead of using motion information between consecutive frames, another class of methods for the object detection task consists in computing image features [Hall et al., 2004], [Mark and Gavrila, 2006], [Vaswani et al., 2003], and in particular contours [Vaswani et al.,

2005], [Veeraraghavan et al., 2005], from a single image. The watershed technique [Beucher and Lantuejoul, 1979] defines the contours in an image as the watersheds of the gradient modulus of the grey value function (considered as a relief surface). This technique has the advantage to be non-parametric compared to those which threshold the gradient. This method and its improvements have been applied to detect the road lanes for obstacle detection in a traffic monitoring application [Beucher and Bilodeau, 1994]. This latter improvement consists in the introduction of a fast watershed transform which enables to speed up the processing time to 5 frames per second.

An alternative to watersheds is active shape models [Cootes et al., 1995]. For instance, in [Baumberg et al., 1994] the authors use active shape models to detect (and also track) the contours of pedestrians. The model is derived from training shapes which are described by known feature points of a B-spline. Then, a PCA (Principal Component Analysis) is applied on these points to obtain a reduced representation. This model is able to detect (and track) deformable shapes. However, this technique requires a training stage but also a manual initialization or an initialization with another technique (such as background subtraction).

Object classification

According to a predefined semantics, once objects are detected, the classification task aims at assigning a label to each object, This task often implies a matching stage with a model of objects to classify. This matching stage can include techniques such as template-base object detection [Ferryman et al., 2000], [Bremond 12], crowd detection [Zhao and Nevatia, 2004], [Andrade et al., 2006], [Guler and Farrow, 2006] human-body detection [Gourier et al., 2006], [Mittal et al., 2003], [Cucchiara et al., 2005], [Dimitrijevic et al., 2005], [Agarwal and Triggs, 2006], face detection [Viola and Jones, 2004], [Zhou and Chellappa, 2005], [Foresti et al., 2003], and gait analysis [Nixon et al., 2005], [Kale et al., 2005]. For instance, in [Haritaoglu et al., 2000], moving regions are classified into one of three predefined classes (single person, people in a group, other objects). For a single person, a comparison is made between the normalized vertical and horizontal projections of the moving region and the projection templates of four different postures (standing, sitting, crawling/bending and lying down) pre-computed experimentally. For a group, they first try to determine how many persons are present in the group by detecting heads. This is done by extracting significant peaks of the vertical projection histogram of the moving region. However, a main disadvantage of this technique precisely consists in estimating with a high confidence whether the moving region corresponds to a single person or to a group of persons. Another drawback is the sensitivity of this technique to shadows (i.e., wrong histogram).

In [Collins et al., 1999], the authors classify the moving regions into one of three classes (human, vehicle and human group) with a three-layer neural network trained with a back propagation algorithm. The inputs are the dispersedness, the area and the aspect ratio of the moving region, associated to the zoom factor. This technique is fairly efficient but it is viewpoint dependent. Another solution is proposed in [Kuno et al., 1996] where the authors use a simple silhouette-based shape representation which combines the mean and the standard

deviation of the silhouette projection histograms with the bounding box of the moving region. This method reliably distinguishes humans from vehicles.

Other techniques use a set of image features (e.g., colour, texture) in order to perform the classification. We can mention the work reported in [Oren et al., 1997] where wavelet coefficients are used as low-level intensity features to represent the appearance properties. These coefficients are obtained by applying a differential operator at various locations, scales and orientations (vertical, horizontal, corner) on the images. A support vector machine is used as the classifier which learns the relationship between these coefficients from a set of positive and negative examples. Compared to classical techniques (i.e., those which use colour or texture), this technique can detect and classify pedestrians from front or rear views, when there is no reliable colour or texture patterns and also in cluttered scenes.

Finally, concerning vehicle applications, we can mention the ESPRIT VIEWS system aiming at monitoring vehicles in roundabouts for traffic control [Buxton and Gong, 1995]. Another interesting work which models vehicles in 3D is proposed in [Nagel, 2000]. These parameterizable models can be distorted in order to improve robustness.

Spatio-temporal analysis

The objective of the spatio-temporal analysis task is to maintain a 3D representation of what is happening in the scene over a period of time (e.g., few seconds up to a minute) by resolving potential incoherence, based on visual invariants in the scene (e.g., a person cannot go through a wall). This task implies the tracking of the previously classified objects of interest and sometimes a fusion of information coming from the different cameras (i.e., in a multi-camera configuration).

Region-based tracking

Region-based tracking is perhaps the most popular technique for performing the matching of objects over time. Most of these techniques rely on a Kalman filter (or equivalent) to perform the association between objects. In [Cox and Higorani, 1996], the authors propose an efficient implementation of a multiple hypotheses tracker. They keep all information about the associations between the currently tracked objects and the new observations until they are able to decide which are the right associations (i.e., they remove incoherent associations). Each hypothesis (i.e., possible association) corresponds to a Kalman filter. In [Piater and Crowley, 2001], a Kalman filter is also used to effectively track interacting targets in a scene. However, this technique suffers from a problem of target initialization. In addition, due to the Gaussian distribution assumption, this technique is not conceived to handle multi-modal distributions of the state parameters. For instance, this technique is inadequate in dealing with the simultaneous presence of occlusions and a cluttered background resembling the tracked objects.

Another solution consists in using a particle filter or more generally probabilistic techniques [Du and Piater, 2006], [Gelgon et al., 2005] and [Coue et al., 2006]. For instance, in [Nummiaro et al., 2002], the authors use a colour particle filter to track objects. The objects are modelled by a weighted histogram which takes into account both the colour and the shape of the objects. The particle filter thus compares the histograms of objects of frame **t-1** and **t** at the sample positions in order to decide whether objects match or not. This technique is robust to partial occlusions, is rotation and scale invariant and is computed efficiently. A current limitation is that the tracker has difficulties to handle significant changes in object appearance.

Feature-based tracking

The tracking of features is a good alternative to region-based tracking [Gavrila and Munder, 2007], [Mathes and Piater, 2006], [Ozuysal et al., 2006] and [Mittal and Davis, 2003]. For instance, in [Shi and Tomasi, 1994], the authors first compute the covariance matrix in a small image window. This window is shifted over the entire image in raster order. Then, the two eigenvectors corresponding to the two maximal eigenvalues of the covariance matrix are kept. These eigenvectors define the directions of maximal gradient and are thus used to detect corners (or T shapes) in the image. Compared to region-based techniques, this technique is robust to partial occlusions and can also be used to track crowds. However, the computational cost is the main disadvantage. In addition, the computation of features is not stable, often leading to tracking errors.

Model-based tracking

Compared to the previous techniques, model-based tracking algorithms are still often considered as sophisticated techniques which are difficult to implement but in the meantime, they usually perform better in complex situations (e.g., clutter, occlusions) [Cucchiara et al., 2001], [Cavallaro et al., 2005b] and [Plankers and Fua, 2001]. Some of them are based on 3D information computation [Desurmont et al., 2006] and [Urtasun et al., 2006]. In [Wren et al., 1997], the authors model the human body in six body parts: head, torso and four limbs. This technique performs well for tracking a single person. However, it has not been conceived to handle multiple objects. The drawback of this simple technique has been improved in [Haritaoglu et al 2000] where shape analysis is used to detect and track people and their parts.

In [Zhao and Nevatia, 2004], the authors present an articulated human walking model composed of two parts: global motion and limb motion. The global motion of objects is tracked in 3D using ellipsoid shape models. Then, the locomotion mode (walking, running, standing) and the 3D body posture are estimated by making inference in an a priori locomotion model. Robust results are reported on challenging sequences containing occlusions, cast shadows and reflections.

Multiple camera tracking

Based on the observation that most of the tracking techniques fail in complex situations (e.g., a lot of interacting objects), several works have studied active vision methods [Foresti and Micheloni, 2003] and [Smith et al., 2005] and mainly fusion of information techniques with multiple cameras [Remagnino et al., 2004a], [Du et al., 2006], [Cavallaro 2005], [Martinez del Rincon et al., 2006], [Krahnstoever et al., 2006] and [Snidaro et al., 2003] to improve results and robustness. This fusion is either performed on the output of different algorithms applied on the same camera [Siebel and Maybank, 2002] or on information coming from several cameras [Black and Ellis, 2001]. In particular, an interesting work [Mittal and Davis, 2003] has addressed the problem of multi-view tracking using synchronized cameras. This system is able to combine information coming from multiple camera pairs (i.e., up to 16 synchronized cameras were used in the experiments) in order to handle occlusions and correctly track densely located objects in a cluttered scene. However, due to the complexity, the system is not yet able to work in real-time (i.e., it currently takes 5 seconds per processing step).

In [Javed et al., 2003], the proposed system is able to track the objects in the scene using non-overlapping field of view cameras. First, the system learns the camera topology and the path probabilities of objects during a training phase. This is based on the assumption that people and cars tend to create redundancy in paths they follow. Then, the associations are performed with a maximum a posteriori estimation framework.

In case cameras have overlapping field of view, a reliable technique for tracking groups of people using multiple cameras is presented in [Bremond, 30]. First, the authors compute a temporal graph of objects for each camera. Nodes of this graph represent the objects of interest (and their properties) while edges are temporal links over time. Then, they fuse the different temporal graphs of objects created for each camera, by using three criteria: 3D position, 3D dimension and temporal criteria. This latter criterion increases the fusion score between two objects at time **t** if their corresponding parents at time $t-1$ have previously been fused.

Finally, as there are few works addressing the problem of tracking groups of people, we want to mention another work [McKenna et al., 2000], and even if the authors use a single camera. The proposed system maintains a representation of regions, persons and groups, by using simple heuristic rules (e.g., a group is composed of several persons and contains therefore one or more regions, a region which matches more than one group triggers the merging of these groups into a new bigger group). This system is able to process video sequences at various frame rates without any modification in parameter setting. However, only preliminary results are reported.

Long-term tracking

In order to improve the robustness of tracking techniques, several works address this issue with a long-term approach, i.e., the computation of the best associations between objects over a large temporal window [Zhang and Gong, 2006] and [Berclaz et al., 2006]. Most of these techniques are based on a graph computation [Taj et al., 2006], [Gatica-Perez, 2005] and [Gomila and Meyer, 2001]. We have to note that this approach does not necessarily replace but complement short-term approaches (i.e., those previously described).

In [Bremond, 26], the authors first compute a set of paths in the temporal graph of objects. These paths represent the possible trajectories that objects may have. Then, the association decision is taken with a delay *T* to allow to study the evolution of all the possible paths. The tracking algorithm chooses at each frame the best path to update the object characteristics, based on a quality factor which is a weighted sum of several criteria, such as the 3D size similarity, the 3D distance between objects. This technique enables to better discriminate people during crossings or to recover temporary occluded persons.

In the same way, in [Khalaf and Intille, 2001], the authors propose to use continuity in space, motion and appearance over time to track persons. Continuity in space refers to the fact that two persons who merge must be the same when they split again, or to the fact that a person cannot disappear without a known explanation (e.g., presence of a door). Continuity in motion refers to the fact that there is a limit to the amount of distance people can travel in a given amount of time. Continuity in appearance refers to the fact that people tend to look more similar to themselves than to others, in average over time. This technique also performs the decision with a predefined delay by using dynamic programming techniques.

5.2 Large variety of real world conditions

After reviewing the various types of videos processing programs, the large variety of real world videos needs to be characterised to get a precise picture of the complexity of video understanding problems. Based on our experience and on previous works on this topic [Ellis, 2002], we propose here a characterization of the variability of video sequences which covers most of the video understanding issues. The scope of this characterization is large but not large enough to be applied directly to specific domains such as the processing of aerial images. In these cases, appropriate criteria should be added to extend this characterization.

Type and position of the sensor

The type of sensor (high resolution, PTZ, CCD, CMOS, infrared, aerial, omni-directional, stereo,...) and the acquisition/storage material are of prime importance because they dictate the processing which can be applied on the corresponding video sequences. For instance, a background subtraction technique to detect motion cannot be directly applied with

a moving camera. We can also have a large field of view camera which introduces distortion or we can have an omni-directional sensor which requires dedicated techniques. CMOS cameras can introduce a grain noise which imposes an additional pre-filtering stage.

We can also underline the effect of the camera position. The distance to the scene can range from a close-up view (e.g., the object height is 200 pixels high) up to a far view (e.g., an object is 2 pixels high). For instance, a far view may induce a higher miss-detection rate. Moreover, a top view camera has a low perspective effect but makes difficult to computate the height of an object. With a side view, the estimation of the 3D parameters is difficult due to the high perspective.

Finally, cameras can deliver compressed images with various compression ratios, a high ratio implying artefact problems such as the blocking artefact in the JPEG protocol. The frame rate and the image resolution have an impact on detection and tracking. For instance, a low frame rate can introduce higher ambiguities in the matching process used during object tracking. A too low resolution may prevent to detect a small object.

Camera and scene motion

Although a system uses fixed cameras, motion perturbations can be induced by two kinds of motion. First, camera vibrations are causing mainly translations between consecutive frames. For example, a camera on a pillar for highway surveillance can be oscillating due to wind. Second, scene motion can be generated by any object in the scene like curtains, trees, escalators or relative motion which occurs when the camera is fixed inside a moving object like a train and a part of its field of view sees the outside of the train. Another example of noise motion is a rainy or snowy weather condition for an outdoor camera. All these motion perturbations can degrade performances of object detection techniques based on motion.

Illumination changes

We can list four main problems caused by illumination changes:

- Slow illumination changes: in outdoor scenes, slow illumination changes can occur due to clouds or solar rotation (e.g., sunset, sunrise). For indoor scenes, it can be caused by intensity variations from the light source (bulb,...). These changes add noise in the object detection algorithm based on motion.
- Fast illumination changes: some image parts are sometimes submitted to fast illumination changes. In an indoor scene for instance, closing blinds can introduce a dark area in the image.
- Automatic gain control of cameras: when a camera is exposed to a strong illumination change (for instance, an object passing close in front of the

camera), its gain is automatically corrected to insure the best image quality. This leads to detect motion in the whole image and prevents to find the objects.

- Visual artefacts: these are perturbation phenomenon like reflections against walls, reflections due to windows or bright floors, reflections in pools of standing water, low target contrast, contrasted shadows (with textured or coloured background),...

Static and dynamic occlusions

Static occlusions, which can be either partial or full, are mainly due to clutter. In a small office for instance, people tend to get close to each other or will be more often occluded by a scene object like a desk or a chair. In such a case, people will probably not be correctly detected, classified and tracked. In fact, the tracking can suffer from variations in the 2D appearance and/or in the estimation of the 3D location of an object. A full static occlusion can be responsible of the loss of a track. Object detection and tracking are thus considerably affected by clutter.

Moreover, the more clutter the more chance to encounter problems in handling contextual objects such as a chair which has been displaced by a human, a newly installed desk,... This adds a lot of complexity in discriminating interesting objects from perturbing ones. Clutter is also synonymous of image complexity. For instance, an airport tarmac, a car park or a road which exhibit a homogeneous colour image can be more easily handled than a textured image coming from a cluttered office.

Concerning dynamic occlusions, there are two potential problems which can arise. First, a more or less long dynamic occlusion can prevent to determine accurately the number of objects present in the scene. For instance, a person occluded by another person for a long time will be hardly tracked by a system, as there are no spatial evidences and very few temporal ones. Second, crossings between objects can generate a swap of identifiers of objects, thus preventing to compute a correct trajectory and other descriptors.

Type, speed and pose of objects

Several problems can arise from the type, the speed or the pose of objects which are present in the scene. Most of video processing programs use a model of objects or a model of movement of objects. The type of objects may influence the frequency of failure of a tracking algorithm because objects can be modelled more or less accurately. For instance, a person can have a totally non-predictable movement (e.g., a person who is making a half turn) compared to a car whose motion can be physically modelled (e.g., known steering lock, known acceleration model). Also, in a scene where objects of different types can occur, the interaction between different types of objects can cause difficulties for a classification process. For instance, a vehicle near (or occluding) a person may prevent the classification

process to correctly classify the person and the vehicle separately but classifying both of them as a compound object.

The speed of objects may also induce more difficulties for a tracking algorithm. Considering a fixed frame rate, an object which is moving fast can be lost by a tracker, if the frame rate is too slow. For instance, it can be a person who begins to run to catch the metro in a metro station. In addition, it is hardly possible to adapt processing parameters for each single data (i.e., to have a different processing for a running person and for a walking one). At the opposite, a stopped object can generate miss detections during the object detection process.

Finally, the posture of people, if not handled, can cause serious problems for detection, classification or tracking procedures. Indeed, common algorithms rely on the constancy of features over time to realize the tracking and to improve their confidence values. This constancy can be broken down in case of sudden variations of postures. Also, a classification process may fail to assign a correct label to a person who is lying on the floor or who is raising his/her hands up, as the adequacy with the model is not good enough anymore.

Camera synchronization and hand-over

The main objective of video understanding is to provide a semantic description of what is happening in a real 3D scene. Depending on the target application, this process may require the use of multiple sensors having overlapping field of view or not. For instance, the event to recognize may involve tracking a person walking first in a corridor and then entering an office. The process of multi-sensor fusion must be able to understand that the perceived information coming from the camera viewing the corridor and the perceived information coming from the camera inside the office are both coming from the same person. This problem is known as the *hand-over problem* and is sometimes not trivial to manage. Indeed, algorithms based on appearance models may suffer from changes in appearance of the same person viewed from two different cameras (e.g., the person is front view by camera 1 and is seen back by camera 2).

A soft synchronization of cameras is usually less accurate than a hard-wired synchronization. However, this is the common configuration in most of existing systems as it is a less expensive and easier solution. A process of multi-sensor fusion may suffer from the variations of timing information during the association phase of perceived information of the same person but coming from different sensors (e.g., a similarity measure falls below a threshold due to the comparison of time-shifted information about the same reality). In consequence, a tracking can lose the track of an object due to the non-association of information coming from the sensors.

Event complexity

The complexity of the events to recognize can pose several problems in terms of processing time. First, events involving a long temporal sequence of states and events may generate a combinatorial explosion if the recognition process is trying every possibility. For instance, a model of the event *a person enters an office and then sits down on a chair* is simpler than a model of the event *a person is vandalizing a ticket machine*. Indeed, the latter event involves several successive states and events like a person A enters a zone 1, the person A moves from a zone 1 to a zone 2, the person A stays for 10 seconds in zone 2, the person moves back to zone 1, the person A moves again from zone 1 to zone 2, etc. Second, events involving several objects are more difficult to handle and may also generate a combinatorial explosion.

5.3 Video Understanding Programs

Based on the state of the art, we have collected and/or developed generic video processing programs, which can be used to process a large variety of real world videos. This section describes these programs which are organised along the main functionalities composing a video understanding process. This section describes first these functionalities and second, the video processing programs used in VSIP platform to build different scene understanding systems.

5.3.1 Video Understanding Functionalities

Object detection and classification

The objective of this task is to detect and classify the physical objects of interest which are present in the scene. This task takes images as input (colour or black and white). These images have been digitized at a variable frame rates and produces a list of labelled physical objects of interest. This task is applied for each camera in case of a multi-camera configuration.

The detection decomposes the image into blobs corresponding to potential physical objects of interest. Advanced functionalities consist of being able to distinguish blobs generated by human activities from those corresponding to noise generated by contextual objects (e.g., moving trees), shadows or reflections. For instance, these advanced functionalities may require the use of contextual information (e.g., 3D geometry of the empty

scene) or chromatic information about pixels. The output of the detection is a grey level image (0 = background, n = identifier of the object of interest).

The classification (possibly including any filtering and splitting/merging process) classifies blobs into labels corresponding to classes of physical objects of interest, with respect to a predefined semantics: person, vehicle, group of persons, etc. Advanced functionalities consist in refining the object classes (e.g., motorcycle, cycle, car, truck, airplane, for the vehicle class), in splitting objects (e.g., two separate persons are better than a group), in computing a posture and an orientation for objects, in computing their 3D parameters while taking into account static occlusions by contextual objects. The output of the classification is a list of physical objects of interest with their properties.

Spatio-temporal analysis

The objective of this task is to maintain a 3D representation of what is happening in the scene over a period of time (e.g., few seconds up to a minute) by resolving potential incoherencies, based on visual invariants in the scene (e.g., a person cannot go through a wall). First, a tracking process matches the objects detected at image time t-1 with those detected at image time t and maintains a unique identifier for each object over the whole video stream coming from the camera. The tracking output is a temporal graph of objects. Nodes of this graph represent physical objects (and their properties) while edges are temporal links over time. The various sequences of edges in this graph represent the various possible trajectories a physical object of interest may have. In case of multiple cameras with overlapping field of view, a fusion operation is performed to obtain a unique graph of objects. Then, this unique graph is analysed over a large temporal window in order to extract temporal properties of physical objects of interest (e.g., the trajectory, the zone of entrance and exit of the object, the speed). Advanced functionalities consist in being able to correctly track objects in a network of cameras with distant field of view using information like the 3D location of objects, in tracking separately rather than globally objects in case of dynamic occlusion. Several alternatives are possible depending on the type of objects being tracked and the complexity of the scene: individual tracking, group tracking, crowd tracking.

Event recognition

The objective of this task is to recognize any event from descriptors given by the preceding tasks (e.g., shape, speed, position and trajectory). An event (e.g., abandoned bag, forbidden zone access, bank attack) is either learnt or predefined by an end-user. An event is characterized by the involved objects, the initial time of the event recognition and its duration. This recognition is performed globally for the scene viewed by one or several cameras. Advanced functionalities consist in being able to handle the uncertainty associated to low-level features in order to maintain a high score of event recognition, or in being able to recognize complex events involving fine descriptors such as the posture of an object.

5.3.2 VSIP Library

This section first gives a description of the library of video processing programs composing the video understanding platform called VSIP which is illustrated on Figure 5.1. This platform which has been described in [Bremond, 9] is made of several techniques. These techniques have been extensively explained in several papers which are given in reference in the text. Here, the objective is to describe how we have decomposed and structured the whole processing chain into programs or typical combinations of programs (and identified their input/output data) by following a general model of video understanding.

The first task is the **detection and classification of physical objects of interest** which are present in the scene. This task takes one image as input (colour or black and white) and produces a list of labelled physical objects of interest. To achieve this task, several distinct steps (including options and alternatives) have been identified:

- *Image acquisition*: this step produces a digitized image coming from a video source. Several alternatives exist: frame grabbing from an acquisition card in case of an analogue camera, image loading from a file which is stored on a hard drive, decompression of a MJPEG live stream coming from an IP camera.
- *Reference image generation*: this step creates a reference image which is used during the segmentation step. There are several ways of computing this image [Bremond, 32]. First, when no a priori knowledge in the form of an image of the empty scene is available, the first image of the video sequence is taken. One drawback of this technique is that any person present at the beginning of the sequence is integrated in the reference image. Second, the reference image is selected among a list of stored images corresponding to the empty scene taken under different illumination conditions. The selected image is the one which is the nearest of a mean image computed over the N first frames of the video sequence, according to a metric based on statistical variables (e.g., standard deviation of intensity). One drawback of this technique is that it is a global approximation process. Thus, a third technique refines this reference image on a local basis but it requires more images (e.g., N equal to 100) and is also more time consuming.
- *Segmentation*: this step detects moving regions by subtracting the current image from the reference image. The resulting difference image is threshold using several criteria based on pixel intensity in order to create moving regions. These moving regions (also called blobs) are associated with a set of 2D features like density or position. Two alternatives exist, depending on whether the chromatic information about pixels is available and usable or not.
- *Chair management*: this step is optional and works on the list of blobs [Bremond, 32]. It may be activated to help differentiating a blob corresponding to a chair from a blob corresponding to a person, by using a priori information

about contextual objects (e.g., by comparing a colour distribution of a blob with a predefined visual template of a chair).

- Door detection: this step is also optional and works on the list of blobs. It allows handling the opening/closing of doors which have been specified in the 3D description of the scene (i.e., a priori knowledge). This algorithm removes from blobs the moving pixels corresponding to a door being opened or closed [Bremond, 9].
- *Classification*: this step takes the list of blobs and produces a list of physical objects of interest. It is composed of three successive sub steps. First, a merge process tries to correct segmentation errors by regrouping small blobs corresponding to the same physical entity (e.g., person, car). Then, a split process is applied on large blobs to verify whether they could correspond to several physical entities. In this situation, the blob is separated into one or several parts, each one corresponding to a physical entity. At this point, the confidence that blobs correspond to physical entities is better. Thus, a set of 3D features like 3D position, width and height are computed for each blob, by using calibration information. By comparing this set of 2D and 3D features with predefined models, these blobs are classified into several predefined classes (e.g., person, group, car, truck, aircraft, unknown, noise,...). A final filtering stage is applied to remove small and unclassified blobs which do not correspond to real entities. At the output of the classification step, the blobs with their associated class label and their set of 3D features are called *physical objects of interest*.
- *3D position correction*: this step corrects the 3D position of physical objects of interest which have been located at a wrong place (such as outside the boundary of the observed scene or behind a wall). This may happen when the bottom part of a person is not correctly detected (e.g., the legs can be occluded by a contextual object or badly segmented).
- *Ghost suppression*: this step aims at removing physical objects of interest which are not due to real objects in the scene (i.e., ghosts). Instead, they are due to stationary objects which have been integrated in the reference image (e.g., a car parked for hours) and which are now moving again but not yet removed from the reference image. For instance, this technique analyses the presence or absence of gradient on the object contour in the current image.
- *Reference image updating*: this step works on the reference image and tries to integrate environment changes appearing in the current image. A simple and fast technique consists in blending slightly each reference pixel with a small coefficient (e.g., 0.01). An alternative consists in discriminating real objects from regions of persistent change in the image (e.g., a new poster on the wall or a newspaper on the table) and to integrate only these regions [Bremond, 32]. This alternative is time consuming and requires computing several features.

Once this first task is performed, the list of physical objects of interest is then processed by the **spatio-temporal analysis** task. This task is also composed of several steps:

- *Frame to frame tracking*: this step is performed for each video sequence in case of a multi-camera configuration (e.g., in sequence or in parallel). The

objective is to link from frame to frame all physical objects of interest [Bremond, 18]. The decision to create a link is based on several criteria such as the amount of overlap, the 3D distance between centres of gravity of objects [Rota and Thonnat, 2000]. The output of this step is a graph containing the detected physical objects of interest updated over time and a set of links between objects detected at time *t* and objects detected at time *t-1*. A physical object of interest with temporal links towards previous ones is called a *tracked physical object of interest*. This graph provides all the possible trajectories of an object.

- Fusion and Synchronization: depending on the camera configuration, this step may be activated. Graphs of physical objects of interest coming from the different cameras with overlapped fields of view are fused together in order to obtain a unique representation. This technique uses combination matrices (combining several compatibility criteria) to establish the good association between the different views of a same object. A physical object of interest detected by a camera may be fused with one or more physical objects of interest seen by other cameras, or can be simply kept alone or destroyed if classified as noise. A temporal synchronization is sometimes necessary when the different cameras are not synchronized by a specific hardware. In such a situation, this step may decide to pause the processing on a camera to wait for the other ones. The output of this step is a *graph of fused tracked physical objects of interest*. They contain all the temporal links of the original objects which have been fused together and their 3D features are the weighted mean of the original 3D features. Weights are computed in function of the distances of original objects from the corresponding camera. In this way, the resulting 3D features are more accurate than original ones.
- *Long-term tracking*: this step works on the (fused) graph of objects. Depending on the events to recognize, several alternatives are activated [Bremond, 25, 26 and 30]. All of them rely on the same idea: they first compute a set of paths (in the graph) representing the possible trajectories of objects to track (e.g., isolated individuals, groups of people, crowd). Then they track the physical objects of interest with a predefined delay T to compare the evolution of the different paths. At each frame, the best path to update the physical object characteristics is chosen.

Once physical objects of interest are tracked over a temporal window and featured in a unique 3D referential, the **event recognition** task is performed. Depending on the type of events to recognize, different alternatives are used:

- For events dealing with uncertainty, Bayesian networks can be used [Bremond, 34].
- For events with a large variety of visual invariants (e.g., fighting), AND/OR trees can be used [Bremond, 31]. Visual invariants are visual features which characterize a given event independently of the scene and of an algorithm. For instance, for a *fighting* event, some visual invariants are an erratic trajectory of a group of people, or one person lying down on the ground or important relative dynamics inside the group.
- For events organised as sequence of events, we can use finite state automata [Bremond 31]. If these events are characterised by uncertainty, HMM (Hidden Markov Models) can be used [Bremond 23].

- Finally, for events involving multiple physical objects of interest and complex temporal relationships, the technique used is based on a constraint network whose nodes correspond to sub-events and whose edges correspond to temporal constraints [Bremond, 28 and 35]. Temporal constraints are propagated inside the network to avoid an exponential combination of the recognized sub-events. For each frame, events are recognized incrementally (i.e., temporal constraints are checked), starting from the simplest ones up to the more complex. This technique uses a declarative language to specify events to recognize. The output of this last task is a list of recognized events.

For efficiency and modularity reasons [Bremond, 9], the exchange of data between programs of the VSIP library is realized through a shared memory. This shared memory is managed by a shared memory manager which is responsible of data management and distribution to programs.

In conclusion, we can see that we have a fairly large amount of distinct programs. In addition, a program can be either elementary or complex (e.g., alternatives or sequences have been identified) and can be either mandatory or optional. As a consequence, a main issue is to model all this variability and flexibility to obtain an efficient scene understanding system. This issue will be discussed in chapter 8.

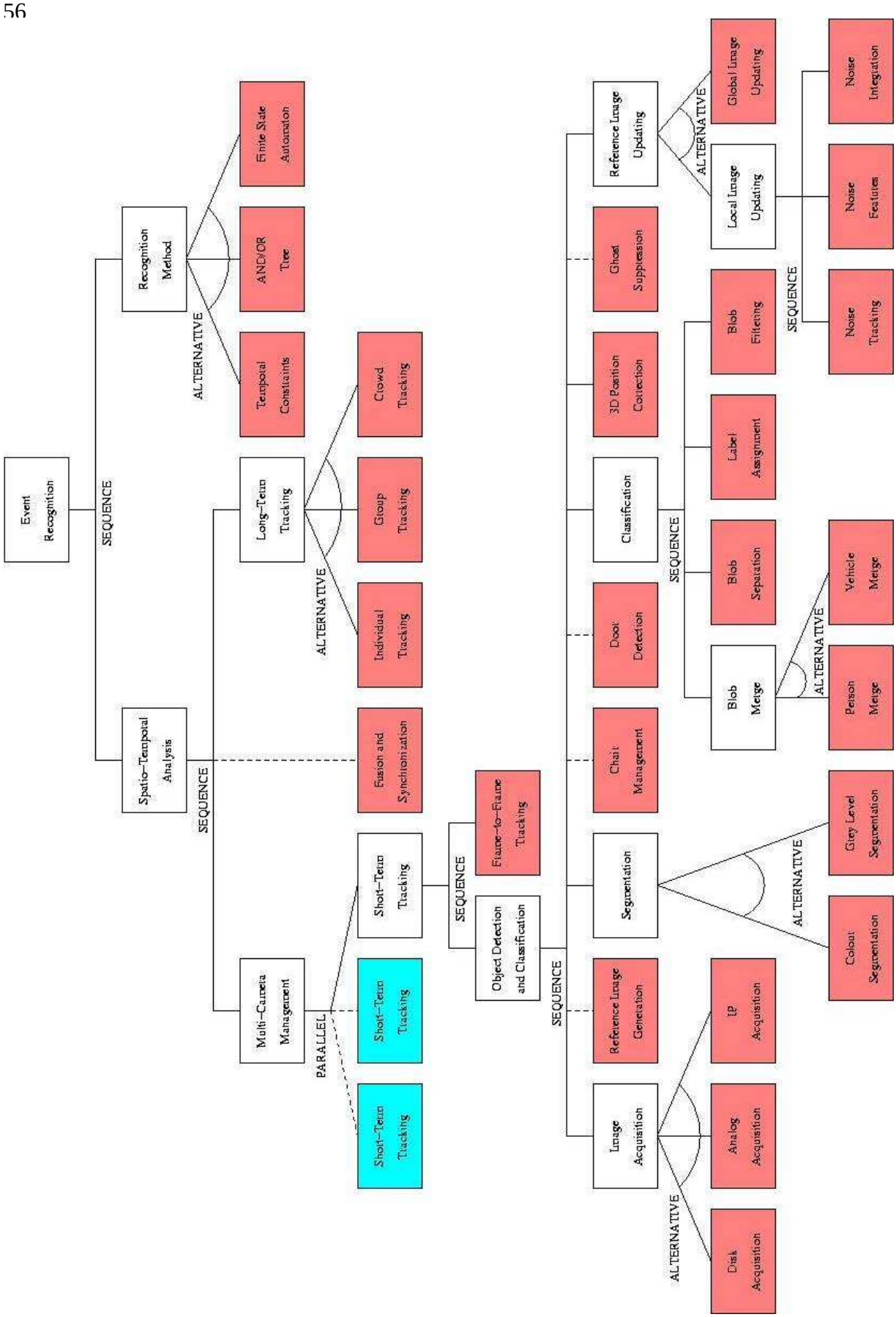


Figure 5.1: *the hierarchy of the library of video processing programs composing the video understanding platform called VSIP. The orange boxes correspond to elementary programs directly executable and white boxes correspond to components composed of elementary programs.*

5.4 Advanced Video Understanding Programs

Advanced video understanding programs are needed either to process specific scenes (e.g. highly cluttered scenes) or to address particular requirements (e.g. people posture recognition). This section describes these two types of works on advanced video understanding programs. The first work has been done in collaboration with Gabriel Davini and Magali Maziere on correcting the position of people detected in a cluttered scene and not complying with 3D geometric constraints. The second work has been done with Bernard Boulay during his phD on posture recognition [Bremond 10, and 36].

5.4.1 A context-aware detection algorithm for occluded objects

In many video interpretation applications, mobile objects may be partially occluded and thus not visible in their wholeness. As a consequence, when computing their 3D position and dimensions, if only the visible part is taken into account, errors may be committed. Moreover, standard appearance-based context-unaware classification algorithms may fail, assigning the objects with wrong types or even considering the objects as noise.

We can identify three types of occlusion:

1. the occlusions due to the vision field of camera, which occur when the mobile object is on the border of the vision field of camera;
2. the occlusions due to contextual objects (e.g. a chair), that occur when one or more contextual objects are placed between the mobile object and the camera;
3. the occlusions due to (other) mobile objects.

Occlusions may be static or dynamic, depending on whether the occluded regions change in time. Usually, the first two types of occlusion are static, since the camera and contextual objects do not move; even if, there are some exceptions as, for example, the doors, the windows or the wheelchairs. On the contrary, the third type of occlusion is typically dynamic.

In order to cope with static problems of occlusion, we propose an algorithm which exploits the *a priori* knowledge of the vision field of camera and of contextual objects so as to estimate the occluded parts of analyzed objects. The second types of occlusion in case of dynamic problems are left to specific algorithms dedicated to particular contextual occluding objects, while the third type of occlusion (including a mixture of previous types) is left to the tracking module.

The proposed algorithm is composed of two parts: the first one deals with the first type of occlusion, the second one deals with the second type in case of static occlusions. Both sections start with a series of tests that are run to establish whether the mobile object is occluded and where the mobile object may have a hidden part. For example, if a person is sitting behind an armchair and the visible part is just his/her head, we may conclude that there is an occlusion due to the armchair and then a hidden part may be found below the head.

At this point, in order to proceed with the analysis, we check whether the visible part of the object is compatible with a mobile object model (e.g. human being). This means that we test whether the mobile object 3D dimensions correspond to those of a mobile object, as specified by its model. For example, if a mobile object is found to be occluded on its left side, we first verify whether its height correspond to the common height of the considered model object, since the occlusion on object left side does not affect its height. Then, we check whether its width is reasonably close to the minimum mobile object width. Exceptions to this rule are made on a per model basis, for some specific situations, as when the visible part is a head that is not compatible with the person standard width.

Since the visible part of an object may comply with more than one mobile object model, the preferred model is the one with the highest compatibility.

In the case of the first type of occlusion, an occluded mobile object, which complies for at least with one 3D dimension (width or height or length) of a mobile object model, is corrected so as the two other dimensions become the standard 3D dimensions of the model; eventually, its 3D position may also be recomputed. At this point, a new classification stage is run so as to assign the object with a type, which corresponds to the considered mobile object model or corresponds to a compatible model with the latter.

In the case of the second type of occlusion, static contextual objects are first scanned in order to find those that partially occlude the mobile object; also, this process estimates the maximum space where the mobile object can be located behind. In the following, a number of hypotheses about the analyzed object 3D position and dimensions are made, till the first of them succeeds in complying with the considered mobile object model. At this point, as before, the analyzed object is modified and reclassified.

The proposed algorithm has been successfully tested on video sequences with different scenarios; an example of its application is provided in the pictures in 5.2.

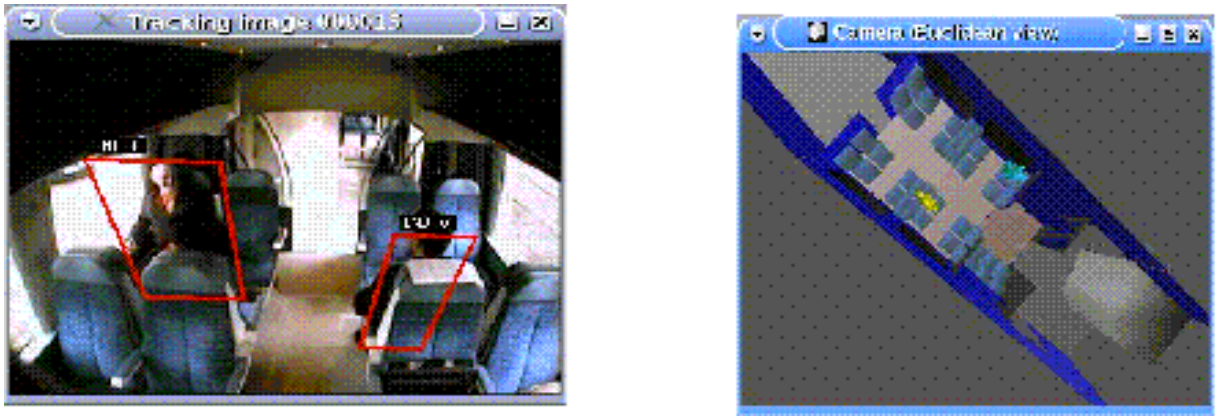


Figure 5.2: *Two occluded objects correctly recognized and positioned.*

5.4.2 Human Posture Recognition

This section describes an algorithm to recognise the posture of people evolving in a scene observed by only one camera. We have proposed an original work on human posture recognition by combining a classical 2D approach and use of a 3D human model. The 3D model (cf. Figure 5.3) is defined by 9 degrees of freedom (articulations): the abdomen, shoulders, elbows, hips and knees. We first define for each posture of interest a specific set of parameters. These parameters are the three Euler angles for each articulation.

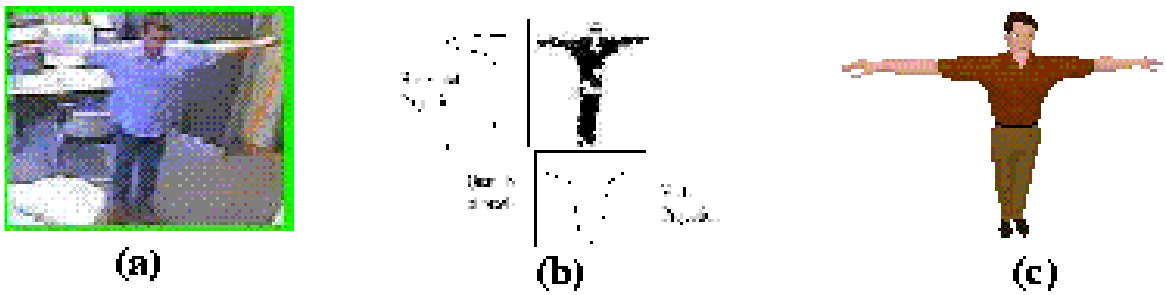


Figure 5.3: *The image (a) depicts a man in T-shape posture. In image (b) we can see the corresponding blob and its horizontal and vertical projections. Image (c) represents the 3D human model in T-shape posture selected by the algorithm.*

For each detected person, the video understanding platform (VSIP) provides us with the position of this person in the 3D space. Then we compare the silhouette of the detected person with reference silhouettes generated from the 3D model. We project the 3D models on the image plan for reference postures (called reference images), which have been generated using the 3D position of the person. To determine the orientation of 3D model, we test all possible orientations using a rotation step. Then, we compare the horizontal and vertical (H. & V.) projections of the reference images with the (H. & V.) projections of the detected person silhouette. The horizontal (resp. vertical) projection on the reference axis is obtained by counting the quantity of motion pixels which correspond to the detected person for each row (resp. column). An example is shown in Figure 5.3 for the T-shape posture.

A first challenge is to tune the parameters of the posture recognition algorithm to obtain better results. The most important parameter is the rotation step. To determine the optimal rotation step, the recognition rate of postures on synthetic data is computed for different rotation step values. For this experiment, we have used a 3D woman model (cf. Figure 5.4*Figure*) which is different of the 3D model used for recognition. We choose a 36 degrees step because it gives a better ratio between recognition rate (76 %) and computation time (1.5 images per second). We have performed tests on synthetic data by giving the correct orientation which shows that computation time is divided by 10, and recognition rate increase from 76% to 95%.

A second challenge is to evaluate the posture recognition approach. We obtain good results on real videos of about 2000 frames depicting targeted postures (Table 5.1): 89% of correct recognition for standing postures and 88% for sitting-bending postures. A more accurate analysis gives 100% of correct recognition for standing posture with arms up (left

and right arms are not discriminated), 62% for standing posture with arms near the body, 87% for T-shape posture, 75% for sitting postures, and only 40% for bending posture.

Standing Postures: 89%						Sitting-Bending Postures: 88%					
Standing posture with arms up: 100%		Standing postures with arms near the body: 62%		T-Shape postures: 87%		Sitting on chair postures: 54%		Sitting on the floor postures: 57%		Bending postures: 40%	
na:100	a:100	na:83	a:41	na:93	a:0	na:61	a:40	na:71	a:13	na:70	a:24

Table 5.1: *Recognition rate of postures organized in a hierarchical way depending on the scene difficulties (a: ambiguous case, na: no-ambiguous case).*

We define an ambiguous case as a situation where the silhouettes of a person in two different postures are very similar (cf. Table 5.1). We determine ambiguous case by using synthetic data. A synthetic video is made for each posture and for all possible orientations. Then, an ambiguous case is found when the posture is not correctly detected by the algorithm. For example, a no-ambiguous case of the bending posture corresponds to the situation where the camera observes the person from the side. By discriminating ambiguous cases, we obtain an average of recognition rate of 70% for non ambiguous cases and only 24% for ambiguous cases on real videos.

Figure 5.4: *3D woman model used for experiments with synthetic data*

About 35% of wrong recognitions are due to the ambiguity problem, and the other 65% to segmentation problems as shown in Figure 5.5. To improve the algorithm we plan to

compute features from the detected person (orientation, 3D height, ...) to compute more appropriate 3D model. We also want to use information provided by the tracking phase to improve results by taking advantages of the temporal coherency.

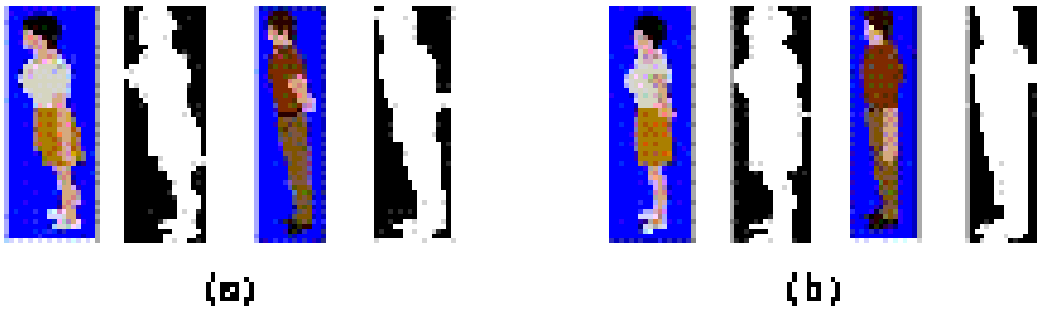


Figure 5.5: *(a) Standing posture with arms along the body recognized as T-shape posture,(b) T-shape posture recognized as standing posture with arms along the body.*

5.5 Conclusion: Advantages and limitations of VSIP platform

In this chapter, we have described the large variety of video processing programs and of videos depicting a scene. We have explained how VSIP platform can help building scene understanding systems, based on two types of programs: (1) generic programs for common video characteristics, (2) advanced programs for handling particular situations. Nevertheless, object detection and extracting perceptual features will stay an open issue for still a long period of time, in particular in real world situations, such as moving cameras, crowd, and limited processing capacities.

The current trend is to establish under which hypotheses the algorithms are valid, and to understand their limits. Based on this result characterisation, the other four research axes proposed in the introduction can be fruitfully explored. Moreover, we believe that pursuing these axes is important for two reasons: first, in some specific conditions (e.g. structured environment), these challenges can be solved and second, some applications do not require perfect vision results (i.e. perfect object detection and tracking). In the next chapters, we will discuss the other research axes.

Chapter 6

Maintaining 3D coherency throughout time

This chapter studies the mechanisms to maintain the 3D coherency along time to bridge the gap between the signal and semantic levels. These studies are done along with two directions. First, we explore the different ways of tracking mobile objects through videos, taking advantage of the temporal coherency. Second, we address the issue of information fusion. Despites all the works done in these domains within the last 20 years, fusion and tracking algorithms remain brittle. To guarantee the coherency of tracked objects, spatio-temporal reasoning is required. In this chapter, the first section reviews briefly three stages to insure temporal coherency: frame to frame tracking, long term tracking and global tracking. The frame to frame tracking computes the links between mobile objects from one frame to the next one. The long term tracking explores the set of possible paths within a temporal window of few frames, typically 10 frames [Bremond 25, 26 and 30]. The global tracking optimises globally these paths among all potentially mobile objects [Bremond 12 and 49]. The main challenge of these tracker is not to loose the track of any mobile object, even in case of noise (e.g. change of lighting conditions), motion of contextual objects (e.g. a door), dynamic occlusions (e.g. people crossing) and large density of mobile objects (e.g. crowd). These trackers have been validated through several European projects such as ADVISOR [Advisor, 2003] and AVITRACK [Avitrack, 2006].

The second section presents different mechanisms for information fusion. There are two types of information fusion: (1) fusion between uni-modal information coming from a network of sensors of the same type [Bremond, 30] and (2) fusion of multi-modal information

coming from heterogeneous sensors, such as video and audio sensors [Bremond, 51]. However, the information fusion issue remains the same for both types; finding a common formalism for information representation which is precise enough to model information uncertainty. Information fusion can be done at three levels: perceptual world, physical world and semantic world. At the level of perceptual world, perceptual features are combined through statistic and probabilistic techniques, such as Principal Component Analysis (PCA) and Bayesian network [Bremond, 43]. The goal is to filter outliers, normalise the features and to reduce the dimension of the features space (to reduce redundancy). At the physical world level, information is combined using invariants corresponding to spatio-temporal constraints (e.g. to handle statistic occlusions), and rules modelling dedicated knowledge (e.g. a physical object cannot disappear in the middle of the scene) [Bremond, 12 and 30]. At the semantic world level, events are combined following scripts which are usually pre-defined by end-users. This combination can be done by different formalisms such as scenarios [Bremond 35] or chronicles [Ghallab, 1996].

The main benefit of using information fusion is to bridge the gap between signal level and semantic level. The main challenge in video processing is the bad quality of data which is often erroneous, corrupted, incomplete, partial, missing and vague. To cope with these issues, we need redundant information to be able to perform the abstraction process. Another question that we need to answer is at which level this information should be combined. Information fusion at the signal level can provide more precise information, but information fusion at higher levels is more reliable and easier to realise.

To solve these issues, there are three main rules for multi sensors information fusion:

- Utilization of a 3D scene representation for combining heterogeneous and uncertain information
- When the information is reliable the combination should be at the lowest level to get the better precision
- When the information is uncertain or on different objects, the combination should be computed at the highest level (semantic) to get better abstraction

During the past few years, we have been involved with several research works dealing with information fusion at the three levels of processing, using different formalisms for information representation. Here, we review briefly these works.

In video processing a common way of having several types of information on a scene is to use different types of process for extracting information. For instance, a process can detect and classify mobile objects in video streams, whereas another one detects specific objects such as human faces or detect specific areas which have consistent colour properties. In the SAMSIT project on video surveillance onboard train [Bremond, 51], we have combined these different types of process to handle the train motion and strong shadows cast all over the scene. In this project, we have also combined video and audio analysis to disambiguate some scenarios. For instance, to detect violence scenes we have combined shouting events detected by microphones with strong motions detected by video cameras.

In a control access application, during the PhD of Binh Bui [Bremond, 43], we have combined optical cells, cameras observing the entrance from the side and a camera observing the entrance from the top. We have combined all these information in a 3D representation of the entrance. We were able to distinguish all types of people and objects coming through the entrance such as adult, kids, stroller, and luggage.

In bank agency video surveillance application, during the PhD of Van Thinh Vu [Vu, 2004], [Bremond, 35], and works done by Magali Maziere [Bremond, 38], we have combined different contact sensors and cameras (visible and infrared) to better understand activities in the bank. Thanks to the sensors we were able to detect precisely when the door of the room and the door of the safes were open.

In ambiance intelligent applications on helping to keep elderly at home, we are planning to combine physiological sensors installed on a person, contact and pressure sensors installed in an apartment and cameras to understand the daily activities of person living in the apartment. Each sensor is dedicated to the detection of specific events which are combined through the video analysis.

In the ADVISOR project on metro surveillance [Bremond, 39] and in the AVITRACK project [Bremond, 12] on apron surveillance, we have combined several cameras observing a scene. In the AVITRACK project, we have combined 8 cameras observing all the servicing operations occurring around a parked aircraft. Thanks to the 8 cameras, we were able to observe all activities occurring on both sides of the aircraft despite the occlusions of vehicles operating around the aircraft.

In this chapter, we cannot describe all the works done in maintaining 3D coherency throughout time. Therefore, the first section presents different types of algorithms for tracking objects of interest throughout the videos. The second section on information fusion focuses more on the following two experiments: (1) multi-camera fusion based on the computation in the physical world of mobile object correspondences and (2) multi-camera fusion combined with optical cells based on perceptual feature matching through a Bayesian network.

6.1 Temporal coherency in the 3D space

This section presents an approach for tracking either isolated individual, group of people, crowd or vehicles in the context of visual surveillance of metro and airport scenes. In this context, tracking algorithms are composed of three tasks: (a) motion detection and frame to frame tracking, (b) long term tracking of individuals, groups of people, crowd and vehicles evolving in the scene and (c) global tracking for copping with any remaining errors. This work has been performed mostly in the framework of the European projects ADVISOR and AVITRACK [Advisor, 2003], [Avitrack, 2006]. To reach this goal, we have developed a system which takes as input video streams coming from one or several cameras and tracks all mobile objects in the video streams. The section is organised as follows: in the first part, we present briefly the global system and then the tracking algorithms.

6.1.1 Overall System Overview

As describe in chapter 5, the video interpretation system is based on the co-operation of a vision and a behaviour recognition module as shown on Figure 6.1.

The vision module is mainly composed of three tasks. First a motion detector and a frame to frame tracker generates a graph of mobile objects for each calibrated camera. Second, a combination mechanism 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, groups of people and crowd evolving in the scene (typically on hundreds of frames). The combination mechanism and the behaviour recognition algorithm are presented further in the next sections. On top of that, we use 3D scene models, one for each camera, as a priori contextual knowledge of the observed scene. We define in a scene model the 3D positions and dimensions of the static scene objects (e.g. a bench, 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 behaviour recognition process.

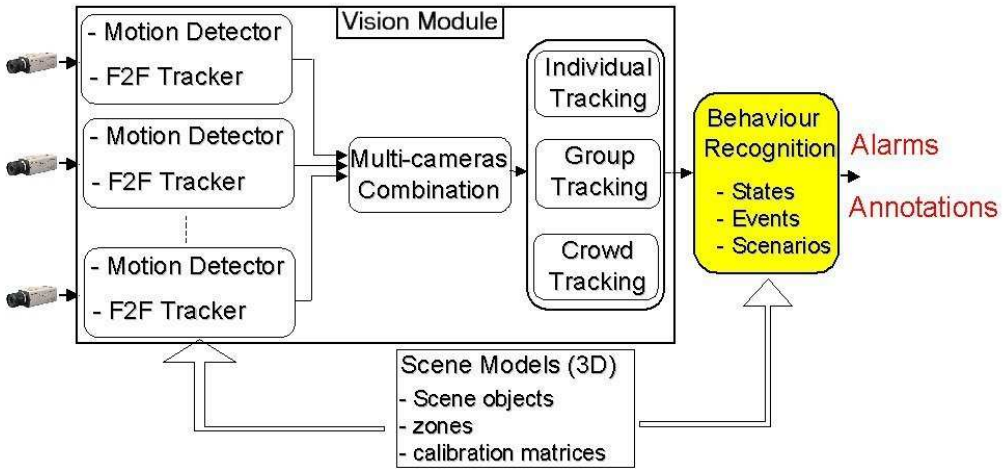


Figure 6.1: *Video interpretation system*

6.1.2 Motion Detector and Frame to Frame Tracking

The goal of the Motion Detector is to detect for each frame the moving regions in the scene and classify them into a list of mobile objects with labels corresponding to their type based on their 3D size, such as PERSON. This task can be divided into three sub-tasks: detection of mobile objects, extraction of features, classification of mobile objects. A list of mobile objects is obtained at each frame. Each mobile object is described by 3D numerical parameters (center of gravity, position ,height, width,...) and by a semantic class (PERSON, OCCLUDED PERSON, GROUP, CROWD, METRO TRAIN, SCENE OBJECT, NOISE or UNKNOWN).

The goal of the frame to frame tracker (F2F Tracker) is to link from frame to frame the list of mobile objects computed by the motion detector. The output of the frame to frame tracker is a graph of mobile objects. This graph provides all the possible trajectories that a mobile object may have. The link between a new mobile object and an old one is computed

depending on three criteria: the similitude between their semantic classes, their 2D (in the image) and their 3D (in the real world) distance.

6.1.3 Individual, Group of people and Crowd Long Term Tracking

The goal here is to follow on a long period of time either Individuals, Groups of people or Crowd to allow the scenarios involving these three different types of actors to be recognised. For example, when we want to detect a group of people (at least two persons) which is blocking an exit zone, we prefer reasoning with the Group Tracker because it provides a more accurate 3D location of the group of people in the scene.

The Individual Tracker tracks each person individually whereas the Group Tracker tracks globally all the persons belonging to the same group. Both trackers perform a temporal analysis of the Combined Graph. The Individual Tracker computes and selects the trajectories of mobile objects which can correspond to a real person thanks to an explicit model of person trajectory. In a similar way, the Group Tracker computes and selects the trajectories of mobile objects which can correspond to the persons inside a real group thanks to an explicit model of the trajectories of people inside a group.

Individual and Group Trackers are running in parallel. When the density (computed over a temporal window) of detected mobile objects becomes too high (typically if the mobile objects overlap more than 2/3 of the image), we stop these two trackers because in such a situation, they cannot give reliable results. At this point, we trigger the Crowd Tracker which is in fact the Group Tracker with an extended model of the trajectories of people inside a group allowing a large density of detected people belonging to the same group that by this way defines a crowd. These trackers are described in more detail in [Bremond 25 and 36].

6.1.4 Global Tracking

As shown in Figure 6.1, the VSIP platform is based on two main modules: one dedicated to vision and tracking, and one dedicated to scene understanding. While the scene understanding module needs accurate input data to recognize behaviors, some incorrectly tracked objects are sometimes present in the vision and tracking output. We thus have developed a Global Tracking module which takes as input the a priori knowledge of the observed environment (static and dynamic context) and the tracked objects and sends corrected tracked objects to the scene understanding module. The correction of the tracking objects is guided by user defined rules.

The Global Tracker is a generic prototype based on rules which allows to quickly and easily adaptable to a new application. Each incorrect tracking result is analyzed and a rule is used to correct it. This rule can be a completely new rule, or a default rule (defined for another situation or application) overloaded with a new set of parameters.

For the Airport Apron Monitoring European project (AVITRACK), we have so far solved two main problems adding new rules in the Global Tracking:

- The first is the loss of tracked objects which occurs when a vehicle stays for a long time at the same place. The defined rule keeps track of the vehicles when they are lost and parked in a specific zone, and ends when the vehicles restart (the newly detected vehicles are linked with the parked ones) or after a predefined period of time (the parked vehicles are removed);
- The second problem appears when a vehicle is detected as several smaller objects (over detection case, as shown on Figure 6.2-a). It occurs when a vehicle is slowly manoeuvring in a small area. The correcting rule merges the mobile objects (as shown in Figure 6.2-b) into the real vehicle. To achieve that, the rule takes into account the predefined vehicle models, and the 3D position and motion of the mobile objects.

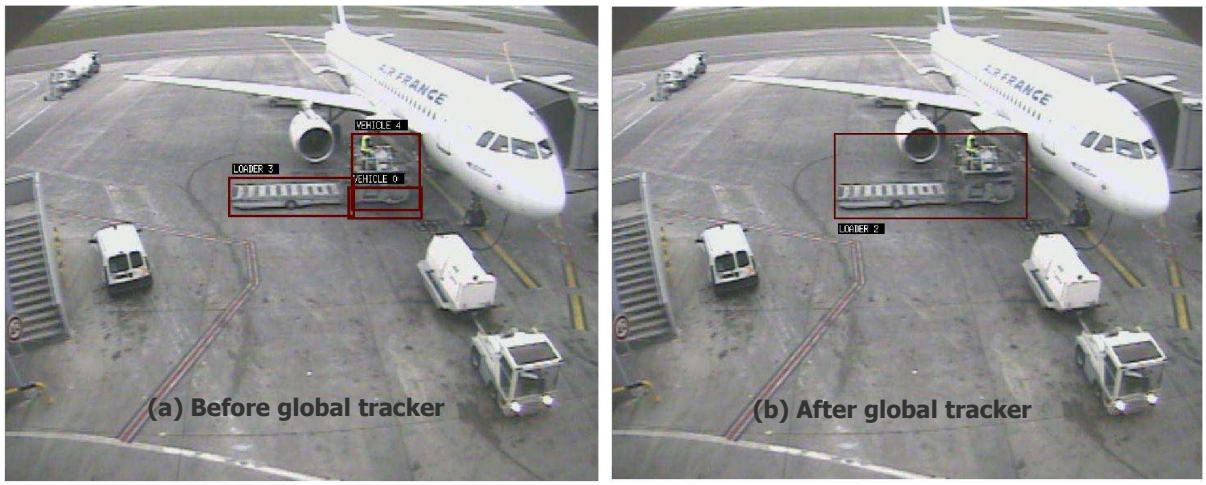


Figure 6.2: *On the left, a Loader vehicle is detected as several mobile objects (over detection) while on the right, the Loader vehicle is correctly tracked as one mobile object.*

The developed Global Tracker has been used in different projects which use the VSIP platform, such as the AVITRACK European project and CAssiopee on bank agency surveillance. This tracker is described in more detail in [Bremond 12].

6.2 Multiple cameras Combination

In this section, we present an approach for tracking people in a cluttered scene (typically a metro scene) viewed by several cameras with overlapping FOVs (Field Of View). The ambitious goal here is to process real world video streams coming from metro stations taken for the European project ADVISOR and to combine information from multiple cameras. The solution we propose is composed of three steps. First a combination mechanism is performed to combine the graphs computed for each camera into a global one. Second, this global graph is used for long term tracking of groups of people evolving in the scene (typically on hundreds of frames). Finally, the results of group tracking are used for recognizing predefined scenarios corresponding to specific group behaviors. In this section, we focus on the graph combination.

6.2.1 Combination of moving region graphs for multiple cameras

The goal of the combination phase is to combine all the graphs of moving regions computed by the Frame to Frame tracker (F2F Tracker) for each camera into a global one that we called the Combined Graph. As we mentioned in the previous section (F2F Tracker), a graph of moving regions is built frame by frame by computing links between the new moving regions detected and the old ones. To combine all these graphs, we combine together the new moving regions detected for all cameras. Each new moving region detected for each camera can be involved in one of the three following types of combination:

- **Fusion:** the moving region is fused with one or several moving regions detected by other cameras. If the labels of all the moving regions are the same, we fuse them into one moving region by computing an average of their 3D features. By making such a fusion, the estimation of the 3D features (position, width, height) of the resulting moving region is improved (closer to the reality). If the label of the moving regions is different, we select for the Combined Graph the one which is the closest to the model of person.

- **Selection:** the moving region is not fused with other moving regions and is selected for the Combined Graph. This moving region corresponds to a mobile object observed by one camera without any correspondence with other cameras.

- **Removing:** the moving region is eliminated and is not present in the Combined Graph. This moving region corresponds to a noise just detected by one camera.

When the combination is done, the temporal links of the Combined Graph are inferred from the links computed for each individual graph. In the case of **Fusion**, if $\{q^t_1,\ldots,q^t_n\}$ is the set of *moving regions* which are fused together into the *moving region* l^t at time t, l^t is linked with the set of combined *moving regions* $\{l_1^{t-1},\ldots,1_p^{t-1}\}$ which corresponds to the result of the combination of all the parents of the *moving regions* at time t – 1. For the case of **Selection**, the process is similar with the difference that the *moving region* l^t results from the selection of only one *moving region* q_1^t. In the case of **Removing**, no link is computed because l^t does not exist. The main step in the graph combination phase is the combination of the *moving regions* newly detected by each camera. To manage this task, we use an iterative approach. If there are N cameras, we first combine the *moving regions* detected by the two first cameras, and then the result is combined with the moving regions detected by the third camera and so on up to the last camera. Thus, we reduce the algorithm to a two dimension space in which we combine N-1 times information about 2 cameras.

6.2.2 Computation of the combination matrix

We compute a n * m combination matrix to compare the correspondences between the *moving regions* $R = \{r_1\ldots r_m\}$ detected by camera C1 and the *moving regions* $O = \{o_1\ldots o_n\}$ detected by camera C2. We have 3 different criteria $\{k_1, k_2, k_3\}$ for doing this. For each criterion k_k, we construct a n * m Matrix N_k (n and m are respectively the number of *moving regions* detected for cameras C2 and C1). The element (i,j) of N_k is in the range [0, 100] where zero indicates that there is no correspondence between the two *moving regions* r_j and o_i and 100 characterizes the strongest correspondence computed for criteria K_k. The final matrix F, called the Combination Matrix, is found as the weighted sum of the matrices N_k

$$F = \sum_{k=1}^{3} w_k N_k$$

where the three global weights W_k are parameters of the combination algorithm. The three criteria are :

- Position criteria: It estimates the spatial coherence between two moving regions o_i and r_j by computing their 3D proximity. This proximity is equal to 100 if both *moving regions* have exactly the same 3D position and decreases exponentially up to zero when the 3D distance between the two *moving regions* is above a threshold T (e.g. 5m).

- Dimension criteria: It estimates the dimension (height and width) similitude between two *moving regions* o_i and r_j. The moving region labels are compared thanks to a class correspondence table. This comparison returns 100 if both *moving regions* have the same label and returns values which decrease up to zero when the labels indicates two classes of object very different (e.g. NOISE and GROUP).

- Temporal criteria: It reinforces the correspondence between two *moving regions* o_i and r_j when one parent of o_i and one parent of r_j have been fused together. This criteria returns 100 when the parents have been fused together, otherwise it returns 0.

We consider that two *moving regions* o_i and r_j have a high correspondence if the element (i, j) of the Combination Matrix F is higher than a threshold T, where T is also a parameter of the combination algorithm (we use T = 70). Then we use a set of rules to decide which type of combination to apply (see next section).

6.2.3 Extraction of sub combination matrices

We extract in the Combination Matrix F, disjoint sub matrices called F_s which correspond to a set of moving regions having correspondences together. Each sub matrix is constituted of two *moving regions* sub sets: one called R_s is a sub set of R and the other called O_s is a sub set of O. These sub sets are the maximum sub sets of neighbour moving regions for which each *moving region* of R_s has a high correspondence with at least one *moving region* of O_s. A *moving region* of R_s is a neighbour of another *moving region* of R_s if they have at least one high correspondence in common with one *moving region* of O_s.

Combination rules

This section describes the rules that we use for combining the *moving regions* $\{r_1 \ldots r_m\}$ and $\{o_1 \ldots o_n\}$ detected for two cameras C1 and respectively C2 by using the Combination Matrix F and the sub matrices F_s.

No correspondence

First, we process the *moving regions* which do not have any high correspondence in the Combination Matrix F. In that case, a *moving region* detected for camera C1 or C2 does not have any high correspondences with a *moving region* detected for the other camera. Two possible situations in the real 3D scene can generate such a case.

- A mobile object (in the real world) is in the field of view of only one camera, consequently, this mobile object is detected by this camera but not by the others (e.g. a person hidden by a pillar will be only detected by one of the two cameras).

- A mobile object is in the field of view of both cameras, but it corresponds to a noise (e.g. a reflection on the floor). Consequently, it is possible that only one of the two cameras has suitable features (position, direction, ...) to detect this mobile object.

When we process a *moving region* without high correspondences, we first check whether it corresponds to a mobile object which is in the field of view of both cameras. If it is the case and if the dimension of the *moving region* is too small (typically a moving region labels UNKNOWN or NOISE), this *moving region* is **removed**. Otherwise, the *moving region* is **selected** for the Combined Graph.

Fusion in case of good correspondence

Second, we process the combination of *moving regions* having good correspondences. In that case, the size of the sub matrix F_s is 1 * 1. Such a dimension means that there are one or several mobile objects (in the real world) which are detected as only one *moving region* for each camera (one moving region r_j detected for camera C1 and one *moving region* o_i detected for camera C2). The two *moving regions* r_j and o_i are **fused** together (see the definition of fusion in Section 3.1).

Combination in case of ambiguity

Third, we process the combination of sub sets of *moving regions* R_s and O_s which have ambiguous correspondences.

Noise removing

This corresponds to a pre-filtering task which is applied on every rectangle n * m ($n \neq m$) sub matrices in order to eliminate the *moving regions* that correspond to noise even if they have one or several high correspondences. This situation can occur when a person is detected by the two cameras C1 and C2 whereas a noise is detected close to this person only by one of the two cameras. In that case, the *moving region* corresponding to the noise could have a high correspondence with the one corresponding to the person. The approach used to solve this ambiguity is similar to the one used in the case of No Correspondence (see previous section). We identify in the sub set (R_s or O_s), with the largest number of elements, the *moving regions* too small (labelled UNKNOWN or NOISE) that belong to the FOV of both camera C1 and C2. As the probability that such *moving regions* correspond to a noise is big enough, these *moving regions* are **removed**. Thus, the size of the sub matrix is reduced and the sub matrix can be processed by the other combination rules.

Correct but noisy correspondence

This corresponds to square sub matrices n * n (n > 1). Such a sub matrix implies that the number of mobile objects detected by each camera C1 and C2 is the same but their correspondence is ambiguous. We distinguish two cases:

The first case corresponds to a 2 * 2 sub matrix F_s. Two fusion combinations between the two sub sets $R_s = \{r_{s,1}; r_{s,2}\}$ and $O_s = \{o_{s,1};o_{s,2}\}$ are possible. Either we fuse the *moving regions* associated to the main diagonal of F_s ($r_{s,1}$ is fused with $o_{s,1}$ and $r_{s,2}$ with $o_{s,2}$) or we fuse the *moving regions* associated to the second diagonal of F_s ($r_{s,1}$ is fused with $o_{s,2}$ and $r_{s,2}$ with $o_{s,1}$). We perform the **fusion** of the moving regions associated to the diagonal (main diagonal or second diagonal) for which the sum of the moving region correspondences is the biggest and is higher than a threshold ($T_d = 90$). If the *moving regions* on the diagonals have not high enough correspondences, we **select** for the Combined Graph the *moving regions* of the camera which have the largest 3D size to keep a maximum of information.

The second case corresponds to a n * n (n > 2) sub matrix F_s. In a theoretical point of view, we should compare all the possible fusion combinations (= n!). However, in order to ease the algorithm scheme and to save time processing, we do not explore and compare all the possible fusion combinations but we rather try to reduce the size of the sub matrix by achieving the fusion between the two moving regions that have the highest correspondence, if this correspondence is high enough (above a threshold $T_s = 90$). If such a correspondence exists, the two corresponding *moving regions* are **fused** together. The size of F_s is reduced and F_s is re-processed by the combination rules. On the other hand, if the highest correspondence is not high enough (<90), we **select** for the Combined Graph the *moving regions* of the camera which have the largest 3D size to keep a maximum of information.

Incorrect and noisy correspondence

This last case corresponds to rectangle n * m ($n \neq m$) sub matrices F_s and is managed similarly to the previous case (n * n, n > 2 sub matrix F_s). We fuse together the two *moving regions* that have the highest correspondence if this correspondence is high enough (> 90). The size of F_s is reduced and F_s is re-processed by the combination rules. If the highest correspondence is not high enough, we **select** for the Combined Graph the set of moving regions of the camera which have the largest number of elements to maximize the number of isolated mobile objects (see Figure 6.3).

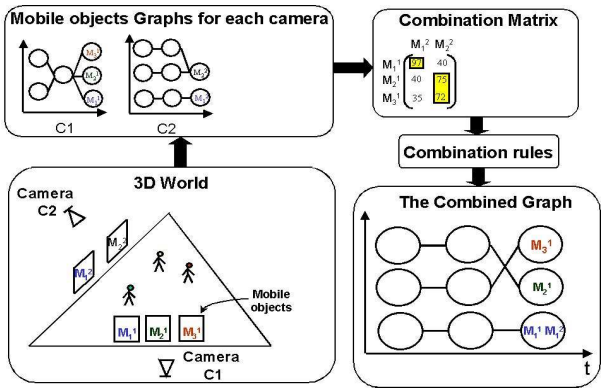


Figure 6.3: *This figure illustrates the multiple cameras combination process. Three persons are evolving in the scene. Camera C1 detects three mobile objects whereas camera C2 detects only two mobile objects. The combination matrix enables to determine (a) a high correspondence between the mobile object M11 of C1 and the mobile object M12 of C2; these two mobile objects are fused together in the combined graph, and (b) an ambiguous correspondence between the two mobile objects M21 and M31 of C1 and the mobile object M22 of C2; the two mobile objects M21 and M31 detected by C1 are selected in the combined graph.*

6.2.5 Graph combination results

This fusion module has been tested on several metro sequences. In this section, we are showing images of the processed videos. In these images, a red box corresponds to a moving region classified as a PERSON and a green box corresponds to a moving region classified as a GROUP. On Figure 6.4, the two top images show the moving regions detected for each camera and the two bottom images show the projection for each camera of the moving regions resulting from the combination process. This figure illustrates the case where the combination process select moving regions from one camera because in this camera the moving regions are corresponding to different individuals who are separated.

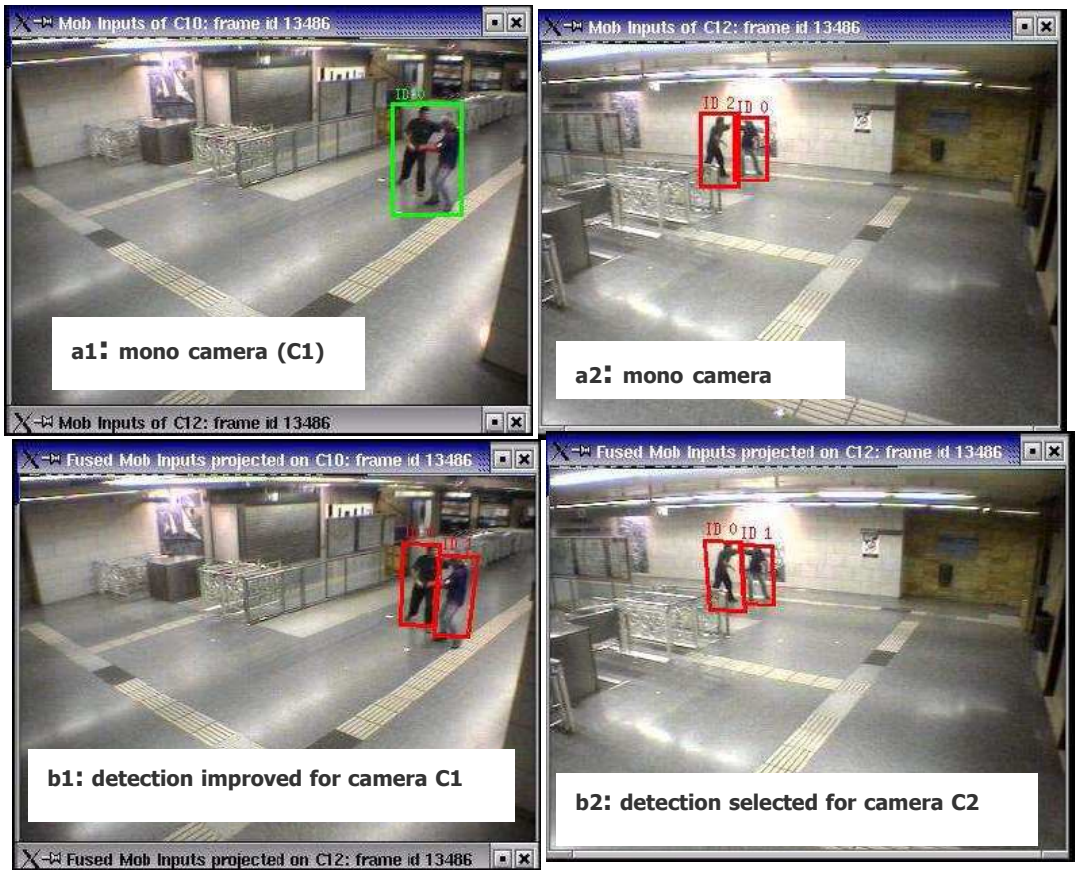


Figure 6.4: *Two persons are detected as one moving region by the camera C1 (a1) whereas they are detected as two separated moving regions by the camera C2 (a2). A 2*1 sub combination matrix Fs is constituted (case: Incorrect and noisy correspondence). As there are not high enough correspondences in F_s and as there are more moving regions detected by camera C2, the moving regions of camera C2 are selected. Thus the combination process has improved the detection of camera C1 (b1): the two persons are now detected as two separated moving regions.*

Currently, the main limitation of the combination module is an over estimation of the number of persons. This can occur when the same person is detected as one moving region (labelled as a PERSON) from two cameras and one of these moving regions is not detected in the field of view of the other camera due to 3D position errors. In that case, the combination process cannot make the correspondence between the moving regions and decide to keep both moving regions (corresponding to the same individual in different cameras) in the Combined Graph.

6.3 Object Recognition Based on a Multi-Sensor System

In this section, we propose a new system for shape recognition based on a video and multi-sensor system in order to classify mobile objects. Our goal is to design a system with very high recognition rate complying with real-time constraint. To achieve this goal, we have conceived a device combining a static camera and a set of lateral sensors. Cameras are often static in visual surveillance network to get a robust low-level detection of mobile objects. The lateral sensors are very useful to separate people entering the access control site. The real-time constraint is very challenging as it implies that the solutions should be kept with a maximal computing time.

Our approach consists in applying Bayesian classifiers for shape recognition to handle the uncertainty accurately.

6.3.1 System Overview

Our goal is to have as much information as possible on the scene to understand precisely who is entering the site. To reach this goal, a fixed camera is placed above, at the height of about 2.5m, while a set of lateral sensors is placed on the side as shown in Figure 6.5. The camera observes the mobile objects from the top to detect and locate them. The lateral sensors observe the side of mobile objects, help to separate the detected mobile objects and provide information on their lateral shape.

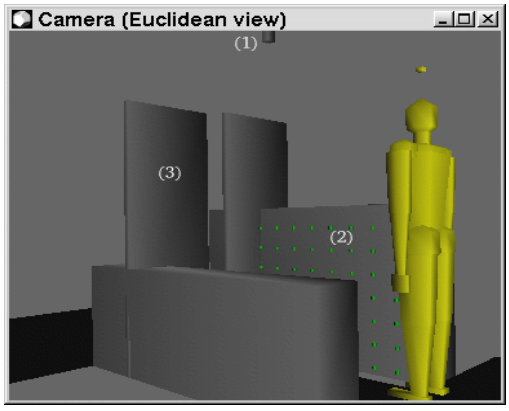


Figure 6.5: *The access control site contains (1) a top camera, (2) a set of lateral sensors and (3) an access door.*

The interpretation process is composed of four main tasks as shown in Figure 6.6. First a motion detector detects mobile objects evolving in the scene thanks to the top camera. Second, the mobile objects detected as one moving region can be separated thanks to the computation of the vertical projections of pixels or thanks to the lateral sensors. Third, the mobile objects are classified into several mobile object categories (e.g. adult, child, suitcase,

two adults close to each other) using Bayesian classifiers. Finally, the mobile objects are tracked to improve the reliability of the recognition process.

Moreover, we use a 3D model of the empty scene as a priori contextual knowledge of the observed environment. We define in the 3D scene model the 3D positions and dimensions of the equipment (e.g. the access door), the zones of interest (e.g. the entrance/exit zone) and the expected objects in the scene (adult, child, suitcase, two adults close to each other). Using context is essential for object recognition and for establishing the confidence in the whole interpretation system.

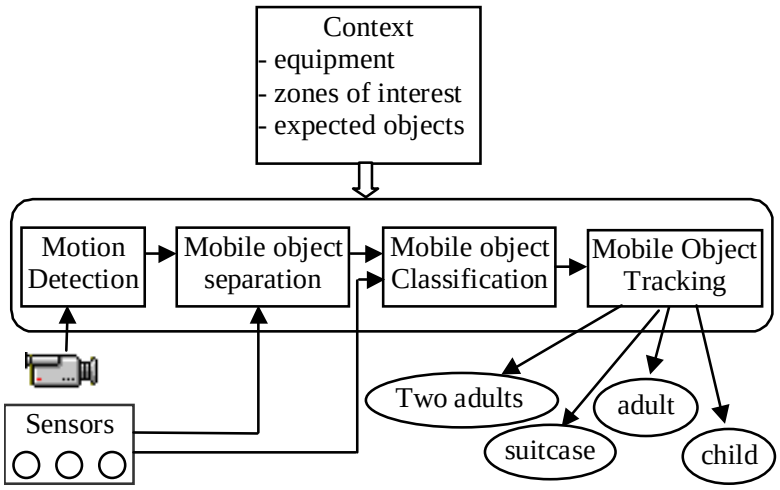


Figure 6.6: *The interpretation process takes as input a video stream and sensor information and outputs the recognized shape of mobile objects.*

6.3.2 Mobile Object Separation

A common error of motion detection is to detect several mobile objects (people walking closely to each other or person carrying a suitcase) as only one moving region (cf. Figure 6.7). The mobile object separation task consists in separating the moving regions that could correspond to several individuals into distinct moving regions. To accomplish this task, two techniques are combined together: computation of pixel projections and utilization of lateral sensors.

Computing Vertical Projections of Pixels

For each moving region, we calculate the potential points of separation (called separators) corresponding to potential borders between two persons. For that, we calculate the vertical projections of the moving region pixels as shown in Figure 6.7. When a “valley” is

detected between two “peaks”, we regard this valley as a potential separator between two distinct persons and the peaks as the gravity centers of these persons. If the size (the 3D length and the 3D width) of both distinct persons matches the dimension of a real person then this separator is valid.

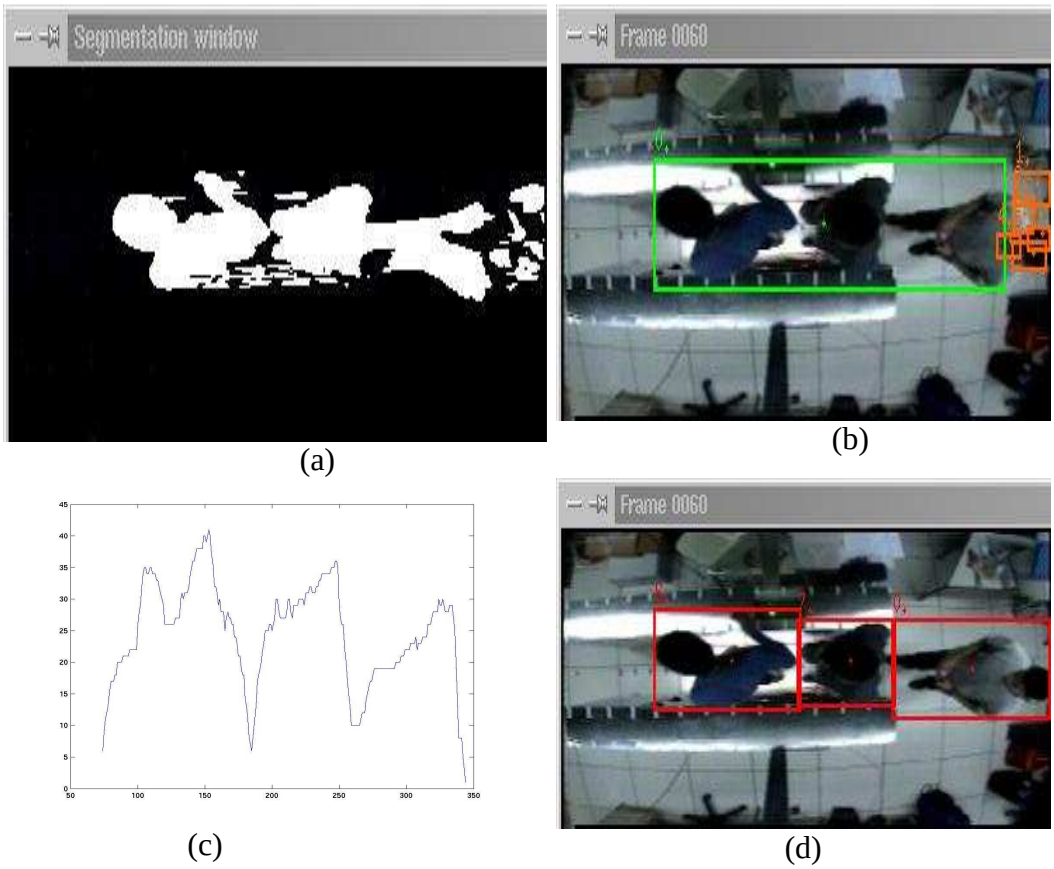
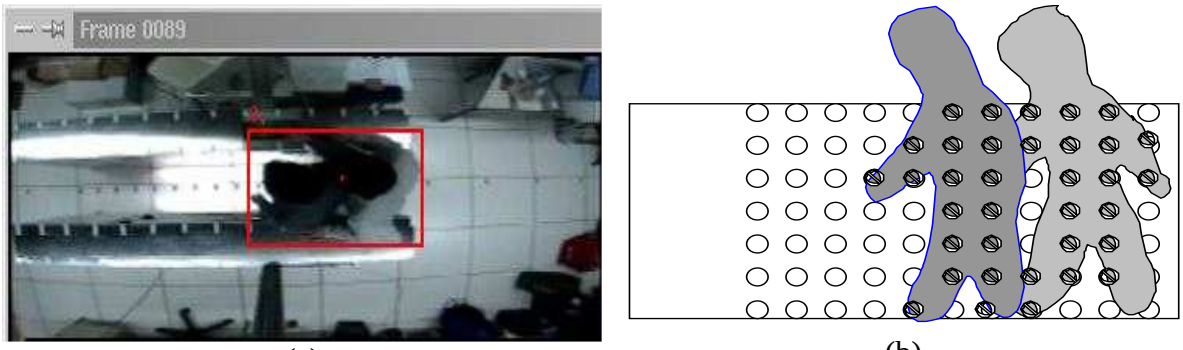


Figure 6.7*: Separation using the vertical projections of pixels: images (a), (b) illustrated three persons detected as one moving region; image (c) illustrated the vertical projections of pixels and image (d) shows that the moving region has been separated into three distinct moving regions.*

Using Lateral Sensors

The separation method using the vertical projections of pixels depends on the position of the persons relatively to the camera. This method cannot separate two adults walking closely to each other or far from the camera (cf. Figure 6.8-a). In this case, we use lateral sensors to detect the point of separation. For example, we can detect the non-occluded sensors announcing a space between two adults. More exactly, a separator is a non-occluded sensor found between two bands of occluded sensors (cf. Figure 6.8-b).



(a) (b)

Figure 6.8: *Separation using lateral sensors. Image (a) shows two adults walking closely to each other and detected as one region mobile; drawing (b) shows three non-occluded sensors detected between two adults. These three sensors form a separator.*

In addition to help to separate two adults walking closely to each other, lateral sensors provide also clues to separate the objects associated to the persons such as bags, suitcases and in certain cases children. To separate objects, we define a separator as a column of sensors having a large majority of non-occluded sensors. These separators enable to separate two consecutive suitcases and a suitcase or a child from the adult if the distance between them is big enough (cf. Figure 6.9).

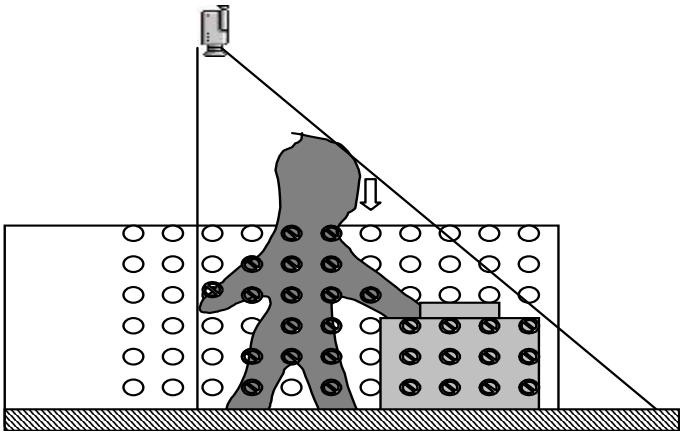


Figure 6.9: *The top camera does not see the suitcase but lateral sensors help to separate it from the person. The separator (column of sensors having a large majority of non-occluded sensors) is indicated by an arrow.*

6.3.3 Mobile Object Classification

Mobile Object Models

Initially, we have to build a model (class) for different mobile objects such as “adult”, “child”, “suitcase” and “two adults close to each other”. We are supposed to have the model

for two adults close to each other because in cases where two adults are walking very closely to each other, neither the vertical projections of pixels nor the lateral sensors can separate them. The model for a mobile object is built from its characteristics obtained by the top camera and the lateral sensors. The top camera provides information on its 3D length L_t and its 3D width W_t. For lateral sensors, we divide the zone of sensors at the mobile object position into n sub-zones (cf. Figure 6.10). Then, for each sub-zone i, we calculate the density S_i of the occluded sensors and we use this density as a characteristic of the mobile object.

The number of sub-zones, their dimension and their position should be chosen intelligently according to the properties and the people body parts (e.g. the legs is one of the sensitive body part for a person). In our experimental test, to simplify the calculation, we divide the zone of sensors into 9 sub-zones. The dimension of each sub-zone is defined proportionally with the dimension of the zone as shown in Figure 6.10.

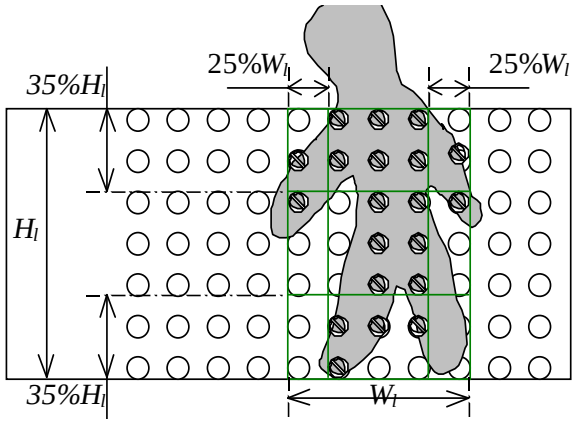


Figure 6.10: *The zone of sensors is divided into 9 sub-zones and the density of occluded sensors in each sub-zone is used as characteristic of mobile object shape.*

To reinforce the mobile object model, we also consider the lateral 3D width W_l and the lateral 3D height H_l of the zone as characteristics of the mobile object.

In conclusion, in our implementation, a mobile object model is a set of 13 characteristics: L_t, W_t, W_l, H_l and S_i, i=1..9. However, we can add other characteristics of mobile object to enrich the model.

Training Bayesian Classifiers

For each class of mobile object, we use about 250 instances representative of the class to train a dedicated classifier. For each frame, we compute and record the values of mobile object characteristics (i.e. L_t, W_t, W_l, H_l, S_i, i=1..9). We count the number of mobile objects having the same value c for the characteristic C. So we obtain the frequency for a given mobile object class to have the value *c* for the characteristic C. In other words, we obtain the conditional probability P(c|F) that a mobile object has the value *c* for the characteristic C knowing that this mobile object belongs to class F.

By counting the number of mobile objects of other classes (i.e. all classes excluding F) having the same value c for the characteristic C, we also obtain the conditional probability P(c|¬F) that a mobile object has the value c for the characteristic C knowing that this mobile object belongs to another class (¬F corresponds to all classes excluding class *F*).

We have developed a tool permitting a user to annotate a frame with information for the learning task. Once the user has chosen a video sequence, the tool visualizes each frame of the sequence and asks the user to delimit the mobile object seen in the frame and to give its class (e.g. adult, child, suitcase, two adults). The tool then, for each mobile object, calculates automatically the values (l_t, w_t, w_l, h_l, s_i, i=1..9) of its characteristics (corresponding to ground truth) and records them into 13 files. These files are used latter as the training data. They are useful also for evaluating the recognition method. For example, we compare the output of the recognition module with the ground truth data in these files.

Mobile Object Classification

To classify mobile objects into the expected classes, we compare, in each frame, the characteristics of the mobile object with the characteristics of the mobile object classes using the Bayes rule.

For each frame and for each mobile object o, we build a vector containing its ***degrees of membership*** D(o∈F) for all classes F. The degree of membership is the ratio of the probability P(o∈F) that the mobile object *o* belongs to the class F divided by the probability P(o∈(¬F)) that the mobile object o belongs to another class (¬F corresponds to any class excluding class F) as shown by equation 6.1.

$$D(o \in F) = \frac{P(o \in F)}{P(o \in (\neg F))}$$

Equation 6.1. *The degree of membership is defined as the ratio of the probability that the mobile object o belongs to the class F.*

By using Bayes rule and by replacing the mobile object by its characteristic set, we obtain:

$$D(o \in F) = \frac{P(F | l_t, w_t, w_l, h_l, s_1, s_2, ..., s_9)}{P(\neg F | l_t, w_t, w_l, h_l, s_1, s_2, ..., s_9)}$$

Equation 6.2. *The degree of membership is computed as the ratio of the conditional probability that the mobile object characteristics correspond to the class F.*

Where P(F |l_t, w_t, h_l, w_l, s_1,…, s_9) is the conditional probability that the mobile object characteristics correspond to class F knowing the value set (l_t, w_t, h_l, w_l, s_1,…,s_9).

After checking that all characteristics are independent we simplify equation 6.2:

$$D(o \in F) = \frac{P(l_t|F) \times ... \times P(s_8|F) \times P(s_9|F)}{P(l_t|\neg F) \times ... \times P(s_8|\neg F) \times P(s_9|\neg F)}$$

Equation 6.3. *the degree of membership is computed as the ratio of the conditional probability of each characteristic corresponding to the class F.*

Where P(c|F) (respectively P(c|¬F)) is the conditional probability that a mobile object characteristic C has the value c knowing that this mobile object belongs to class F (respectively to another class). These conditional probabilities are obtained from the ground truth data as discussed in the precedent section.

The mobile object o is then classified into the class with the biggest degree of membership.

6.3.4 Mobile Object Tracking

The Bayesian classifiers sometimes miss recognize or do not recognize the class for a mobile object due to the large variety of lateral shapes (i.e. due to lack of training data). To increase the recognition reliability, we track mobile object when they evolve through the scene. This tracking stage enables, on one hand, to correct potential frame to frame classification errors and on the other hand, can help latter to recognize human behaviors and scenarios.

Mobile Object Matching

The mobile object matching stage consists of matching the mobile objects previously detected at time t-1 with new ones detected at time t. To calculate these correspondences, we currently use three different criteria: their compatibility of lateral shape, their 3D distance and the overlap between their bounding boxes. The decision to match or not two mobile objects is made based on thresholds on the weighted sum of these criteria.

For each criterion k, we construct a binary matrix $n_o \times n_n$ M_k (n_o and n_n are the number of mobile objects detected at t-1 and the number of new mobile objects detected at t) containing the correspondences for the criteria k. The final decision matrix M is computed as the weighted sum of the matrices M_k as shown by equation 6.4.

$$M = \frac{\sum_{k=1}^{3} w_k M_k}{\sum_{k=1}^{3} w_k}$$

Equation 6.4. *The matrix M combines the three matrices of correspondences between mobile objects.*

If M(i, j), (i=1..n_o; j=1..n_n) is greater than a certain threshold ∂ then the mobile objects o_i at t-1 and o_j at t are matched. If an old mobile object at t-1 matches with several new mobile objects at t, the mobile object having the best correspondence (i.e. the greatest correspondence) is chosen.

Recognition Refinement

To increase the reliability of the shape recognition algorithm (i.e. to correct potential classification errors), we maintain the temporal coherency of the membership degree vector D composed of the membership degrees for all classes. For each previously detected mobile object at t-1, we update this vector D with the temporary membership degree D_t detected at time t as shown by equation 6.5.

$$\boxed{D = D + \omega_t D_t}$$

Equation 6.5. *Update of the membership degree vector D.*

Where ω_t, $0 \leq \omega_t \leq 1$ is the confidence weight at t of the Bayesian classifiers. This confidence weight is chosen according to the lateral sensor density where the mobile object is detected. For example, a high confidence weight is chosen if the mobile object at time t is found in a zone where there are many lateral sensors (i.e. where we obtain a more precise shape of mobile object).

The final class of mobile object is chosen according to the biggest value in the new vector of membership degrees.

6.3.5 Results

The recognition module has been tested in two stages: a stand-alone experimentation on recorded image sequences (i.e. test offline) and an experimentation in live in interaction with the kernel of an existing access control device used in subways at RATP.

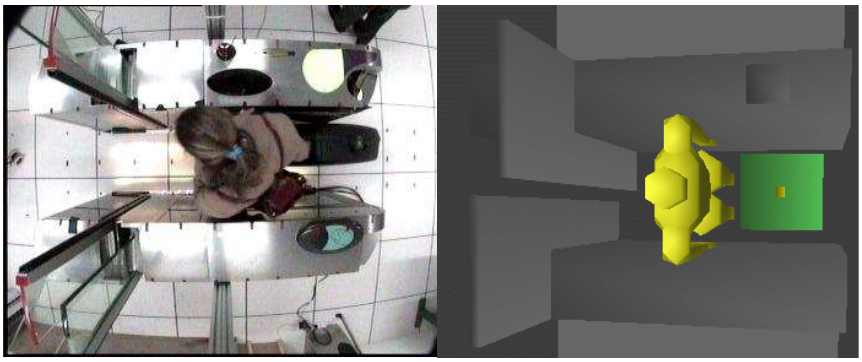
To train the Bayesian classifiers, for each class “adult”, “child”, “suitcase”, we used about 300 frames as training data and about 1000 frames for testing. For the class “two adults close to each other”, at the present time, we have only 32 frames in total to represent this class. For this class, we used 15 frames as training data and 17 frames for testing.

In the stand-alone stage, the results are very promising. A large majority of mobile objects have been correctly recognized with a high degree of membership (cf. Figure 6.11 and Table 6.1). More than 94% of adults, children and suitcases are correctly recognized. More

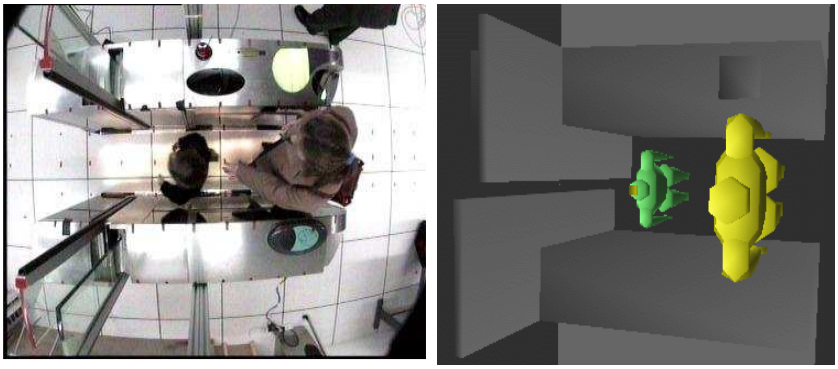
precisely, for the adult class, the true positive is 98%, the false positive is 1% and the false negative is 2%. The true positive for “two adults close to each other” is about 73% due to the lack of training data in the learning phase.

Mobile Object	True Positive	False Positive	False Negative	Frames used for testing	Frames used as training data
Adult	98%	1%	2%	1102	327
Child	94%	3%	6%	1050	295
Suitcase	95%	2%	5%	1008	305
Two adults close to each other	73%	0%	27%	17	15

Table 6.1: *More than 94% of adults, children and suitcases are correctly recognized.*



(a) An adult with a suitcase



(b) An adult with a child

(b) Two adults walking closely to each other

Figure 6.11: *The images on the right show the recognition result in a 3D animation from the processing of images on the left.*

The recognition module sometimes miss classifies a child with a small suitcase and vice versa due to the similarity of appearance. Almost all the potential errors in the frame to frame classification have been corrected by the frame to frame tracking.

The recognition result depends on training videos used in the learning phase. To obtain better results, we should enrich the training data for each class. For example, for the class “adult”, the training data should include large variety of persons (fat, thin, tall, short, adult in summer/winter clothes, etc).

In the live experimentation, the recognition module runs on a PC (Pentium IV 2.8 GHz, 1GB memory, Linux) and receives a video stream at 25 images per second. The maximal time for processing one image is inferior to 35ms. The real-time constraint is then satisfied.

6.3.6 Conclusion on multi-sensor system

We have described in this paper a video and multi-sensor interpretation process for shape recognition. The key of the recognition method is to compute mobile object properties thanks to a camera and a set of lateral sensors and then to use Bayesian classifiers. The system has been tested offline and on live video streams, and gives very promising results.

The recognition, as previously said, depends on training videos and sensor data. Since realistic training data cannot include all varieties of mobile object classes and shapes, the first next step will consist in studying supervised and non-supervised machine learning techniques in order to learn dynamically new classes of mobile objects. Moreover, with the objective of helping the system to control the access safely and comfortably while preventing from fraud, the second next step should consist in human behaviour and scenario recognition in order to understand and anticipate the evolutions of mobile objects. For this, we plan to adapt methods proposed in [Bremond 34, and 35]. Finally, for the system to be more robust, the third next step will consist in studying the system autonomy. For example, the system should be able to detect failures (sensors or camera breakdown, change of light) and set up a degraded

operation mode according to the resource available. The objective will be to have a system that can reconfigure itself dynamically and autonomously.

6.4 Conclusion on maintaining 3D coherency throughout time

In this chapter, we have reviewed several works on maintaining 3D coherency throughout time. These works are organized following two directions: coherency throughout time and coherency of multi-sensor information in the 3D space. These works have presented some solutions to bridge the gap between signal and semantic levels. The key issues are:

- Building a common knowledge representation for combining all information describing the scene.

- Modelling and managing the uncertainty and the incompleteness of data and models characterizing the scene and its dynamics.

These works have described some efficient information fusion algorithms that can lead to a coherent understanding of the observed scene. Despites these success, there are still some limitations depending on the uncertainty and lack of information. Nevertheless, thanks to a precise formalism combining uncertain information coming from heterogeneous sensors, we are able in some cases to understand the evolution of the mobile objects in the scene and so to recognise the events characterising the behaviours of these objects of interest.

Chapter 7

Event recognition

A problem of current focus in cognitive vision is event recognition. The goal is to develop a systematic methodology to the design, to implement and to integrate cognitive vision systems for recognizing events (called also scenarios) involved in a scene observed by sensors (mostly cameras).

Since the years 90s, there have been several research units and companies defining new approaches to design systems that can understand human activities in dynamic scenes many works have been done to recognise from videos [Haering and Lobo, 2001], [Ayers and Shah, 2001], [Olguin and Pentland, 2006], [Chellappa et al., 2005], [Xiang and Gong, 2006a], in particular using motion based techniques [Crowley, 2006], [Shah, 2003], [Brdiczka et al., 2006], [Andrade et al., 2006], [Parameswaren and Chellappa, 2005] and [Ogata et al., 2006]. For instance, several works have been applied to home-care [Cucchiara et al., 2005], [Cucchiara et al., 2003] and [Huang et al., 2005]. We have classified event recognition approaches into two main categories: probabilistic/stochastic approaches and constraint-based approaches. This section describes several of these works and a short discussion of issues that still need to be studied.

Probabilistic and Stochastic Approaches

The main probabilistic and stochastic approaches that have been used to recognize video events include neural networks [Howell and Buxton, 2002], bayesian classifiers [Bremond, 24], [Intille and Bobick, 2001], and Hidden Markov Models (HMM) [Bremond, 8], [Barnard et al., 2003].

The first two approaches (i.e. neural networks and bayesian classifiers) are well adapted to combine observations at one time point, but they have not a specific mechanism to represent the time and temporal constraints between visual observations such as Allen's interval algebra operators [Allen, 1981]. For instance, Dynamic Bayesian Networks (DBN) have been used successfully to recognize short temporal actions [Buxton, 2002], but the recognition process depends on time segmentation: when the frame-rate or the activity duration change, the DBN has to be retrained. Other extensions of DBN have been proposed to model sequences of events using hidden nodes to represent hidden temporal states [Gong and Xiang, 2003]. These Bayesian networks are similar to HMMs. Thus, these approaches are not well adapted to process temporal sequences.

The third approach based on HMM is very popular and has been successfully used to recognize temporal sequences of simple events taking into account the uncertainty of the visual observations. For instance, a typical algorithm as the one presented by Hongeng et al. [Bremond, 23] uses HMM to recognize multi-state activities in dynamic scenes. A multi-state activity is a temporal sequence of mono-state activities and is inferred from the probability of mono-state activities observed in a time interval. A mono-state activity is a primitive activity corresponding to a coherent unit of motion. For instance, "a car slows down toward an object" is a mono-state activity contained in the multi-state activity "a car turns back at a check-point". Thus, the HMM approach shows the advantage of modelling the uncertainty of the observed environment and reasoning with uncertainty by propagating the recognition likelihood through a finite state automaton. However, even if few solutions have been proposed for handling two mobile objects, such as coupled HMMs ([Brand et al., 1997] & [Oliver et al., 2000]) and DPNs (Dynamic Probabilistic Networks) [Gong and Xiang, 2003], HMMs are not well suitable for recognizing activities involving several mobile objects for four main reasons. First, all the combinations of events involving different mobile objects need to be modelled in the HMM. Second, HMM can easily model sequences of events, but not complex temporal relationships (e.g. Allen's interval algebra operators [Allen, 1981]). Third, the normalization process of the recognition of several concurrent activities (modelled by different HMMs) is not well addressed. Usually, the activity lasting a longer time period gets a lower recognition likelihood. Moreover, experimentations are usually done with short videos starting and ending synchronously with the activities (i.e. one video corresponding exactly to one activity) and not with long videos containing mixed and successive concurrent activities. Fourth, the learning phase is critical to learn the parameters of HMMs that model complex activities. For instance, the authors of [Gong and Xiang, 2003] use only 16 activity samples (8 loading and 8 unloading videos) to train their HMMs.

Other formalisms based on mobile object trajectories have been used to recognize frequent activities. For instance, the authors in [Foresti et al., 2005] have used clustering techniques to learn frequent trajectories and to deduce abnormal events. The authors of [Rao et al., 2002] have computed the motion units (called dynamic instants and intervals) of a trajectory to define an activity. These formalisms have the same drawbacks than classical HMMs, they are well adapted to recognize a sequence of events performed by mono-physical-object.

Constraint-Based Approaches

Constraint-based approaches have been largely used to recognize activities for few decades. The main trend consists in designing symbolic networks whose nodes or predicates correspond to the boolean recognition of simpler events. The first constraint-based approaches have been developed in the 70s and include plan recognition, [Kautz and Allen, 1986], [Schmidt et al., 1978] and event calculus [Shanahan, 1990] and [Kowalski and Sergot, 1986]. However, these approaches have not been applied to scene understanding based on real-world perceptual observations.

Other approaches including Petri Net [Castel et al., 1996], [Tessier, 2003], logic programming [Davis et al., 2005], script-based language, constraint resolution [Needham et al., 2005] and [Ivanov and Wren, 2006] and chronicle recognition [Nokel, 1989], [Kumar and Mukerjee, 1987], [Dousson, 1994], and [Dousson et al., 1993] have been adapted for recognizing activities through videos.

For instance, the authors in [Lesire and Tessier, 2005] have designed a Petri Net to recognize a given activity, whose nodes correspond to typical situations and the tokens to the mobile objects involved in the activity. But, this approach uses just one Petri Net to recognize one activity type and cannot recognize all the occurrences of the same activity.

Stochastic grammar has been proposed to parse simple actions recognized by vision modules [Ivanov and Bobick, 2000]. Logic and Prolog programming have also been used to recognize activities defined as predicates [Davis et al., 2005]. Constraint Satisfaction Problem (CSP) has been applied to model activities as constraint networks [Rota and Thonnat, 2000]. These last three approaches are interesting and have successfully recognized complex activities. However, they do not have specific mechanisms to handle temporal constraints so they have to explore all possible temporal combinations of events. To reduce this combinatorial explosion, the authors in [Rota and Thonnat, 2000] have proposed to first check whether the constraint network has a solution using AC4 arc consistency algorithm [Mohr and Henderson, 1986]. However, to find all the solutions, these approaches Store all Totally Recognized Events (and are called in the following STRE) to be used to recognize other more complex events. In practice, these approaches can recognize in real-time only activities involving a small number of physical objects.

Other techniques for the recognition of human activities have been proposed to reduce this combinatorial explosion by propagating the temporal constraints inside the constraint network. Then, the recognition is limited to only the subnetworks (complying with the satisfied temporal constraints) that can lead to a possible activity. These approaches Store all Partially Recognized Events (and are called SPRE). The SPRE algorithms envisage all combinations that can occur and store only these predictions (a prediction corresponds to an event that is occurring in the observed environment and partially recognized by the recognition process at the current instant) to recognize complete events in the future. For instance, an efficient version was proposed by the authors of [Pinhanez and Bobick, 2003] who described a temporal constraint network (called PNF for Past, Now and Future) to recognize activities. However, this network cannot represent event duration and is mainly dedicated to the recognition of event sequences.

More generally, the notion of chronicle was first introduced in [Kumar and Mukerjee, 1987] (and called dynamic situation) and then extended by [Dousson and Ghallab, 1994] and [Ghallab, 1996]. A chronicle is represented as a set of events (detected by specific routines) and sub-chronicles (recognized by the recognition process) linked by temporal constraints. The temporal aspects are the starting/ending time points of a chronicle and also the delay between two chronicles. A graph propagation technique is used to compile pre-defined chronicles. For a given chronicle, the compiler first builds a temporal constraint graph whose

nodes correspond to the time points defined within the given chronicle and whose arcs correspond to the temporal relations of those time points. The chronicle recognition algorithm recognizes incrementally predefined chronicles by using this graph and by storing all Partially Recognized Chronicles (PRC). Each time an event is received or a chronicle is recognized, new temporary structures are created for the new PRCs. In each PRC, there are time windows corresponding to the time delay when another event/sub-chronicle is expected or forbidden. If there is any violation of these time windows, the PRC is deleted. If not, the chronicle is recognized when all the time windows are correctly fulfilled. This approach recognizes correctly predefined chronicles and makes the recognition of chronicles possible in real-time. This approach has been applied to the video surveillance of metro stations [Chleq and Thonnat, 1996].

However, the SPRE algorithm was designed to recognize mono-physical-object events (i.e. chronicles), so, it constitutes a drawback for multi-physical-object events. For a multi-physical-object events, the algorithm has to create all predictions corresponding to all combinations of potential physical-objects. Thus, this technique can lead to a combinatorial explosion.

Related Work Discussion

All these techniques allow an efficient recognition of scenarios, but there are still some temporal constraints which 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 and Thonnat, 2000]. Thus, complexity and real-time processing are two major issues partially addressed to recognize complex activities involving multi-physical-objects. As shown previously, both STRE and SPRE algorithms can recognize efficiently pre-defined events. However, they show several drawbacks, for instance, (1) the SPRE algorithms store and maintain all occurrences of partially recognized events as a potential recognition in the future so can lead to a combinatorial explosion and (2) the STRE algorithms perform at each instant a complete search among all possible events and sub-events recognized in the past and all possible combinations of physical-objects so it can also lead to an exponential algorithm.

In the literature, a second point concerns scene understanding algorithms recognizing activities from audio-video sensors. Only few works have addressed this issue limiting themselves to the fusion of information at the feature level. Most of these works combine audio and video clues to track more robustly people in a scene [Cristani et al., 2006]. Few works have also addressed the recognition of activities for meeting rooms [Al-Hames et al., 2005] and broadcasting videos [Barnard et al., 2003] using HMM, but these works are limited to the recognition of simple events. Moreover, audio and video events cannot always be combined at the feature level because they can be related to different aspects of the activity. In these cases, audio-video information has to be fused at the semantic level. For instance, in a fighting event, the audio event “shouting” and the video event “hitting” are weakly constrained and have to be combined at the semantic level.

In this chapter, we studied one example of both approaches, numerical and symbolic approaches. The first section presents an algorithm for event recognition based on finite state automata and AND/OR trees, which has been designed during the European project ADVISOR with the help of three engineers, Frederic Cupillard, Alberto Avanzi and

Christophe Tornieri [Bremond 3, 30, 31 and 39]. This approach is well adapted for the effective recognition of events with simple temporal constraints, and with strong connections with vision algorithms. Another advantage of this approach is the easy prototyping of new recognition methods (e.g. automata).

The second section describes an efficient real-time algorithm, which has been proposed during the PhD of Thinh Van Vu [Bremond 28, 29, 33 and 35]. This scenario recognition algorithm takes as input the a priori knowledge of the scene and a stream of individuals tracked by vision algorithms as described in previous chapters. This algorithm is specially adapted for processing any type of temporal constraints (including temporal Allen algebra). This algorithm manages to search in a linear time for previously recognised sub-scenarios in order to recognise a current scenario. Therefore, based on the analysis of the state of the art, our work focuses on a novel algorithm for activity recognition by taking advantages of both SPRE and STRE approaches. A second goal is to show that this novel algorithm is also adapted to recognize complex audio-video activities and that the recognition can be done at the event level [Bremond, 51]. We have also designed a declarative language for helping end-users to describe their scenarios of interest.

7.1 Numerical Approach for Event Recognition

We propose in this section an approach for recognising either isolated individual, group of people or crowd behaviours in the context of visual surveillance of metro scenes using multiple cameras. In this context, a behaviour recognition module relies on a vision module which tracks all mobile objects (called also actors) evolving in the scene. For each tracked actor, the behaviour recognition module performs three levels of reasoning: states, events and scenarios. We have also defined a general framework to easily combine and tune various recognition methods (e.g. automaton, Bayesian network or AND/OR tree) dedicated to the analysis of specific situations (e.g. mono/multi actors activities, numerical/symbolic actions or temporal scenarios). A main problem in behaviour recognition is the ability to define and reuse methods to recognise specific behaviours, knowing that the perception of behaviours is strongly dependent on the site, the camera view point and the individuals involved in the behaviours. Our approach consists in defining a formalism allowing us to write and easily reuse all methods needed for the recognition of behaviours. This formalism is based on three main ideas. First, the formalism should be flexible enough to allow various types of operators to be defined (e.g. a temporal filter or an automaton). Second, all the needed knowledge for an operator should be explained within the operator so that it can be easily reused. Finally, the description of the operators should be declarative in order to build an extensible library of operators.

Validation results on different methods used to recognise specific behaviours are illustrated through three behaviour recognition examples: "Fighting", "Blocking" and "Fraud" behaviours. This work has been performed in the framework of the European project ADVISOR (http://www-sop.inria.fr/orion/ADVISOR).

7.1.1 Behaviour representation

We call an actor of a behaviour, any scene object involved in the behaviour, including static objects (equipment, zones of interest…), individuals, group of people or crowd. The entities needed to recognise behaviours correspond to different types of concepts which are:

1. **The basic properties:** a characteristic of an actor such as its trajectory or its speed.

2. **The states:** a state describes a situation characterising 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 state: *"an individual stays close to the ticket vending machine"*, 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 sub scenarios. Behaviours are specific scenarios (dependent on the application) defined by the users. For example, to monitor metro stations, end-users have defined 5 targeted behaviours: *"Fraud"*, *"Fighting" "Blocking"*, *"Vandalism"* and *"Overcrowding"*.

To compute all the needed entities for the recognition of behaviours, we use a generic framework based on the definition of **Operators** which are composed of four attributes:

Operator name: indicates the entity to be computed such as the state *"an Individual is walking"* or *"the trajectory is straight"*.

Operator input: gives a description of input data. There are two types of input data: basic properties characterising an actor and sub entities computed by other Operators.

Operator body: contains a set of competitive methods to compute the entity. All these methods are able to compute this entity but they are specialised depending on different configurations. For example, to compute the scenario *"Fighting"*, there are 4 methods (as shown on Figure), for example, one method computes the evolution of the lateral distance between people inside a group. A second one detects if someone, surrounded by people, has fallen on the floor.

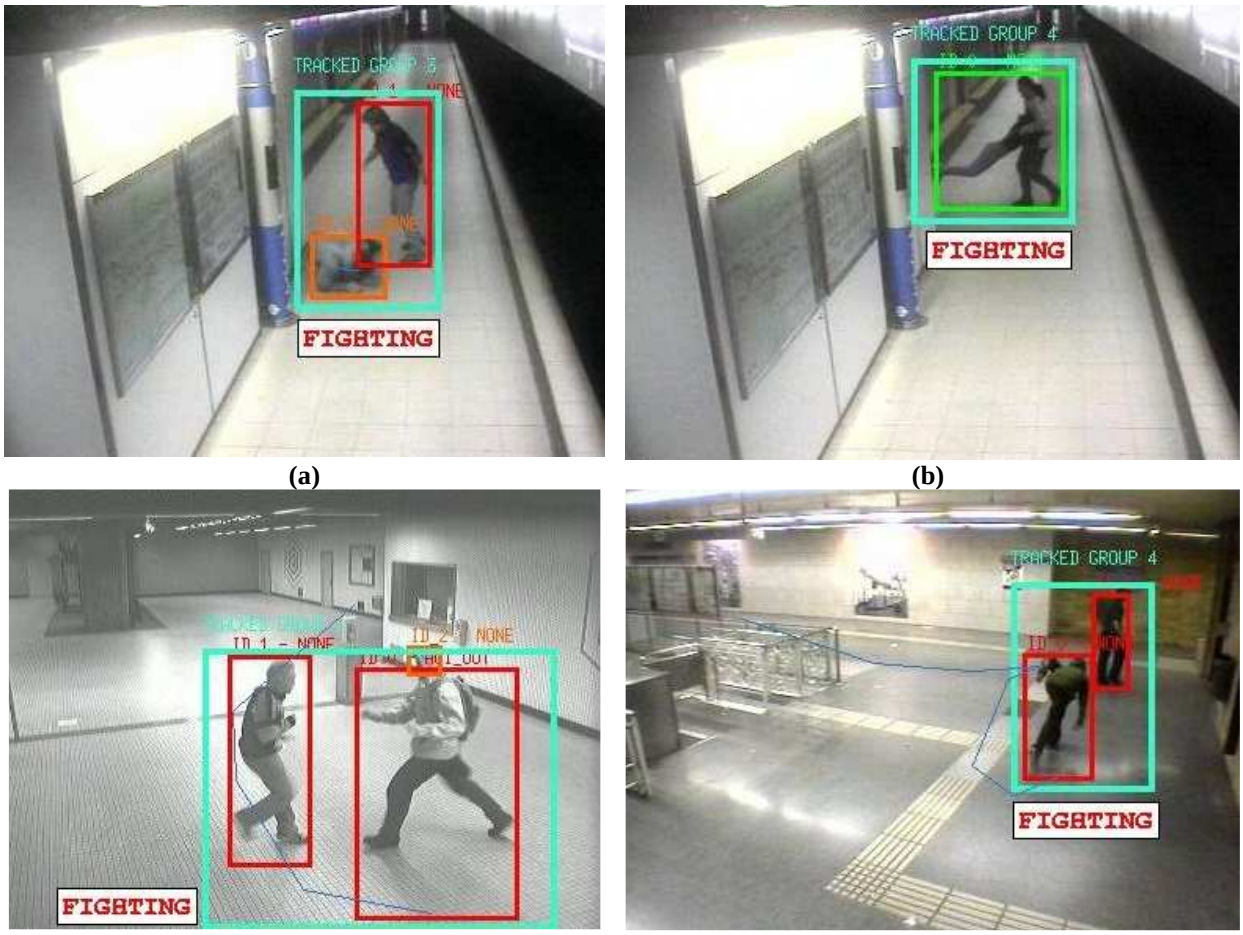


Figure 7.1: *This figure illustrates four methods combined by an AND/OR tree to recognise the behaviour "Fighting". Each image illustrates a configuration where one method is more appropriate to recognise the behaviour: (a) lying person on the floor surrounded by people, (b) significant variation of the group width, (c) quick separation of people inside a group and (d) significant variation of the group trajectory.*

Operator 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 gives two advantages: It first enables us to test a set of methods to compute an entity, independently of other entities. So we can locally modify the system (the methods to compute an entity) while keeping it globally consistent (without modifying the meaning of the entity). Second, the network of Operators to recognise one scenario is organised as a hierarchy. The bottom of the hierarchy is composed of states and the top corresponds to the scenario to be recognised. Several intermediate levels composed of state(s) or event(s) can be defined.

7.1.2 Behaviour recognition

We have defined four types of methods depending on the type of entities:

Basic properties methods: we use dedicated routines to compute properties characterising 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.

State methods: we use numerical methods which include the computation of: (a) 3D distance for states dealing with spatial relations (e.g. *"an individual is close to the ticket vending machine"*), (b) the evolution of temporal features for states dealing with temporal relations (e.g. *"the size of a group of people is constant"*) and (c) the speed for states dealing with spatio-temporal relations (e.g. *"an individual is walking"*) and (d) the combination of sub states computed by other operators.

The output of these numerical methods is then classified to obtain a symbolic value.

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 event *"a group of people enters a zone of interest"* is detected when the state *"a group of people is inside a zone of interest"* changes from false to true.

Scenario methods: for simple scenarios (composed of only 1 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 recognise the scenarios "Blocking" and *"Fraud"* as described on Figure and Figure .

For composed scenarios defining a single unit of movement composed of sub scenarios, we use Bayesian networks as proposed by [Bremond 23 and 25] or AND/OR trees of sub scenarios as illustrated on Figure . A description of Bayesian networks for scenario recognition can be found in [34]. We have defined one Bayesian network to recognise the *"violence"* behaviour composed of 2 sub scenarios: *"internal violence"* (e.g. erratic motion of people inside a group) and "*external violence"* (e.g. quick evolution of the trajectory of the group).

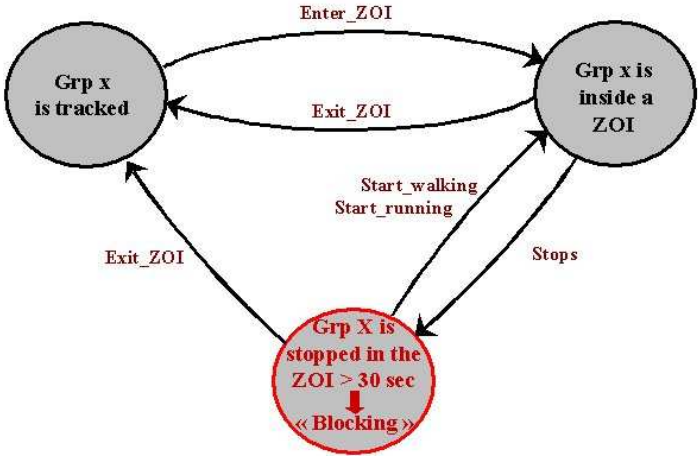


Figure 7.2: *To check whether a group of people is blocking a zone of interest (ZOI), we have defined an automaton with three states: (a) a group is tracked, (b) the group is inside the ZOI and (c) the group has stopped inside the ZOI for at least 30 seconds.*

Both of these methods need a learning stage to learn the parameters of the network using ground truth (videos annotated by operators). Bayesian networks are optimal given ground truth but the AND/OR trees are easier to tune and to adapt to new scenes.

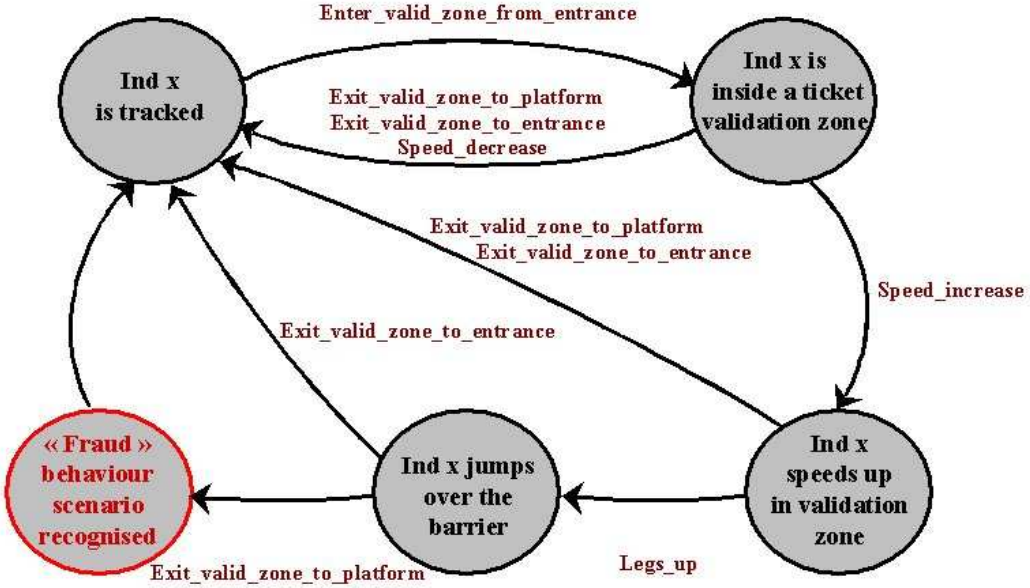


Figure 7.3: *To check whether an individual is jumping over the barrier without validating his ticket, we have defined an automaton with five states: (a) an individual is tracked, (b) the individual is at the beginning of the validation zone, (c) the individual has a high speed, (d) the individual is over the barrier with legs up and (e) the individual is at the end of the validation zone.*

For scenarios with multiple actors involved in complex temporal relationships, we use a network of temporal variables representing sub scenarios and we back-track temporal constraints among the already recognised sub scenarios as described in the next section.

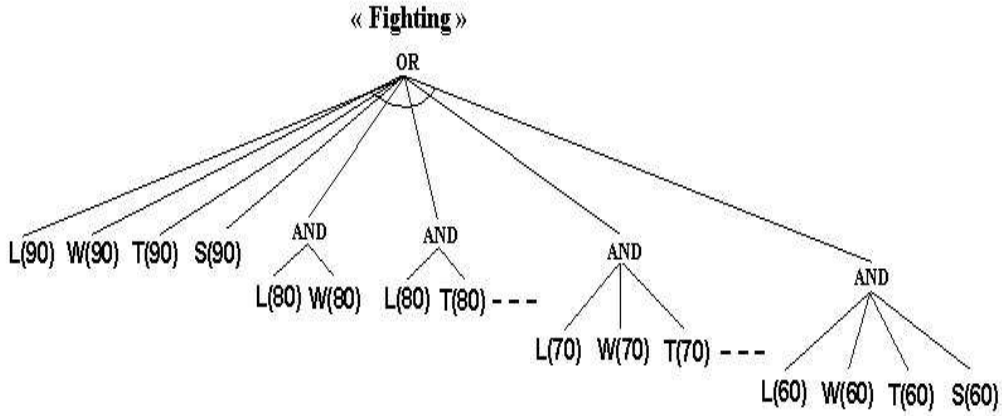
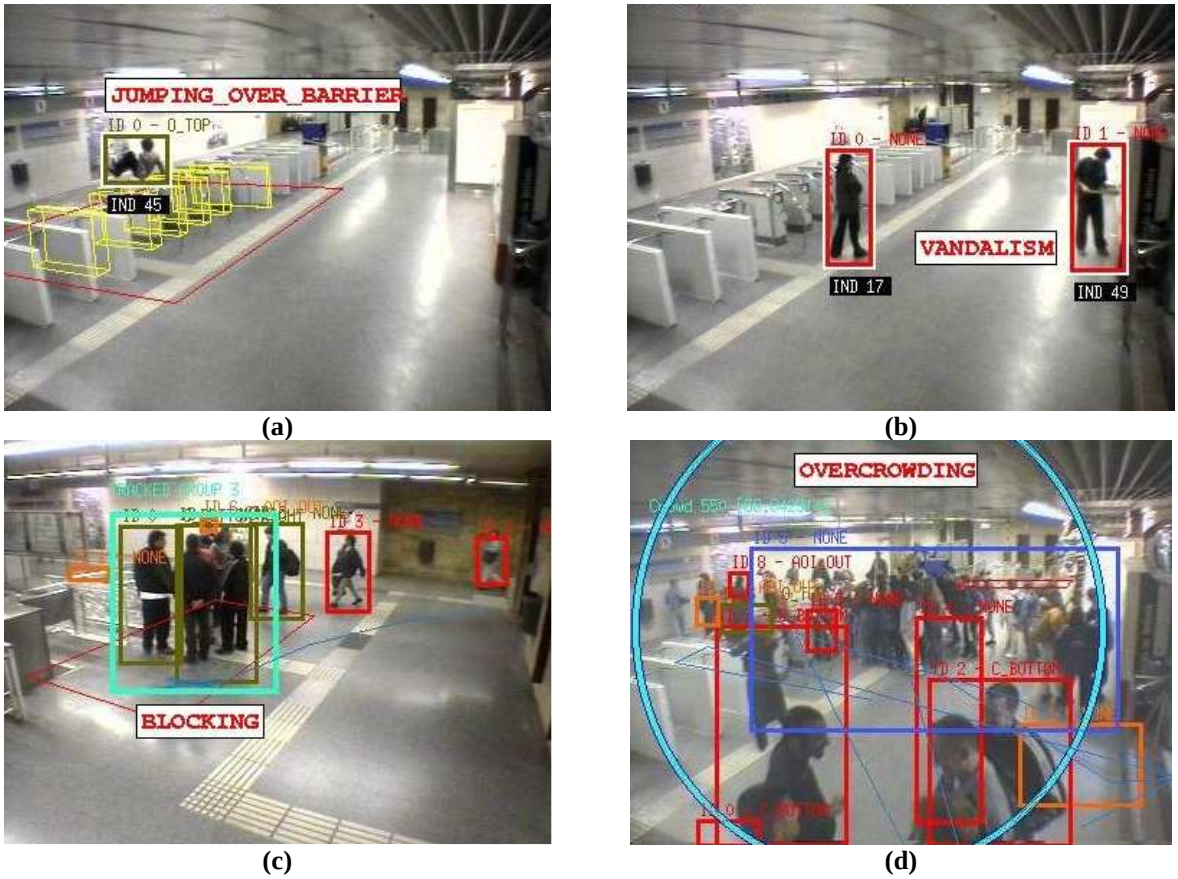


Figure 7.4: *To recognise whether a group of people is fighting, we have defined an AND/OR tree composed of four basic scenarios: (L) lying person on the floor surrounded by people, (W) significant variation of the group width, (S) quick separation of people inside the group and (T) significant variation of the group trajectory. Given these four basic scenarios we were able to build an OR node with all combinations (corresponding to 15 sub scenarios) of the basic scenarios. These combinations correspond to AND nodes with one up to four basic scenarios. The more basic scenarios there are in 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 parameterise these thresholds, we have performed a learning stage consisting in a statistical analysis of the recognition of each basic scenario.*



(a) **(b)**

(c) **(d)**

Figure 7.5: *This figure illustrates four behaviours selected by end users and recognised by the video interpretation system: (a) "Fraud" recognised by an automaton, (b) "Vandalism" recognised by a temporal constraint network, (c) "Blocking" recognised by an automaton and (d) "Overcrowding" recognised by an AND/OR tree.*

7.1.3 Behaviour recognition results

The behaviour recognition module is running on a PC Linux and is processing four tracking outputs corresponding to four cameras with a rate of 5 images per second. We have tested the whole video interpretation system (including motion detection, tracking and behaviour recognition) on videos coming from ten cameras of Barcelona and Brussels metros. We correctly recognised the scenario *"Fraud"* 6/6 (6 times out of 6) (Figure .a), the scenario *"Vandalism"* 4/4 (Figure .b), the scenario *"Fighting"* 20/24 (Figure), the scenario *"Blocking"* 13/13 (Figure .c) and the scenario *"overcrowding"* 2/2 (Figure .d). We also tested the system over long sequences (10 hours) to check the robustness over false alarms. For each behaviour, the rate of false alarm is: 2 for *"Fraud"*, 0 for *"Vandalism"*, 4 for *"Fighting*", 1 for *"Blocking"* and 0 for *"Overcrowding"*.

Moreover, in the framework of the European project ADVISOR, the video interpretation system has been ported on Windows and installed at Barcelona metro in March 2003 to be evaluated and validated. This evaluation has been done by Barcelona and Brussels video surveillance metro operators during one week at the Sagrada Familia metro station. Together with this evaluation, a demonstration has been performed to various guests, including the European Commission, project Reviewers and representative of Brussels and Barcelona Metro to validate the system. The evaluation and the demonstration were conducted using both live and recorded videos: four channel in parallel composed of three recorded sequences and one live input stream from the main hall of the station. The recorded sequences enabled to test the system with rare scenarios of interest (e.g. *fighting*, *jumping over the barrier*, *vandalism*) whereas the live camera allowed to evaluate the system against scenarios which often happen (e.g. *overcrowding*) and which 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 nine *blocking* incidents, seven alarms were generated, giving a detection rate of 78%. Out of 42 instances of *jumping over the barrier*, including repeated incidents, the behaviour 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 in the live camera, with some 28 separate events being well detected.

In conclusion, the ADVISOR demonstration has been evaluated very positively by end-users and European Committee. The algorithm responded very successfully to the input data, with high detection rates, less than 5% of false alarms and with all the reports being above approximately 70% accurate.

7.1.4 Conclusion on using a numerical approach

In this article, we have described a video interpretation system able to automatically recognise high level of human behaviours involving individuals, groups of people and crowd.

Different methods have been defined to compute specific types of behaviours under different configurations. All these methods have been integrated in a coherent framework enabling to modify locally and easily a given method. The system has been fully tested off-line and has been evaluated, demonstrated and successfully validated in live condition during one week at the Barcelona metro in March 2003. The next step consists in designing the video interpretation system to be operational (able to cope with any unpredicted real world event, and to implement dynamic tuning of algorithm parameters), and working on a large scale. For that, we need to design a platform able to be configured dynamically and automatically.

7.2 Temporal Scenario

We can distinguish two main categories of approaches to recognize a scenario based on a symbolic network: the STRS approaches (Stores Totally Recognized Scenarios) recognize scenarios based on an analysis of scenarios recognized in the past [Bremond 34], whereas the SPRS approaches (Stores Partially Recognized Scenarios) recognize scenarios based on an analysis of scenarios that can be recognized in the future [Ghallab 1996]. The STRS approaches recognize a scenario by searching in the set of previously recognized scenarios a set of sub-scenarios matching the scenario model to be recognized. Thus, if the system fails to recognize a scenario, it will have to retry the same process (re-verify the same constraints) at the next instant, implying a costly processing time. A second problem is that STRS algorithms have to store and maintain all occurrences of previously recognized scenarios. The SPRS approaches recognize a scenario by predicting the expected scenarios to be recognized at the next instants. Thus, the scenarios have to be bounded in time to avoid the never ending expected scenarios. A second problem is that SPRS algorithms have to store and maintain all occurrences of partially recognized scenarios, implying a costly storing space.

The method presented in this section is a STRS approach taking advantages of the SPRS approaches. The objective is to reduce the processing time (1) when searching in the past (list of previously recognized scenarios) for an occurrence of a given scenario model and (2) when trying to recognize a scenario involving several actors by avoiding checking all combinations of actors.

In this section, to solve scenario recognition issues, we first propose a language to describe scenario models and second a temporal constraint resolution approach to recognize in real-time scenario occurrences. Our scenario representation is mainly based on the representation of [Bremond 29] and inspired by the work of [Ghallab 1996]. In this section, we focus on the optimization of the recognition method. We first enhance the processing of temporal operators by pre-compiling scenario models to decompose them into simpler scenario models. By this way, the scenario recognition algorithm uses a linear search compared to an exponential search for non-compiled scenarios. Secondly, we propose a novel algorithm to recognize composed scenarios that takes advantages of the actors of its sub-

scenarios when they are recognized instead of trying all combinations of actors as this is often the case for similar state of the art algorithms.

7.2.1 Temporal Scenario Representation

Our goal is to make explicit all the knowledge necessary for the system to be able to recognize scenarios occurring in the scene. The description of this knowledge has to be declarative and intuitive (in natural terms), so that the experts of the application domain can easily define and modify it. Thus, the recognition process uses only the knowledge represented by experts through scenario models.

Let Ω be the set of scenario models and Φ be the set of scenario instances (recognized scenarios). For a scenario model $\omega \in \Omega$ and a scenario instance $\rho \in \Phi$, we note $\omega = \omega(\rho)$ the scenario model of ρ and we note $\rho(\omega)$ the set of recognized scenarios of the model ω. A scenario is composed of four parts:

a) $\alpha(\omega)$ is the set of actor variables (characters) involved in the scenario ω. $\alpha(\rho)$ corresponds to the actors. An actor can be a person tracked as a mobile object by a vision module or a static object of the observed environment like a chair.

b) $\sigma(\omega)$ is the set of sub scenarios that compose ω. ω is an elementary scenario if $\sigma(\omega) = \varnothing$, if not, ω is called a composed scenario. Each sub-scenario is represented by a variable v. These variables are called temporal variables because their value is a recognized scenario defined on a temporal interval. We note $\rho(v)$ a scenario instance corresponding to the value of the temporal variable v, and $\alpha(v)$ the set of variables corresponding to the actors of $\rho(v)$.

c) $\varphi(\omega)$ is the set of sub-scenarios that should not occur during the recognition of the scenario ω. We call $\varphi(\omega)$ the forbidden sub scenarios. $\alpha_\varphi(\omega)$ is the set of forbidden actor variables involved in $\varphi(\omega)$ but not already defined in $\alpha(\omega)$. $\varphi(\omega)$ and $\alpha_\varphi(\omega)$ are called the forbidden variables.

d) $\kappa(\omega)$ is the set of constraints of ω. There are three types of constraints:

- the set of temporal constraints, noted $\kappa^T(\omega)$, on at least one variable of $\sigma(\omega)$ and not on any forbidden variable,
- the set of atemporal constraints, noted $\kappa^A(\omega)$, on only $\alpha(\omega)$,
- the set of forbidden constraints, noted $\kappa^F(\omega)$, on any forbidden variable.

The three subsets $\kappa^T(\omega)$, $\kappa^A(\omega)$ and $\kappa^F(\omega)$ constitute a partition of $\kappa(\omega)$. We use the operator "and" to link the constraints within a set of constraints. To use the operator *"or"*, we propose to define two similar scenario models with different constraints. An elementary scenario is only composed of a set of characters and atemporal constraints.

Figure represents a model of a composed scenario "Bank attack". This scenario involves two actors, a cashier and a robber.

In our representation, any scenario ω involves at least one person, and is defined on a time interval. An interval is represented by its starting and ending times noted *start*(ω) and

end(ω). Defining the scenarios on a time interval is important for the experts to describe scenarios in a natural way.

```
Scenario(Bank_attack,
          Characters((cashier:Person), (robber:Person))
          SubScenarios(
            (cas_at_pos,inside_zone,cashier,"Back_Counter")
            (rob_enters, changes_zone, robber,
               "Entrance_zone", "Infront_Counter")
            (cas_at_safe, inside_zone, cashier, "Safe")
            (rob_at_safe, inside_zone, robber, "Safe") )
          ForbiddenSubScenarios(
            (any_in_branch, inside_zone, any_p,"Branch"))
          Constraints(
           Temporal ((rob_enters during cas_at_pos)
                    (rob_enters before cas_at_safe)
                    (cas_at_pos before cas_at_safe)
                    (rob_enters before rob_at_safe)
                    (rob_at_safe during cas_at_safe))
           Atemporal((cashier ≠ robber))
           Forbidden((any_p ≠ cashier) (any_p ≠ robber)
            (any_in_branch during rob_at_safe))))
```

Figure 7.6: *The scenario "Bank attack" is composed of four steps: (1) the cashier is at his/her position behind the counter, (2) the robber enters the bank and moves toward the front of the counter then (3) both of them arrive at the safe door and (4) nobody else in the branch during the attack.*

7.2.2 Scenario Recognition

The scenario recognition process has to detect which scenario is happening from a stream of observed persons tracked by a vision module at each instant. The recognition process takes also as input the a priori knowledge of the scene and the scenario models. As defined in the previous section, there are two types of scenarios: elementary and composed.

Recognition of elementary scenarios

The algorithm to recognize an elementary scenario model ω_e consists in a loop of the selection of a set of actors then the verification of the corresponding atemporal constraints $\kappa^A(\omega_e)$ until all combinations of actors have been tested. Once a set of actors satisfies all constraints $\kappa^A(\omega_e)$, we say that the elementary scenario ω_e is recognized and we generate an elementary scenario instance ρ attached with the corresponding scenario model, the set of actors and the recognition time t. The scenario instance is then stored in the list of recognized scenarios. If at the previous instant, a scenario instance ρ' of same type (same model, same actors) was recognized on a time interval $[t_0, t-1]$, the two scenario instances are merged into a scenario instance that is recognized on the time interval $[t_0, t]$.

The selection of actors leads the recognition algorithm to an exponential combination in function of the number of actor variables. However, in practice, there are few actor variables in elementary scenario models, so the recognition algorithm is still real-time.

Compilation of composed scenarios

A composed scenario is a sequence of sub-scenarios ordered in time. Each sub-scenario corresponds to a temporal variable in the corresponding scenario model. The STRS algorithms of the state of the art perform at each instant a extensive search process among all possible scenarios and sub-scenarios leading to an exponential algorithm. We propose to decompose the scenario model into a set of simple scenario models containing at most two sub-scenarios.

To compile a predefined composed scenario model ω, we define three steps: (1) build a graph with the temporal variables $\sigma(\omega)$, (2) generate intermediate scenario models for ω and (3) link the generated intermediate scenario models based on the constraints $\kappa(\omega)$.

As proposed by [Ghallab 1996], we first build a graph which nodes correspond to the temporal variables and which arcs correspond to the temporal constraints $\kappa^T(\omega)$. The arcs are oriented and are associated with a time interval corresponding to the time delay between the ending time of the two variables. For example, the constraint c_i between v_i, v_j is associated with an interval [a, b] indicating that v_j can end in the interval [*end*(v_i)+a, *end*(v_i)+b]. The constraint *before* is associated with [1, ∞]. After building the graph (called initial graph) with all temporal constraints between temporal variables $\sigma(\omega)$, we compute the equivalent complete graph and we check the graph consistency. These two graphs are equivalent because the only difference between them is the redundancy of some constraints. Then, we simplify the complete graph by removing unnecessary arcs to obtain the least constrained graph. These two graphs are also equivalent for the same reason. The initial and simplified graphs for the scenario *"Bank attack"* (Figure) are shown on Figure **Erreur ! Source du renvoi introuvable.**. Thanks to the simplified graph, all the temporal variables $\sigma(\omega)$ are ordered by their ending time.

Second, we generate intermediate scenario models composed at most of two sub-scenarios. For each intermediate scenario model ω, we call *start* (noted $\xi(\omega)$) the first sub-scenario of ω; and we call *termination* (noted $\tau(\omega)$) the second sub-scenario of ω.

As $\sigma(\omega) = (v_1, v_2, \ldots, v_n)$ is a sequence of n (n > 2) ordered temporal variables, we generate n-1 intermediate models $\omega^1, \omega^2, \ldots, \omega^{n-1}$ as followed:

a) $\sigma(\omega^1) = (v_1, v_2)$ and $\sigma(\omega^i) = (v^i, v_{i+1})$ for i > 1, where v^i is a new temporal variable corresponding to the scenario model ω^{i-1},

b) $\alpha(\omega^i) = \alpha(\xi(\omega^i)) \cup \alpha(\tau(\omega^i)) = \alpha(v_1) \cup \ldots \cup \alpha(v_{j+1})$,

c) $\varphi(\omega^{n-1}) = \varphi(\omega)$ and $\varphi(\omega^i) = \varnothing$ for i < n-1,
$\alpha_\varphi(\omega^{n-1}) = \alpha_\varphi(\omega)$ and $\alpha_\varphi(\omega^i) = \varnothing$ for i < n-1,

d) $\kappa^A(\omega^1) = \kappa^A(\omega) \cap \kappa(\alpha(\omega^1))$ and $\kappa^A(\omega^i) = \kappa^A(\omega) \cap \kappa(\alpha(\omega^i)) - \kappa(\alpha(\omega^{i-1}))$ for i > 1

$\kappa^T(\omega_i)$ contains the constraints corresponding to the arcs entering v_i+1 (i.e. $\tau(\omega_i)$) in the simplified graph.

$\kappa^F(\omega^{n-1}) = \kappa^F(\omega)$ and $\kappa^F(\omega^i) = \varnothing$ for i < n-1.

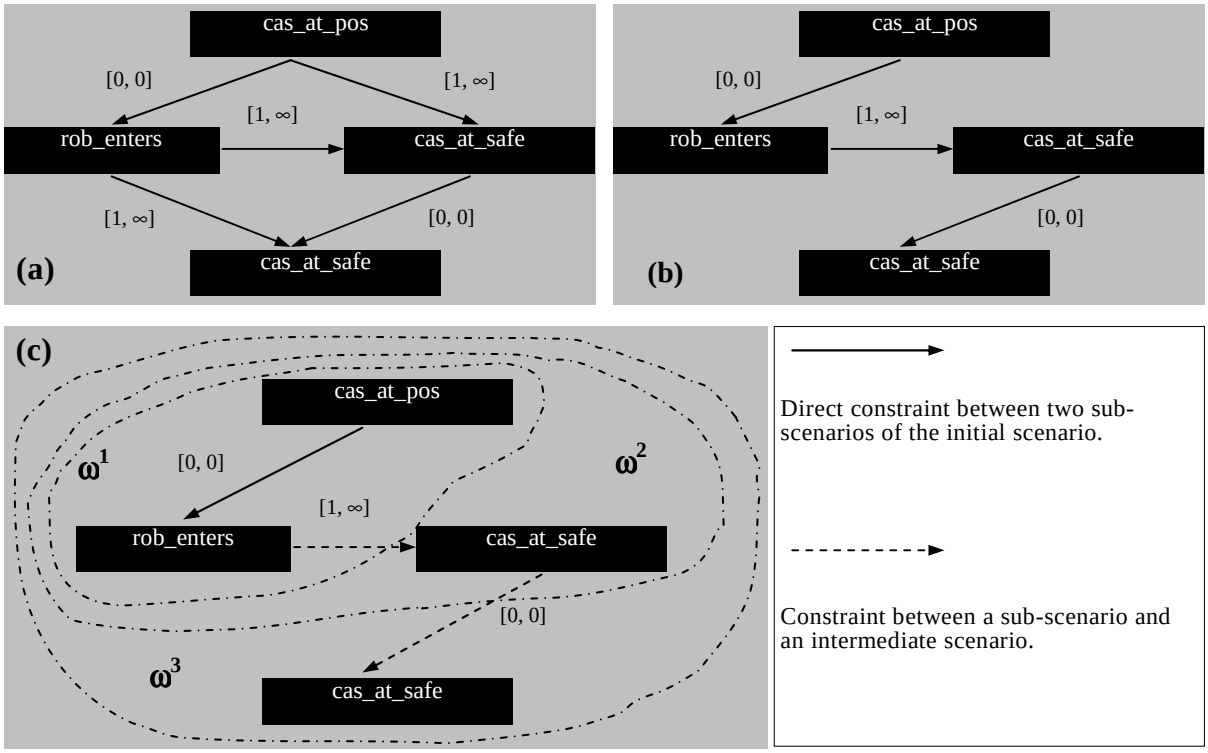


***Figure 7.7**: To compile the scenario "Bank attack", we build: (a) the graph with all temporal constraints $k^T(\omega)$, (b) the simplified graph with only the necessary constraints, the generated intermediate scenario models (c).*

The resulting scenario model ω^{n-1} is equivalent to the initial scenario model ω. This two scenarios have the same actor variables and equivalent set of constraints. The only difference is that the constraints of the scenario ω are verified at several intermediate levels corresponding to the intermediate scenario models as shown on Figure .

By using this compilation method, we can decompose all scenario models into scenarios containing only one or two temporal variables. The recognition of compiled scenario models is described in the next section. The gain in processing time is due to the search algorithm: we just try several times to link two scenario instances instead of trying to link together a whole set of combinations of scenario instances.

```
Scenario(Bank_attack_1, # ω1
Characters((cashier:Person), (robber:Person))
SubScenarios(
        (cas_at_pos, inside_zone, cashier,
              "Back_Counter")
        (rob_enters, changes_zone, robber,
        "Entrance_zone", "Infront_Counter"))
```

```
Constraints((cas_at_pos during rob_enters)
        (cashier ≠ robber) ))


Scenario(Bank_attack_2, # ω²
Characters((cashier:Person), (robber:Person))
SubScenarios(
        (att_1, Bank_attack_1, cashier, robber)
        (cas_at_safe, inside_zone, cashier, "Safe") )
Constraints(
        ((start of att_1) before cas_at_safe) ))

Scenario(Bank_attack_3, # ω³
Characters((cashier:Person), (robber:Person))
SubScenarios(
        (rob_at_safe, inside_zone, robber, "Safe") )
ForbiddenSubScenarios(
        (any_in_branch, inside_zone, any_p,"Branch"))
Constraints(
        (rob_at_safe during (termination of att_2))
        (any_in_branch during rob_at_safe) ) )
```

***Figure 7.8**: Three intermediate scenario models are generated for the compilation of the scenario model "Bank_attack", and the initial model is equivalent to "Bank_attack_3".*

Recognition of composed scenarios

The recognition of a composed scenario model ω_c is triggered by a scenario template, which has been generated when the last sub-scenario ρ_t terminating ω_c has been recognized. The scenario template contains ω_c and the scenario instance ρ_t with its list of actors $\alpha(\rho_t)$ that partially instantiates $\alpha(\omega_c)$. As ω_c is composed of two sub-scenarios, only one sub-scenario ρ_s starting ω_c still needs to be found.

If such a scenario instance ρ_s has been previously recognized in the past, then we are able to finish instantiating the remaining actors of $\alpha(\omega_c)$. Thus, just a few combinations of actors need to be checked avoiding an exponential search.

The last step of the algorithm consists in verifying whether all temporal and atemporal constraints ($\kappa^T(\omega_c)$ and $\kappa^A(\omega_c)$) are satisfied with ρ_s and ρ_t. If one forbidden constraint of $\kappa^F(\omega_c)$ cannot be satisfied then the scenario ω_c is recognized and stored in the list of recognized scenarios.

Discussion

In the domain of temporal scenario recognition and among SPRS algorithms, the chronicle recognition algorithm [Ghallab, 1996] is one of the most popular. By storing partially recognized scenarios, it can speed up the whole recognition process. A partially recognized scenario corresponds to a prediction and enables to store all previous computations that do not need to be recomputed at the following instants. A main difference between the chronicle algorithm and our algorithm is that the chronicle algorithm has been developed to process scenarios defined with only one "actor" and can only recognize events detected at one time point. Thus, this algorithm is efficient for the configuration "mono-actor". However, in the configuration "multi-actors", the chronicle algorithm has to duplicate

the partially recognized scenarios for each combination of actors not already instantiated. This can lead to an explosion of memory allocation and to an exponential search. Our algorithm is as efficient for the configuration "mono-actor" because we store the recognized scenarios which have been compiled. Moreover, it is efficient for the configuration "multi-actors" because the recognized scenarios do not need to be duplicated even if some actors are not instantiated. The worst case occurs with elementary scenarios because to recognize them at the current instant, all combinations of actors need to be checked. In real world applications, elementary scenarios do not contain many actor variables (less than 3) making the proposed algorithm sufficient to obtain an operational real time video interpretation system.

7.2.3 Experiments and results

To validate our recognition algorithm, we first integrated the algorithm with a vision module to obtain an operational interpretation system and then we have realized four types of tests: (1) on recorded videos taken in a bank branch and in a metro station to verify if the algorithm can correctly recognize the predefined scenario models, (2) on live videos acquired on-line from cameras installed in an office and in a bank branch to verify if the algorithm can work robustly on a long time mode, (3) on recorded videos taken in a bank branch and on simulated data to study how the complexity of the algorithm depends on the scenario models (i.e. number of sub-scenarios and of actor variables) and (4) on simulated data to study how the complexity of the algorithm depends on the complexity of the scene (i.e. number of persons in the scene).
In the first experiment, we verify on recorded videos that the algorithm correctly recognizes several types of *"Bank attack"* scenarios and several types of *"Vandalism against a ticket machine"* scenarios.

Table 7. *1* shows that the predefined scenarios were correctly recognized in most of the cases. The interpretation system fails to recognize some scenarios only in the cases when the vision module misses to detect the people in the scene. We have not detected any false alarm during all the experiment. The non-detection of false alarms can be explained by the fact that the scenarios are very constrained and there are unlikely to be recognized by error.

In the second experiment, we installed the interpretation system in an office and in a bank and we connected the system to two on-line cameras to acquire directly live videos. In this experiment, we use the bank scenarios and we slightly modified them to use them in the office. We ran the system in the bank for few hours and continuously during 24h in the office. As in the first experiment, the scenarios were most of the time correctly recognized, showing that the recognition algorithm can work reliably and robustly in real-time and in continuous mode.

	Number of tested sequences	Average number of persons/frame	Recognition rate (%)	Number of false alarms

Bank cam. 1	10	4	80	0
Bank cam. 2	1	2	100	0
Metro cam. 2	3	2	100	0

Table 7. 1*. The recognition of temporal scenarios in videos of a bank branch and of a metro station.*

In the third experiment, we studied the processing time of the recognition algorithm in function of the scenario models. First, we studied the processing time of the algorithm focusing on the resolution of temporal constraints. In this experiment (shown on Figure), we tested eight configurations of scenario models: the first configuration is made of scenarios containing 3 sub-scenarios and the last configuration is made of scenarios containing 10 sub-scenarios.

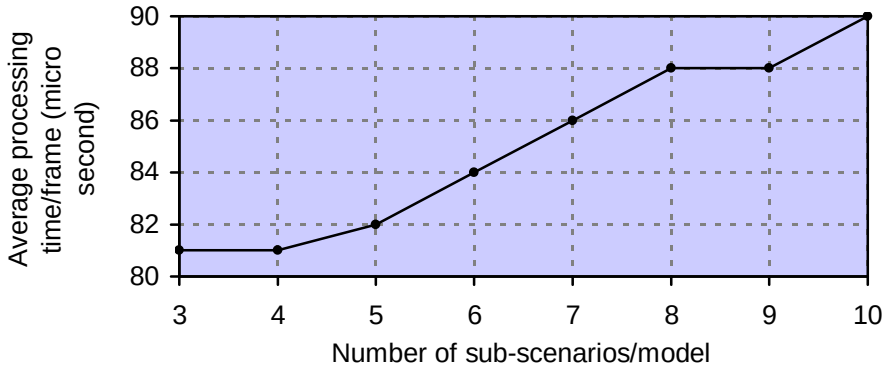


Figure 7.8 *: The processing time of the new algorithm is closely linear in function of the number of sub-scenarios.*

On the bank videos containing 300 frames, we found that the processing time of the classical STRS algorithm is exponential in function of the number of sub-scenarios, whereas the processing time of our algorithm is closely linear in function of the number of sub-scenarios.

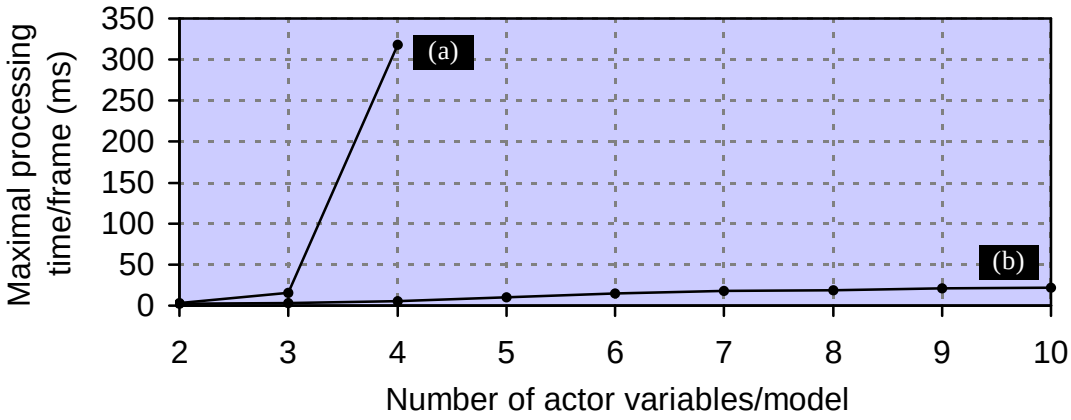


Figure 7.9. *The processing time (a) of the old algorithm and (b) of the new algorithm depend on the number of actor variables of predefined scenario models.*

Second, we studied the processing time of the algorithm focusing on the number of actor variables of the scenario models. In this experiment (shown on Figure), we tested nine configurations of scenario models: the first configuration is made of scenarios involving 2 actor variables and the last configuration is made of scenarios involving 10 actor variables. To run the algorithm with enough actors, we simulated bank videos containing 35 persons. On these videos, we found that the processing time of the classical STRS algorithm is exponential in function of the number of actor variables, whereas the processing time of our algorithm is closely linear in function of the number of actor variables.

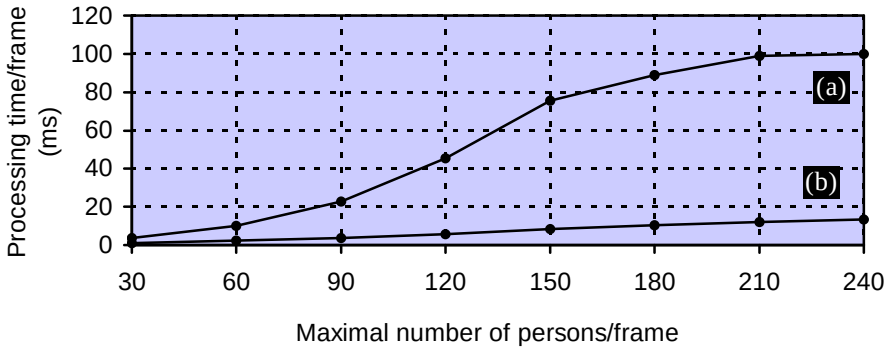


Figure 7.10. *The (a) maximal and (b) average processing time/frame of the new algorithm depend on the number of detected persons.*

In the fourth experiment, we studied the processing time of the recognition algorithm in function of the scene. To have a continuous variation of the scene, we simulated the scene. We built a scene environment with eight zones of interest and ten pieces of equipment. We simulated the individuals evolving in the scene at each instant. In these simulated videos, the number of individuals changed from 30 up to 240. To verify if our algorithm can recognize in real-time the predefined scenarios, we measured the maximal processing time per frame. We found that, the maximal processing time for each frame is 100ms for a scene of 240 persons. We also found that the average processing time for each frame is closely linear in function of

the number of persons. Figure shows several tests of this experiment to illustrate how the processing time depends on the complexity of the scene.

With the fourth experiment, we can conclude that our recognition algorithm can recognized in real-time the predefined scenarios if the number of persons/frame is less than 240. All these tests are realized on a PC linux: CPU 700MHz, 320MB RAM.

7.2.4 Conclusion on Temporal Scenario

In the literature, there are two categories of symbolic approaches to recognize temporal scenarios: the STRS algorithms reasoning on the past and the SPRS algorithms reasoning on the future. First, we have shown that the STRS algorithms recognize usually a scenario by performing an exponential combination search. Then, we have explained that even if our proposed algorithm is a STRS algorithm, it checks temporal constraints nevertheless by performing a linear search thanks to a step of pre-compilation of scenarios. Second, we have also shown that the SPRS algorithms have to try all combinations of actors to be able to recognize "multi-actors" scenarios. Thanks to the pre-compilation step this drawback for our algorithm is limited to elementary scenarios. For these two reasons, the proposed algorithm enables the integrated video interpretation system to be real-time. Up to our knowledge, this video interpretation system is the first operational system able to recognize complex temporal scenarios.

Our future work consists in taking care of the errors and the uncertainty of the vision module. The goal is to be able to continue the interpretation of the videos even when the vision module cannot cope with the real world complexity.

7.3 Conclusion on Event Recognition

In this chapter, we have presented the two types of approach for recognising events: numerical and symbolic. The numerical approaches are well adapted to events closely related to vision features involving few actors (mostly one actor with or without interaction with his/her environment). However, their effectiveness depends on the training stage as the parameters are learned with training video sequences containing positive and negative event samples. Moreover, developing these recognition algorithms is not straightforward and required a strong expertise in vision. They are relatively easy to implement but their tuning for a particular application is a main problem. In addition, they suffer from a difficult modelling stage of events involving multiple actors (i.e., objects of interest). The reason is that the combination of events is often exponential given the number of actors, leading to the utilization of a huge training set. Finally, their description is not declarative and it is often

difficult to understand how they work (especially for Neuronal Networks). In consequence, it is relatively difficult to modify them or to add a priori knowledge. Therefore, new learning mechanisms need to be defined to ease the construction of event recognition algorithms: to define the training video sets and to learn the structure and the parameters of the network. This point will be discussed in the next chapter.

Symbolic approaches are well adapted to model complex temporal events involving multiple actors. Despite a large number of potential combinations of events to be explored, well designed algorithms can still recognise events in real-time. When good vision results are obtained, the symbolic event recognition algorithms can recognise all scenarios. An intuitive language has been also defined to help the end-users to describe their scenarios of interest. The main problem of symbolic approaches is the mechanism to handle the errors of vision algorithms. Most of the time, the recognition algorithms take the hypothesis that vision algorithms do not make errors and generate perfect tracked objects. Moreover, modelling the scenarios selected by the end-users is an error prone process (i.e. models of scenario are often vaguely defined), which can be time consuming, especially in case of monitoring all everyday activities. Therefore, two main improvements still need to be done. First, managing the uncertainty of vision features (in particular the lost of tracked objects), is a crucial point. Second, learning the models of scenario becomes essential while dealing with video monitoring applications. This second point will be discussed in chapter 9.

Chapter 8

Performance evaluation and learning

To be able to improve scene understanding systems, we need at one point to study vision system architecture [Remo and et al., 2003], [Greenhill et al., 2002], [Velstin and Remagnino, 2005], [Davies and Velastin, 2005], [Cucchiara et al., 2004], [Enficiaud et al., 2006] and to evaluate their performance [Borg et al., 2005], [lazarevic-McManus et al., 2006], [Hall et al., 2005], [Desurmont et al., 2006], [Manohar et al., 2006]. The classical methodology for performance evaluation consists in using reference data (called ground truth) [Bashir and Porikli, 2006], [Thirde et al., 2006], [Li et al., 2006], [Ziliani et al., 2005]. However, generating ground truth is tiresome and error prone. Therefore, an issue is to perform the performance evaluation stage using unsupervised techniques. During European projects, such as ADVISOR and AVITRACK [Bremond, 3 and 12], we have acquired a long experience in evaluating the performance of particular video understanding systems. Moreover, during the PhD of Benoit Georis [Bremond, 13, 37 and 38] and the ETISEO program on performance evaluation, we have proposed an accurate definition of ground truth, a set of metrics, and video criteria to get a meaningful assessment of the performance of video understanding programs.

Once evaluation is possible, a real challenge consists in optimising the scene understanding system using machine learning techniques in order to find the best combination of programs, the best set of parameters of these programs with the best control strategies to obtain an efficient and effective real-time process. The difficulty is three fold: first, programs depend on environmental conditions and this optimisation process has to be dynamic to take into account environmental changes. Second, all these programs are interlinked with each others, so the modification of one program parameter can mess the functioning of all other programs. Finally, the knowledge on these programs is not formalised and usually, even the

developers cannot tell what will be the program output under even specific conditions. During Benoit Georis PhD [Bremond 48], we have defined a framework to formalise the construction of video understanding systems by specifying all the knowledge required: knowledge of the program, knowledge of the scene environment and end-users goals. This framework is intended to ease the building, the configuration, the tuning and the evaluation of video understanding systems.

Another way to improve system performance is to add higher reasoning. Scene understanding is essentially a bottom-up approach consisting in abstracting information coming from signal (i.e. approach guided by data). However, in some cases, a top-down approach (i.e. approach guided by models) can improve lower process performance by providing a more global knowledge of the observed scene. For instance, the global coherence of the 4D world can help to decide whether moving regions correspond to noise or physical objects of interest. During most of past projects, such as SAMSIT [Bremond, 51] we have tried to incorporate high-level feedback to help with the detection and tracking of physical objects of interest. However, a convenient formalism still needs to be defined to make these experiments seamlessly integrated in new scene understanding systems.

So this chapter, mostly based on Benoit Georis PhD [Georis, 2006] and [Bremond, 13], consists in exploring performance evaluation and machine learning techniques for the easy generation of effective real-time scene understanding systems. The first section describes supervised (with user interactions) and unsupervised (fully automatic) evaluation mechanisms with the goal to get an insight on the evaluated algorithms. This work has also been supported by the ETISEO program with the help of Valery Valentin a research engineer. The second section presents preliminary results on learning techniques for scene understanding algorithms. The goal is mostly to learn the best parameter set of a given algorithm, knowing a characterisation of the scene.

8.1 Performance Evaluation

The analysis of existing evaluation techniques shows that few of them enable to really improve performances. This is due to the fact that they lack standard evaluation criteria and also a clear evaluation protocol. Most of the time, they define evaluation criteria in function of the particular application of interest. However, the essential issue comes from the fact that performances depend on the video sequences used for testing, on the hypotheses which have guided the design of programs and on all the parameters. Thus, it is very difficult to understand results of other systems even if they are evaluated under the same assumptions. In consequence, in order to be able to diagnose problems and to take advantage of the evaluation results, the only possibility is to test a system by varying one parameter at a time, all others being fixed.

In parallel, several workshops and projects have been created to answer to this increasing need of having effective evaluations. The overall trend is to propose automatic tools allowing comparing results with reference data. These automatic tools enable to produce significant quantitative results on standard test video sequences. However, a standard

database does not exist yet and the ground truth definition is a hard task. In addition, there is a large variability of end-user needs. In consequence, we need a general evaluation framework to get better insights on the video understanding approaches and on current technical issues. In the same way, we need a flexible evaluation tool which enables to acquire expertise on the video processing programs composing a video understanding platform. The evaluation tool should be parameterized in function of the task to evaluate, the target application and the environmental description. An example of flexibility which is required is the ability to select the part of the ground truth which is relevant to evaluate a given task. For instance, to evaluate a face tracker, we do not need a bounding box but an ellipse delineating the face. On the opposite, in a people counting application, the bounding box or just the centre of gravity of the objects is often sufficient.

8.1.1 Supervised Evaluation

By definition, a supervised evaluation performs an assessment of the quality of the obtained results by comparing them to the ideal expected results. Most of the time, these expected results are given by a human and are called *ground truth data*. The scheme of a supervised evaluation is illustrated in Figure .

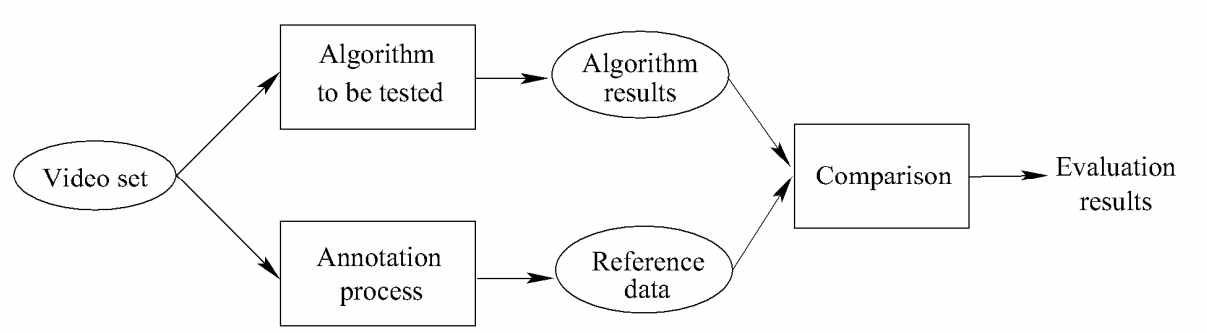


Figure 8.1*: An algorithm is tested on a set of video sequences by comparing its output results with reference data.*

This scheme can be applied to most video understanding problems but may not be easily applicable with PTZ cameras due to the problem of defining a correct ground truth (i.e., which position of the camera is best for tracking which object).

The objective of the supervised evaluation is twofold. First, this evaluation allows a system endowed with an intelligent control to configure itself according to its environment (e.g., selection of algorithm, initial parameter setting). In this case, the system is run offline on recorded video sequences which are representative of the future conditions of use of the system. Second, this evaluation allows the video processing experts to compare various techniques in competition and to run extensive series of testing for their programs on various

video sequences and environmental conditions in order to acquire expertise. This expertise can then be explicitly written using a video processing program formalism [Bremond, 13 and 48], [Moisan and Thonnat, 2000]. For instance, this expertise can take the form of a precondition of utilization of a program (e.g., not valid for outdoor sequences) or a rule to adapt a parameter according to a characteristic of the environment (e.g., set the threshold value to twice the default value if the mean image luminance is greater then 150).

Although the evaluation can be interactive (i.e., the system prompts a user to assess results at each processing step) and useful in the configuration mode of the system, we recommend to use an automatic evaluation. The reason is that we must ensure the significance of the evaluation results and to prevent to find a too specific piece of knowledge due to the analysis of a too particular situation or problem. An automatic evaluation is thus a good way to test several long video sequences but implies the tedious task of defining ground truth.

Ground truth, reference data and annotations

To perform an effective and useful evaluation of a technical issue, it is important to distinguish the three following concepts: *ground truth data*, *reference data* and *annotation*, even if they are often confused or mixed in the literature:

- **Ground truth data:** data given by a human operator and which describe real world expected results (e.g., objects of interest, events) at the output of a video processing program. These data are supposed to be unique and corresponding to end-user requirements even if in many cases, this information can contains errors (annotation bias). These data can be written in a various formats, such as XML or MPEG7.
- **Annotations:** information associated to a video sequence. They inform about the video content, but also about technical difficulties (e.g., presence of shadows) and recording conditions (e.g., weather conditions) of the video sequence under consideration. These annotations can help to get a more precise classification of false or incorrect results (e.g., wrong classification, wrong detection) and thus can help the expert in his/her knowledge acquisition objective.
- **Reference data:** data supposed to be constant and unique, corresponding to a functionality of a video processing task and used to evaluate the output of a video processing program at a given task level. Reference data include ground truth data, information given by a video expert and data computed from all annotations and contextual information. For instance, the 3D position of a person is a reference data computed from the bounding box given by a video expert and the calibration matrix.

Ground truth generation

The generation of ground truth consists in defining the ideal output of a video processing functionality. This is not as trivial as it first seems. First, the level of details of ground truth attributes depends on the functionality which has to be evaluated. For instance,

the evaluation of a motion detection algorithm ideally requires a pixel-based ground truth whereas the evaluation of a tracking algorithm may only use a list of bounding boxes. Of course, if we want more details, the process of annotation becomes more tedious and time-consuming.

Second, the ground truth definition is a subjective process. There are several key questions a human has to answer before he/she can define ground truth. For instance, is it better to draw the bounding box of the whole object when this object is occluded? Is it better to draw two bounding boxes when two people walk very close to each other or to draw only one for the group? These choices have to be made with respect to the target application and the features to evaluate. What is important is to stay coherent and to give rules to persons making the annotation in order to define as objectively as possible particular data.

We suggest giving as much precise information as we can during the annotation. The main reason is that we must be able to evaluate the most sophisticated algorithms (existing ones as well as future ones, in order to avoid redefining over time several ground truth on the same video sequences) and being impartial for all algorithms. For instance, we have chosen to draw the full bounding box for each object even when it is dynamically or statically occluded. In this way, we are able to evaluate all types of detection algorithms. In case the algorithm it does not need to re-compute accurately the detection of objects, it is always possible to compensate the detection errors by re-computing the observable parts of the mobile objects. In case the algorithm wants to correct detection results and process a whole object (e.g., to recover the feet of a person even if this person is partially observable due to an occlusion), we are able to determine whether the object detection task has correctly labelled the person as *occluded*. Of course, due to the fact that the coordinates of invisible parts are guessed, this choice can lead to imprecision but they are not significant when performing an evaluation at the level of details of the bounding box.

In addition, care must be taken to avoid introducing other bias when defining ground truth. For instance, the assessor may be aware of typical video processing problems and thus define the ground truth to increase performances. Moreover, it is important to annotate at a task level without using any knowledge or information which are not available at that level. For instance, the assessor must not draw a bounding box for an object on a frame if he/she does not see this object on this frame, even he/she can guess its presence by looking at the whole sequence (due to the evidence of motion). Concerning the annotation of high-level tasks, it is sometimes difficult to annotate accurately the starting time of an event (e.g., the person began to fight at frame 150 and not a frame 148) because of the time shift between the beginning of an action and the presence of the corresponding visual evidences.

Finally, we want to point out that ground truth can also help to classify video sequences (see previous chapter). Indeed, instead of manually creating classes for classifying the variability of video sequences according to a criterion hardly computable automatically (e.g., the number of objects in the scene), we can extract the information from the ground truth and classify automatically the set of video sequences with respect to this criterion.

Several performance indicators

Once video sequences have been collected and ground truth has been defined, video processing experts have to define evaluation criteria and metrics for each functionality which has to be evaluated:

- **Evaluation criterion:** an evaluation criterion is an evaluation functionality to compare video understanding algorithm results. Usually, this comparison is performed with reference data. For instance, for the detection of physical objects of interest, a criterion can correspond to the evaluation of the accuracy of the 2D or 3D object location; another one can correspond to the evaluation of the quality of the object shape. For the classification of physical objects of interest, a criterion can correspond to the evaluation of the quality of the assigned class labels. In addition, these criteria can be detailed with regard to video clip categories. For instance, in the previous example, the assignment of class labels could be evaluated under static occlusion situations.

- **Evaluation metric:** a distance between video understanding algorithm results and reference data implementing an evaluation criterion. A way of displaying evaluation results is to use a ROC (Receiver Operating Characteristic) curve defined as a plot of the true positive rate against the false positive rate. Each sample point of the curve is obtained with a particular parameter setting of the algorithm under evaluation.

Based on these metrics, four evaluation statistics can be computed:

- **True Positive (TP):** a result which is confirmed by a reference data.
- **False Positive (FP):** a result which does not match any reference data (i.e., a false alarm).
- **False Negative (FN):** a result which is missing with respect to reference data.
- **True Negative (TN):** an absence of result which correctly corresponds to the absence of reference data.

Most of the time, true negatives are difficult to compute. For a detection task, it is possible to compute this number provided that the ground truth is defined for each pixel. On the opposite, it is difficult to define the notion of true negative for a tracking functionality. For instance, this would require defining the complementary set of the set of links between objects belonging to reference data.

We are now able to perform the evaluation. In order to analyse evaluation results, performance indicators are required. We propose here several concepts which are currently under acceptance of the scientific community [Etiseo]:

- **Precision:** the precision of an algorithm is defined as the percentage of good results among all computed results, i.e., the number of true positives over the sum of true positives and false positives. For instance, the precision can measure the reliability of a technique under given environmental conditions.
- **Sensitivity**: the sensitivity of an algorithm is defined as the percentage of good results among all expected results, i.e., the number of true positives

over the sum of true positives and false negatives. For instance, the sensitivity can indicate whether or not a technique is able to achieve the target objective in given conditions.

- **Specificity:** the specificity of an algorithm is defined as the percentage of correctly detected noise among all noise, i.e., the number of true negatives over the sum of true negatives and false positives. For instance, the specificity can measure the robustness of a technique when it is applied beside the preconditions. Nevertheless, this notion is more complex to use as it requires the computation of true negatives.

The choice of the indicator may depend on the target objective. For instance, if the expert wants to optimize a technique to obtain a maximum detection rate (e.g., ≥ 90%), he/she will prefer the sensitivity. On the opposite, if the objective is to obtain as less false alarms as possible then precision should be chosen equal to *1*. A synthetic view of these concepts and their relationships is illustrated in Table 1.

	Reference Data (RD)	Noise (N)	
Computed	True Positive (TP)	False Positive (FP)	Precision (TP/(TP+FP))
Not Computed	False Negative (FN)	True Negative (TN)	
	Sensitivity (TP/(TP+FN))	Specificity (TN/(TN+FP))	

Table 8.1*: Performance indicators and their relationships.*

8.1.2 A Supervised Evaluation Tool

As said before, video understanding systems need to be validated in order to verify that it produces satisfactory results regarding end-user requirements. Moreover, performance evaluation is a main way to acquire a deep expertise on programs and to be able to increase incrementally the knowledge bases of the systems. In this section, we present a supervised evaluation tool and the associated metrics to compare the results with the reference data.

The supervised evaluation tool we have proposed is intended to four main purposes. First, this tool enables to compare the performances of different algorithms on a set of test video sequences. Second, this tool enables to quantify the improvement in performances of an algorithm over time. Third, this tool enables video processing experts to acquire a deep expertise on programs and on their use (e.g., how to tune program parameters). Finally, this tool enables to verify that a constructed system produces satisfactory results corresponding to end-user requirements. This supervised evaluation tool is illustrated in Figure .

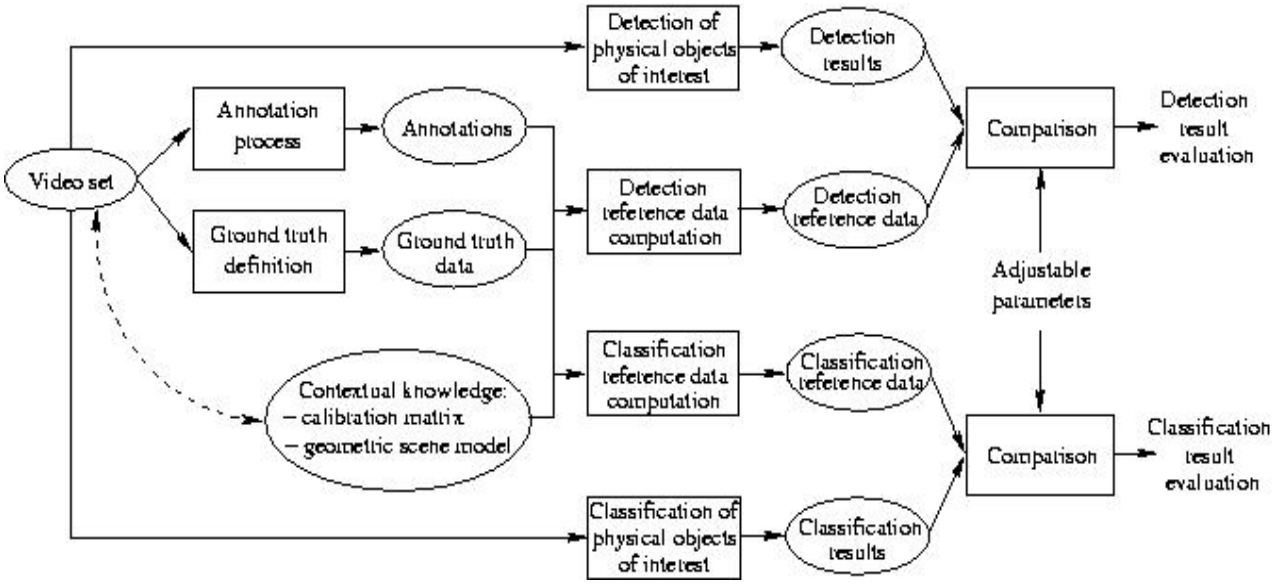


Figure 8.2: *First, video processing experts define annotations and ground truth data on the whole set of test video sequences. Then, for each video processing task, an automatic tool computes reference data, using annotations, ground truth data and contextual information corresponding to the video sequences. Finally, algorithm output results are automatically compared with the corresponding reference data. This process is here illustrated for two different tasks: the detection and the classification of physical objects of interest.*

The first step consists in defining ground truth data and annotations on the selected set of test video sequences. This manual operation is performed only once by video processing experts. Then, an automatic tool computes reference data for each video processing task which corresponds either to a primitive or a composite operator. These reference data are computed from ground truth data but also from contextual information and annotations. For instance, the expert does not provide any 3D information when defining ground truth data. However, 3D information is useful to evaluate the accuracy of the 3D object location estimation. This information is thus automatically computed from ground truth data and the calibration matrix. Another illustration is the knowledge of reference data which are statically occluded. Instead of asking an expert to assign a *static-occlusion* label to these reference data, it is faster and more accurate to automatically compute statically occluded reference data from ground truth data and the knowledge of the scene geometry (including static contextual objects). Once reference data are available for a video processing task, the evaluation consists in comparing output results with these reference data. This comparison can be tuned according to the evaluation objective. For instance, if the expert wants to study the effectiveness in close field of an algorithm which detects physical objects of interest, an adjustable parameter may indicate to not take into account reference data which are too far from the camera location. Another illustration is the study of static occlusions. In this particular case, the expert asks the comparator to only compute statistics for reference data which are statically occluded.

In conclusion, this adjustable tool enables experts to deeply study a particular problem in order to acquire expertise on it (e.g., which is the best technique to solve this problem, what parameterization is the most appropriate given the environmental conditions specified in the annotation). In other words, experts are able to study one class of problems at a time by decomposing the whole complex video understanding problem with a fine granularity. There are actually four directions for the decomposition: the goal to achieve (i.e., which functionality for the system), the scene environment (described in the annotation), the video

data (described in the annotation) and the task under evaluation. Each intersecting point between these four decomposition axes defines a problem to study. For instance, we can study the impact of shadows in people outdoor scenes on the tracking process for the recognition of a fighting event. Of course, the finer the decomposition is, the most effective is the evaluation. Future investigations will achieve finer decompositions. We can now describe the tool in more details.

Ground truth definition

In order to perform the comparison, we first have to define the necessary comparison material which is ground truth. To this end, we use, among all the available ground truth acquisition tools, the ViPER software from the University of Maryland [Doermann and Mihalcik, 2000], [Viper], [Mariano et al, 2002]. This tool enables to draw bounding boxes and to assign user-defined semantic information to objects (e.g. *occluded*, *person*,...). Following the framework guidelines, we have several rules for defining ground truth:

1. Define precisely the starting and ending time of the video sequence for which ground truth is defined. It can be the exact duration of a clip (i.e., occurrence of an event) or a little bit more than a clip (e.g., few seconds before and after the occurrence of an event).

2. Select a reference image of the scene corresponding to the video sequence under consideration (i.e., with respect to the starting and ending time). The scene is either an empty scene or a scene with physical objects. This step is important and intended to determine which physical objects are considered as contextual objects belonging to the background. A good example is a parked car: either the car has been parked for hours (i.e., before the starting time and after the ending time) and is thus a contextual object or the car has just been parked and will move during the selected video clip and is thus a physical object of interest that must be defined in the ground truth.

3. Define a bounding box for each physical object of interest which has at least a visible part. In other word, we draw the full bounding box even when objects are statically or dynamically occluded. However, we do not draw a bounding box for an object we do not see at all. This rule is related to the next one.

4. Do not define a bounding box for a physical object of interest which is not visible when looking at a single image (i.e., do not use temporal information to guess an object).

5. Assign a unique identifier for a physical object of interest, even if the object disappears temporarily from the field of view. Unlike the previous rule, the identifier is related to a video processing task which uses temporal information. It is thus normal that a human expert uses this information too (i.e., his/her memory).

An example of a bounding box definition using the ViPER graphical user interface is illustrated in Figure . The white person is statically occluded by the desk. Nevertheless, we can see that the full bounding box has been drawn, the feet position being guessed.

Figure 8.3: *Bounding box definition using ViPER graphical user interface.*

Each physical object of interest is featured with the following attributes: 2D width and height, 2D position and an identifier. These attributes are stored in a file in the ViPER XML format.

Beside ground truth data, we provide several annotations of the video sequences:

- Scene type: *indoor* or *outdoor*.
- Number of physical objects of interest: *few, a lot of, crowd*.
- Frequency of crossings between objects: *few, a lot of*.

Finally, we compute reference data from ground truth data, contextual information and annotations. For instance, we determine static occlusions by re-projecting the 3D bounding box of a contextual object in the image plane and then computing whether or not there is an overlap with any ground truth bounding box. The relative position of the overlap informs us on the type of occlusion (e.g., left occlusion, bottom occlusion,...). We can note that this computation is possible thanks to the full bounding box definition in the ground truth. Now that reference data are available, we are ready to assess the performances of video processing programs by comparing their output results with these reference data.

Evaluation criteria and metrics

Evaluation criteria and metrics are the ingredients of the comparison. A criterion is an evaluation functionality whereas a metric is a distance implementing an evaluation criterion. Different metrics can be used for a same criterion and can be selected by an adjustable parameter of the comparator. We list hereunder several criteria and corresponding metrics for the main video processing tasks:

1. **Detection of physical objects of interest:**
    - Criterion qualifying the number of blobs corresponding to physical objects of interest: the metric checks if the overlapping area between blobs and reference data is above a given threshold. A true positive is a reference data having a sufficient overlap with blobs. A false positive is a blob having no sufficient overlap with reference data. A false negative is a reference data having no sufficient overlap with blobs.

    - Criterion qualifying the area corresponding to physical objects of interest, globally in an image: the metric computes for each image a per pixel statistic based on bounding boxes. Percentages for a video clip are computed as the sum of percentages per image divided by the number of images containing at least one reference data. A true positive is a pixel belonging to both the reference data set and the blob set. A false positive is a pixel belonging to the blob set but not to the reference data set. A false negative is a pixel belonging to the reference data set but not to the blob set.

2. **Classification of physical objects of interest:**
    - Criterion qualifying the classification of physical objects of interest: different metrics can be used to count the number of physical objects having a correct type. A first metric only takes into account objects which are correctly detected in order to obtain true positives, false positives and false negatives. A second metric makes a distinction between mis-classifications due to classification shortcomings (e.g., unknown) or due to detection shortcomings (e.g., lack of contrast). A third metric compares the physical objects of interest at different levels of the class hierarchy. For instance, a motorcycle classified as vehicle is considered a true positive at a higher level of the class hierarchy even if it is a false positive at a lower level. Finally, a last metric checks the ability of separating an object of composed type (e.g., group of people) in two or more objects of elementary type (e.g., person) when the video clip contains several physical objects side by side.

    - Criterion qualifying the properties of physical objects of interest in case of isolated physical objects correctly detected: for each property, a metric computes the distance of that property between physical objects and reference data (e.g., 3D center of gravity, 3D width, 3D height, 2D center of gravity).

3. **Tracking of physical objects of interest:**
    - Criterion qualifying the frame-to-frame tracker. This criterion estimates whether the link between two physical objects detected at two consecutive time instants is correctly computed or not. This criterion depends on three types of information: the overlap between bounding boxes of detected physical objects and reference data, the type of detected physical objects, the presence of one or several links between bounding boxes of physical objects at time t and t+1 corresponding to an identical real object: the metric computes the number of links between physical objects compared to reference data links. A true positive is a reference data link matching a link between two physical objects. A false positive is a link between two physical objects not matching any reference data. A false negative is a reference data link not found.

    - Criterion qualifying the long-term tracker. This criterion estimates whether trajectories of physical objects are correctly detected over the duration of their presence in the scene or not: the metric computes the number of detected trajectories compared to reference data trajectories. A true positive is a reference data trajectory matching a physical object trajectory. A false positive is a physical object trajectory not matching any reference data trajectories. A false negative is a reference data trajectory not found.

    - Criterion qualifying the long-term tracker properties: for each property, a metric computes the distance of that property between physical objects and reference data (e.g., 3D speed, 3D direction).

4. **Recognition of events:**
    - Criterion qualifying the recognition of events globally in a video clip or in a scene in a multi-camera configuration: the metric computes the number of correctly recognized events compared to reference data, for each event type.

    - Criterion qualifying event properties: the metric computes the average difference of the initial (and also ending) event time between detected event and reference data event.

In conclusion, we can see that we can obtain a parameterization of the evaluation by deciding to study a particular criterion and for this criterion, a particular metric. This parameterization provides a first level of flexibility for performing experiments. A second level of flexibility can be provided by refining these metrics with contextual information (e.g., study the accuracy of the 3D width estimation only in a particular zone delimited in the scene).

8.1.3 Unsupervised Evaluation

Unsupervised evaluation is the most convenient way to perform an evaluation as it does not require any user intervention. Nonetheless, this evaluation is less accurate and far more complicated to set up than the supervised evaluation. Indeed, the problem is to determine invariants results have to verify. In this section, we give few examples of unsupervised evaluation criteria which have been defined after having performed several tests and experiments.

Besides the difficulties to automate the evaluation process, it is interesting to consider obtaining an unsupervised evaluation since it provides autonomous adaptability abilities to a system, when combined with a repair mechanism. In video understanding, a feedback from a high-level module to a low-level module is a main way to achieve an unsupervised evaluation. For instance, an event detector currently recognizing an event involving three persons can put this information into a fact base. At the next processing time step, the reasoning engine is able to trigger an adjustment of some parameter values of the object detector by measuring the difference between the current number of outputs of the object detection process and the number of outputs expected by the high-level process.

Once video processing experts are able to run extensive series of testing on their algorithms, they were able to formalize their expertise in the form of unsupervised evaluation criteria. Unsupervised evaluation is more complicated to set up than a supervised evaluation but in the meantime, is more interesting as systems become endowed with an autonomous **adaptability** property. For instance, we have defined two criteria to assess a bad segmentation. In both situations, the associated repair mechanism is to change the reference image. The assumption underlying these two criteria is that both situations cannot correspond to a real scene recorded by cameras having a sufficient frame rate.

Unsupervised Evaluation coupled with a repair mechanism

We illustrate now the global repair mechanism in both cases. The first one is a bad result assessment of the *Segmentation* program. For this program evaluation, experts have defined two unsupervised criteria to assess the segmentation results:

- A difference between the number of blobs detected at two consecutive time instants which is greater than 10.
- A variation of more than 20% of all blob area with respect to the image area.

The second automatic diagnosis is a bad result assessment of the *ImageAcquisition* program. In this case, the illumination conditions have drastically changed globally on the image from the previous to the current time step (e.g., we switch from a dark to a light image or the opposite). This change implies to compute, for the control mechanism, a new reference image which is more adapted to the new illumination conditions.

In order to demonstrate the effectiveness and the general scope of this automatic control mechanism (i.e., the knowledge of control is validated and is not too specific), we illustrate this dynamic system adaptation for two different sites. The first site is a camera

which is installed inside a safe in a bank agency. In consequence, this room is usually dark unless a person comes into it and turns on the light. The second site is the office environment equipped with the vandalism detection system. A sequence has been recorded where a system developer tests a sequence of light switch on and off. In fact, a usual background adaptation process is not sufficient to handle a fast transition. In addition, during a light switching, the sensor undergoes a transient period during which images are of very poor quality. In this case, an intelligent control enables to skip these frames in order to recover a normal processing after this transient period.

For the first site, Figure illustrates the temporal sequence of images with the repair mechanism. Figure illustrates the same temporal sequence of images without the repair mechanism. The condition of the experimentation are identical, this means that the same sequence of programs is applied with the same parameters in both cases, except that the control criteria have been disabled in the second case. For the sake of clarity, all the images are not represented for each time step. We illustrate the key idea which is the assessment-repair action at two time steps (282-283 and 283-284). We can see that after a transient period of 15 frames, the system is able to detect the person at time step 296 and that the system can run with a correct background corresponding to the new environmental conditions. On the opposite, we clearly see that the old version of the system keeps subtracting a wrong background to each input frame, during and after the transient period. As a result, the foreground covers the entire image and the person is thus not detected at time step 296.

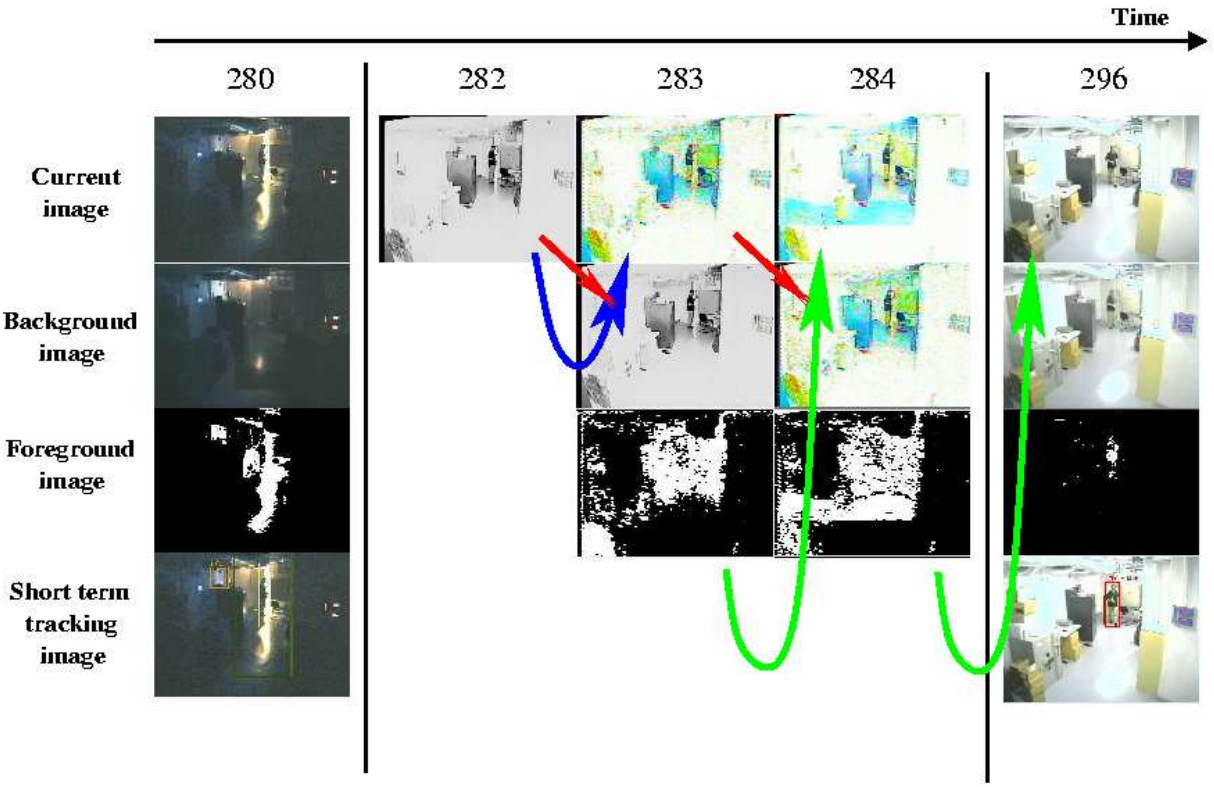


Figure 8.4: *Bank monitoring dynamic controlled sequence. From top to bottom, a column represents the current, background, foreground and the short term tracking output image. A red arrow indicates that the previous image is taken as the new background image. The repair mechanism is either triggered by a bad assessment of the ImageAcquisition program (blue arrow) or by a bad assessment of the Segmentation program (green arrow). In both cases, the subsequent processing at this time step is interrupted, which corresponds to the fact that no image is illustrated for the corresponding processing. We can see that the person is successfully detected at time step 296.*

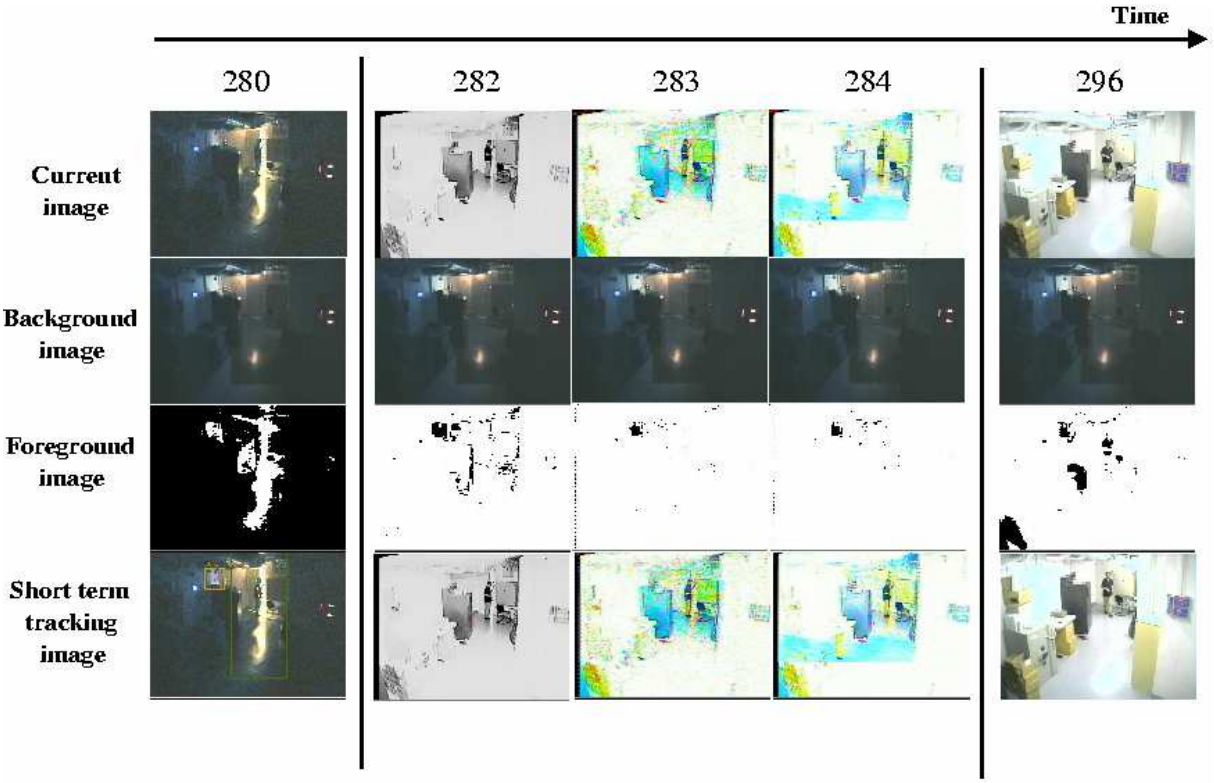


Figure 8.5: *Illustration of the same video sequence of the bank agency without repair mechanism. From top to bottom, a column represents the current image, the background image, the foreground image and the short term tracking output image. A wrong background generates a bad detection over the entire image, thus preventing to detect the person after the transient period.*

For the second site, Figure represents a similar control sequence for the office environment. In this case, the transient period is shorter, due to the fact that the sensor is of better quality. Nevertheless, the control criteria remain mandatory to be able to detect the person under the new environmental conditions. For the sake of clarity, we do not reproduce here the results for the old system without the repair mechanism. The same conclusion applies for this situation: the old system fails to detect the person while the system with the repair mechanism provides satisfactory results.

These results demonstrate the effectiveness of the evaluation-repair mechanism to provide the **adaptability** property. It enables the system to react to unforeseen environment changes, such as a sudden illumination change due to a light switch off.

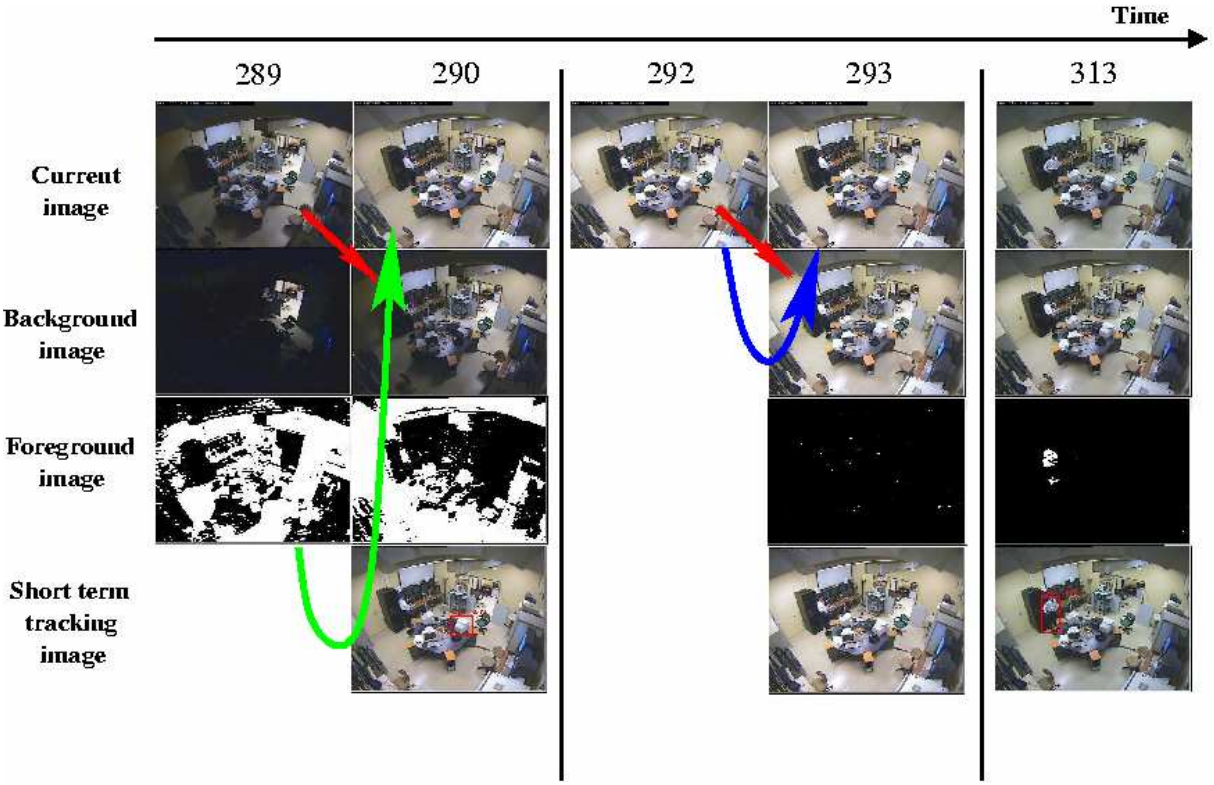


Figure 8.6: *Vandalism detection dynamic controlled sequence. From top to bottom, a column represents the current, background, foreground and the short term tracking output image. A red arrow indicates that the previous image is taken as the new background image. The repair mechanism is either triggered by a bad assessment of the ImageAcquisition program (blue arrow) or by a bad assessment of the Segmentation program (green arrow). In both cases, the subsequent processing at this time step is interrupted, which corresponds to the fact that no image is illustrated for the corresponding processing. We can see that the person is successfully detected at time step 313.*

8.1.4 Discussion on Evaluation

Evaluation is a key element to insure that systems will not only perform well on few video sequences but that they will really address end-user goals. In this section, we have presented a methodology for evaluating system performances. We have seen that several issues need to be addressed in order to obtain a pertinent and useful evaluation: the characterization and selection of test video sequences, the generation of ground truth and the choice of indicators. Once these decisions are taken, the evaluation enables experts to acquire a deep expertise of programs and techniques. This expertise acquisition is usually performed offline and can be then explicitly written using video processing program formalism [Bremond, 13]. This knowledge can then be used through system control during live executions (e.g., by taking into account preconditions, by using unsupervised adaptation rules).

The proposed evaluation tool offers to video processing experts the possibility to perform a large series of experiments for the main video processing tasks and the ability to acquire expertise on programs. However, the design of an effective tool is a very long process and the current version of the tool does not allow all decompositions. This is due to the fact that the tool not only consists in the implementation of criteria and metrics but also in having a large video database for testing (and above all, the corresponding ground truth). This large annotated database is mandatory to obtain significant evaluation results and to understand thoroughly a video understanding issue. This is actually a main objective of the ongoing ETISEO project. Otherwise, by testing on few sequences, there is a chance to obtain a too factual knowledge of the problem. In conclusion, this preliminary tool need to be and will be continued during the ETISEO project [Etiseo, 2005].

8.2 Learning Techniques for Video Understanding

Because of the complexity to tune parameters or to acquire knowledge, many techniques have been envisaged for object recognition [Noor et al., 2006], [Munder and Gavrila, 2006], [Hung et al., 2002], [Lpatev, 2006], [Dalal et al., 2006], [Larlus and Jurie, 2006] and for event recognition [Zaidenberg et al., 2006], [Gong and Xiang, 2005], [Marcon et al., 2005], [Laptev et al., 2007], [Hogg et al., 2005], [Scalzo and Piater, 2006], [Panini and Cucchiara, 2003], [Foresti et al., 2005] mostly based on statistics computation. Three main categories of learning techniques have been investigated for video understanding. The first class of techniques learns the parameters of a video understanding program [Hall et al., 2006], [Sage and Buxton, 2004] and [Micheloni and Foresti, 2003]. These techniques have been widely used in the case for the event recognition methods based on neural networks [Buxton, 2002], [Foresti, 2006], [Huang et al., 2005], naïve Bayesian classifiers [Sheikh and Shah, 2005], [Lv et al., 2006] and HMMs [Galata et al., 2001], [Wilson and Bobick, 2001], [Andrade et al., 2006], [Bremond, 24].

The second class consists in using unsupervised learning techniques to avoid the tedious task of labelling training video sequences [Johnson and Hogg, 1996], and to deduce abnormalities from the learnt events [Foresti et al., 2002], [Gong and Hung, 2005], [Xiang and Gong, 2005] and [Zhang et al., 2005]. These techniques are usually based on clustering techniques and are especially used to learn the structure of HMMs [Brand, 1999].

The third class of methods consists in learning the event model by computing frequent temporal patterns as reported in [Dousson and Duong, 1999] for log reports in computer networks. This technique has recently been applied to video understanding [Bremond, 47]. In this case, an event is defined as a temporal sequence of elements. The frequent event corresponds to the interesting ones. These techniques start to be a necessary step when conceiving video understanding systems but they are still at an early stage of development.

In this section we address the first issue by designing a tool to learn the parameters of a given program. The third class of methods is discussed in the next chapter on Knowledge Acqisition.

8.2.1 Learning Tool for Parameter Adaptation

Based on our experience, the definition of initialization and adjustment criteria by video processing experts remains usually a main problem. In order to cope with that missing knowledge which is difficult to acquire, we propose to apply a learning tool to that particular situation. We first describe the general learning method and we then explain the derived tool to adapt program parameters to illumination changes.

The method proceeds in three steps. The first step is the identification of a set of parameters which is dependent in some sense to a criterion characterizing the system environment. For instance, it can be a subset of parameters of a motion detector which are sensitive to the image contrast. Thus, this first step consists in finding a piece of knowledge which is ideally given by a video processing expert. Based on his/her experience, an expert is generally able to give such a relationship. These parameters can be pre-processed in order to obtain independent parameters (e.g., with a principal component analysis).

The second step is the study of the variability of the selected criterion characterizing the environment (e.g., the image contrast). This study can be either qualitative or quantitative. In case of a qualitative description of the variability, the expert decomposes the variability into several classes, again based on his/her experience. For instance, if the selected criterion is the number of persons in the scene, the expert may create 5-10 classes labelled from few *people* (e.g., 2 or 3 persons) to *crowded scene* (e.g., more than 20 persons). In case of a quantitative description, a clustering algorithm (e.g., k-mean, PCA, ascending hierarchical classification) creates several classes according to a metric. For instance, the metric can be the Euclidean distance between two histograms which are representing the image contrast.

The third step is the determination of a set of optimal parameter values for each class which has been created. These optimal values are obtained with an offline supervised learning (e.g., neural networks, the simplex algorithm, heuristic methods). These values obtained with a training input data set are then used in a generalization phase on a testing input data set.

There are two ways of using this learning method for building video understanding systems:

- **Initial parameter configuration:** if the criterion is qualitatively described, the system is able to retrieve and use the set of optimal parameter values from the qualitative information it receives from an end-user. For instance, an end-user starts the system and provides in the meantime the description *crowded situation* so that the system uses the *crowded scene* set of values.

- **Dynamical parameter configuration:** if the criterion is quantitatively described, the system is able to dynamically retrieve and use the optimal set of parameter values corresponding to the class which has the smallest distance with the current value of the criterion. For instance, the system computes an intensity histogram on the current image, and retrieve the class label which has the closest histogram from the current one.

In both cases, providing an accurate description is a difficult task. On one hand, a qualitative description may be biased due to a limited expertise or a subjective interpretation of the expert (e.g., a wrong number of classes). On the other hand, a quantitative description implies the selection of a metric. This metric is most of the time not perfect and may create wrong element labels (i.e., an element that should obviously belong to class A but that is put to class B due to the metric).

When possible, quantitative descriptions should be preferred as they offer more possibilities. For instance, we can obtain a self-diagnosis property for the system. The idea is to raise a signal when the system has not been able to find an appropriate class label (i.e., a class which is sufficiently close) several times. This signal is a mean to alert the end-user that the system has encountered a new situation/configuration which requires running a supervised learning on this new configuration. Thus, an evolution towards incremental learning can be investigated. Nevertheless, the qualitative description is most of the time the only one available, due to the chicken and egg problem (e.g., how to automatically quantify the number of persons in a scene which is precisely one of the goal of the video understanding task) and due to the difficulty to quantify an abstract concept (e.g., how to automatically quantify the amount of clutter?). There is room here to discover new techniques for quantifying abstract characteristics of environments of video understanding systems.

8.2.2. Method implementation to handle illumination changes

We have applied the general method described above to adapt the *ColourSegmentation* algorithm parameters to illumination changes arising from an outdoor camera. First, we have chosen the image contrast as a representative criterion influencing the segmentation results. The image contrast is here characterized by the repartition of grey level intensities of an intensity histogram over the whole image. The final objective is to learn the relationship between the image contrast and the segmentation parameters. In other words, for a given scene and a given intensity histogram, we want to have the optimal values for segmentation

parameters.

Figure 8.7: *These figures represent two types of illumination conditions: (a) sun with shadow and (b) dark*

Following the method, after the selection of a criterion, we have to study the variability of this criterion (i.e., the image contrast). To this end, we have recorded several hours of video sequences of the same scene corresponding to different illumination configurations, as shown in Figure . For instance, we have chosen different moments in a day (early in the morning or late in the evening), different days (sunny day or cloudy day) and different months (March to June). An intensity histogram is computed for each image taken every five minutes over the whole range of video sequences. Then, we use a standard ascending hierarchical classification algorithm in order to cluster these histograms. This choice is guided by the consideration that the selection of a similarity measure is more intuitive than giving an a priori number of classes (e.g., as in a k-mean algorithm). In fact, an expert can express his/her knowledge on how similar are two histograms through this similarity measure. We have currently chosen the Euclidean distance as similarity measure between two histograms.

An initial distance matrix between elements is created using the similarity measure between two elements. Elements are progressively gathered into classes according to a distance threshold (i.e., the minimum distance is processed first). Thus, we need also a similarity measure between two classes. We have chosen the distance between two classes as the maximum distance observed between all possible pairs of elements, one from each class. This ensures that we never gather classes containing elements whose distance is greater than the distance threshold (i.e., each class element is at least as close from each other as the distance threshold).

Now that several classes have been identified, we have to learn optimal parameter values for each class. The preliminary step to this optimization is to define ground truth data for comparison with operator output results. As we are interested in segmentation results, we have to define an accurate pixel-based ground truth. This step is the most critical one because a main issue when using a learning technique is to provide good learning examples. This difficulty is illustrated in Figure , where we can see that the current boundary of the physical object of interest implies few wrongly labelled pixels. This step is thus a compromise between the time spent by an expert to accurately define the ground truth and the effectiveness of the learning tool.

Concerning the optimization algorithm, we currently use the Downhill Simplex method [Numerical, 2002]. This method minimizes multi-dimensional functions without making any assumption on function derivatives. This method is thus well indicated for achieving our objective as we do not have any information about the derivatives. The main corollary is that we do not have a guarantee of obtaining a global minimum. However, this method provides a mechanism to overcome this potential problem. In fact, the method is run several times starting with a different initial vector and is stopped when the same result value is found at least three successive times. Currently, we maximize **S** which is defined as the sum of correctly detected pixels (true positives) and correctly non-detected pixels (true negatives) over the total number of pixels (true positives, true negatives, false negatives and false positives).

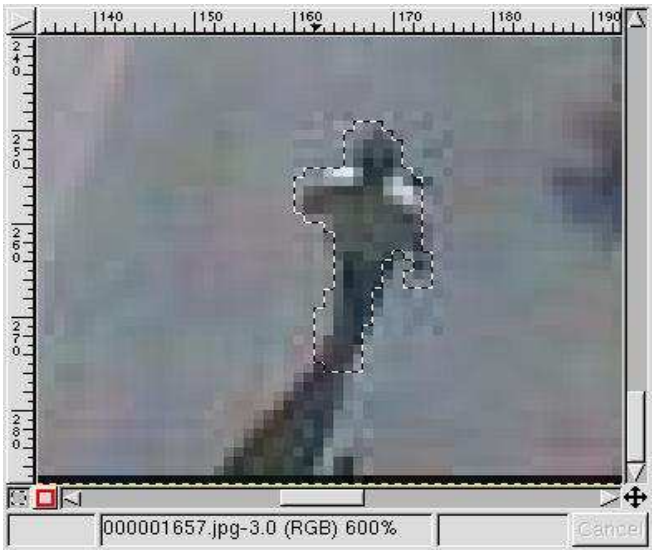


Figure 8.8 *: Definition of the shape of a physical object of interest to obtain a pixel-based ground truth.*

The last step before being able to use learnt values for dynamical parameter adaptation during a live system execution is to compute a class representative (i.e., an intensity histogram). We have currently defined the class representative as the mean histogram of a class. This way, the system first computes the best match between the current intensity histogram and available class representatives and then uses the optimal values corresponding to the selected class. More concretely, each set of values corresponding to a class is loaded in a *LearntParameters* table in the fact base at the starting of the system. The matching operation is performed inside the *Acquisition* module (after the load of a new image) and the result (i.e., a class label) is put in the VSIP shared memory. This class label is retrieved from the shared memory manager by the control module. Finally, this label is used by the initialization criteria to choose the right *LearntParameters* from the fact base table in order to assign values to parameters of the *ColourSegmentation* program.

Experiment for validation of the learning tool

For the validation of the proposed learning tool, we have realized an experiment with an outdoor camera on top of a building at INRIA Sophia-Antipolis. An AXIS2420 IP camera is viewing a car park with a far field of view. The objective of this experiment is to

demonstrate that it is possible to decompose the variability of illumination changes in a given number of clusters and then to learn an optimal set of parameter values for the segmentation program for each of these clusters, using a training set. Then, during a generalization phase (i.e., when processing new unknown input data), the system can automatically retrieve the set of optimal parameter values by selecting the cluster which is the closest to the current illumination conditions.

The assumptions for the validity and scope (i.e., the **reusability**) of this experiment are the following:

- The video sequence is taken during the period of day from 7 am to 9 pm.
- The field of view is restricted to the car park area (i.e., the tree area is not managed).

In order to learn the relationship between illumination changes and segmentation parameters, we first characterize the illumination by the repartition of grey level intensities of an intensity histogram over the whole image. The image histograms which have been computed for each image taken every five minutes over the whole range of recorded sequences are represented in figure

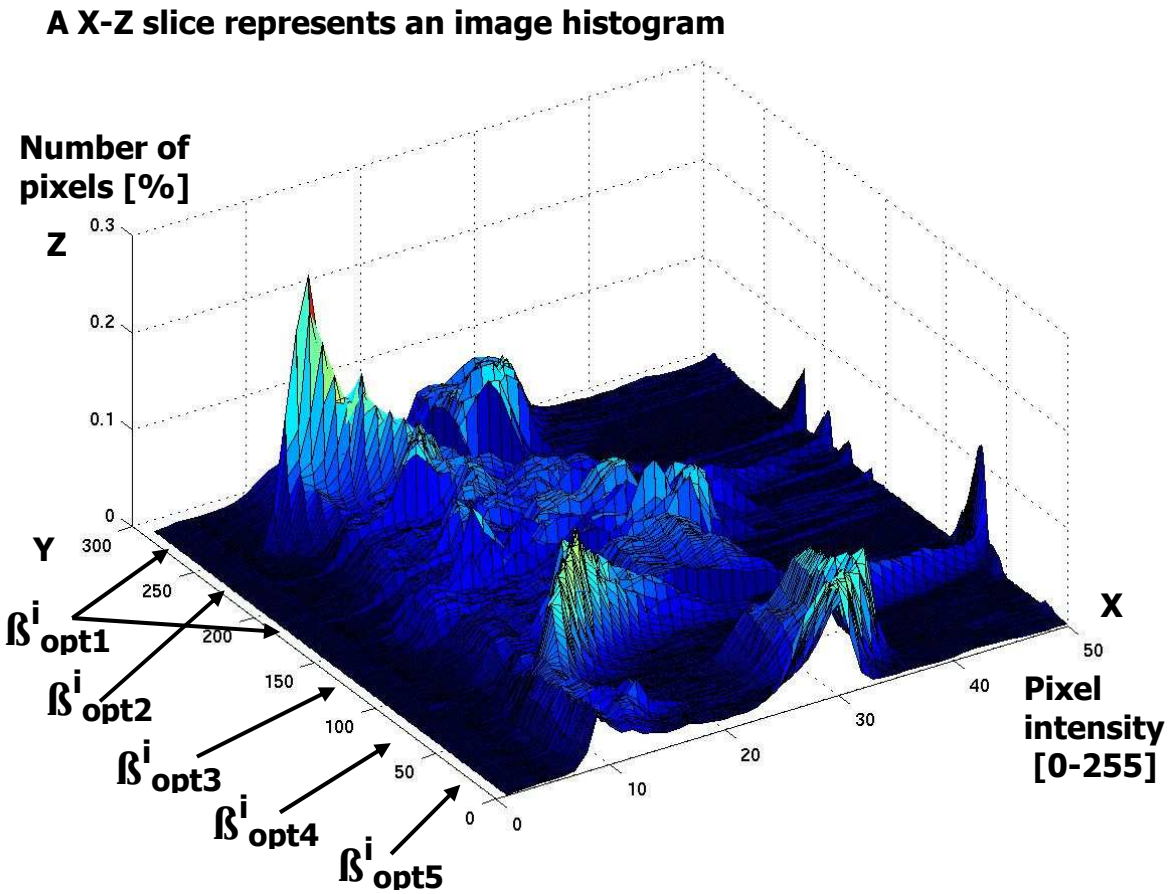


Figure 8.9: *This figure shows the image histogram for each instant of the day on the y axis. The x axis corresponds to the intensity level of the pixels and z axis the frequency of the intensity levels. The histograms have been clustered into five classes. For each class, we have computed the optimal parameter set* β^i.

We have recorded a total of 23 hours of small video sequences of about 30 minutes each, during different moments in a day (early in the morning or late in the evening), different days (sunny day or cloudy day) and different months (March to June) and with different activities (car or pedestrians). These non consecutive intensity histograms are ordered along the Y axis. Each X-Z slice of this 3D diagram corresponds to an intensity histogram of a whole image. Each histogram represents on the Z axis the percentage of the total number of pixels belonging to a given intensity interval on the X axis. The intensity scale (i.e., [0-255]) is divided into 51 intervals of 5 intensity levels.

For the sake of illustration of the ascending hierarchical classification method applied on the whole set of video sequences, we have obtained 5 clusters along the Y axis:

- [0-40] and [261-280]: this cluster corresponds to images taken during sunny days.
- [41-115]: this cluster corresponds to images taken during sunny days with few clouds.
- [116-215]: this cluster corresponds to images taken during cloudy days.
- [216-249]: this cluster corresponds to images taken in the late morning, before the sun appears.
- [250-260]: this cluster corresponds to images taken very early in the morning.

The parameter optimization step is the most critical one of this learning process, due to several reasons. First, it is difficult to obtain good training examples. As the optimization process usually takes more than 1 day to be completed, the total time spent to validate the training stage is huge. A second reason is that the ground truth definition at a pixel level is time consuming (e.g., one frame per 2 minutes). Finally, it is sometimes difficult to record sequences containing some activity outside the office hours. In this case, an alternative solution consists in recording sequences with actors.

In consequence, due to the fact that we had a limited amount of time to realize this experiment, we have selected a subset of the whole range of the video sequences to perform the optimization. We have selected the range [177-215] corresponding to several non-consecutive video clips taken the same day, for a total of more than 3 hours. First, we have selected several video clips for which we have manually defined the ground truth. In order to give an indication, we have defined the ground truth for a thousand of frames and the whole annotation process took about 30 hours. Second, we have classified these video clips into two sets: a training set and a testing set. Third, we have defined 2 different clustering situations: the first situation contains one cluster and the second situation contains 2 clusters. The first situation is directly obtained (i.e., the whole range) while the second has been obtained by running the clustering algorithm with a predefined threshold.

The first cluster roughly corresponds to video sequences recorded in the morning (the sun is rising) whereas the second cluster corresponds to video sequences recorded in the

afternoon (very sunny). The training score **S** (which have been introduced before as the function to be maximised) is equal to 99% in each case.

In conclusion, the optimization algorithm has given us a set of parameter values for both clusters. Learning these values is a first step to provide an optimal configuration for each illumination situation of the INRIA car park surveillance system. However, more experiments are needed to learn values for the other clusters that can be defined based on the whole set of recorded video sequences.

These learning results may be directly applied to other systems which use a static camera of the same type and of the same orientation, which view the same type of scene (i.e., a large portion of the image is asphalt) and which are colour calibrated. Finally, as a histogram can be easily automatically computed for input frames on a regular basis (e.g., once each 5 minutes), the program parameters can be automatically (dynamically) adjusted by taking the set of parameters of the cluster which is the closest of the current illumination conditions.

8.2.3 Discussion on Learning

Despite the encountered difficulties to obtain good learning examples when defining a pixel-based ground truth, and the two days required in average for computing on a standard computer the optimization phase, this tool is easy to use. Currently, the tool is not fully automatic and a manual step is needed to make the transition from the clustering to the optimization part. Indeed, the clustering algorithm is implemented in Matlab and the optimization algorithm is implemented in C++. In the future, we want to enlarge the capabilities of this learning tool by taking into account other criteria characterizing the scene environment. For instance, we would like to learn the relationship between the number of persons in a scene and the tracking parameters. This is based on the observation that tracking has to handle more or less frequent occlusion situations depending on the number of persons in the scene.

Nevertheless, the current utilization of this tool during the construction of a video understanding system has demonstrated the benefit of learning to obtain more **effective** and **adaptable** systems. In addition, these results may be directly **reused** for other systems in the same conditions. However, this learning tool is only a first step towards a complete learning tool which can be used to learn different relationships between the parameters and the environmental conditions and which can be applied at each level of the video understanding chain (i.e., for low-level tasks as well as high-level ones). In consequence, future investigations are needed to extend and validate the current learning tool.

8.3 Conclusion on Evaluation and Learning

In this chapter, we have presented several works on performance evaluation of video understanding algorithms and on learning techniques for parameter tuning.

On evaluation, we have described a methodology and a tool for evaluating the performance of video understanding systems. Given a video data base, ground truth and a set of metrics, the user can adapt the tool to evaluate a specific video understanding task (e.g. object tracking), or specific scene conditions (e.g. sudden illumination change). We have also presented an algorithm to evaluate automatically the quality of a segmentation program for object detection in order to re-compute dynamically the reference image when it is necessary.

On learning, we have described an algorithm to learn automatically the parameters of the segmentation program, by computing a characterisation of the illumination conditions of a given scene. This is the first stage towards the dynamic configuration of video understanding systems.

These evaluation and learning mechanisms are at a preliminary stage, but are necessary to obtain an effective video understanding system, operational 24/7 at a large scale. Therefore, more efforts still need to be done. In particular, an appropriate formalism needs to be defined to make these mechanisms seamlessly integrated in new scene understanding systems.

Chapter 9

Knowledge Acquisition

Even when the correct interpretation of the scene has been performed, the scene understanding system still has to **communicate** its understanding to the users. There are at least three types of users: program developers (e.g. vision and software engineers), experts of the application domain (e.g. security experts, domain experts) and end-users (e.g. human operators, managers, police). The first challenge is to enable program developers to understand all specific components and in the same time the global architecture of the scene understanding system, so that they can adapt efficiently their programs and configure and install the system on a site. To reach this goal, we have proposed a formalism to express program knowledge and to capitalize experience in building scene understanding systems [Bremond, 13 and 48].

Second, if we want an effective system, the **a priori knowledge** on the application domain needs to be formalized to enable the domain experts to describe their knowledge and their methodology for analyzing the scene. In scene understanding, there are four main types of knowledge to be represented: (1) the empty scene of the surrounding (e.g. its geometric), (2) the sensors (e.g. calibration matrices of the cameras), the network and the processing units, (3) the physical objects expected in the scene (e.g. 3D model of human being) and their dynamics, and (4) the events of interest for end-users. We have defined formalisms to model all these types of knowledge. We have proposed formalism to model the empty scene (called also contextual information) corresponding to the static environment of the scene, including

the sensor configuration, the geometry of the scene and associated contextual information specific to the application (subway platform, street, escalator,…). This formalism is based on a 3D spatial representation of the scene [Bremond, 4, 14, and 16]. We also use several models to represent the physical objects of interest. For instance, when the information is not accurate, we use the ratio of the 3D height by width of the physical objects [Bremond, 3]. When more information is available, we represent the physical objects by a 3D parallelepiped corresponding by the volume occupied by the objects [Bremond, 53]. When accurate information is available, we use a 3D geometrical realistic and articulated model of human beings [Bremond, 10]. Finally, we use an intuitive representation of events and a declarative language for helping end-users to describe their scenario of interest [Bremond, 28, 33 and 38]. This event representation contains four main parts: (1) the physical objects involved in the event, (2) the components (i.e. sub-events) composing the event, (3) the forbidden components which should not occur during the event, and (4) the constraints linking the physical objects with the components.

To describe this a priori knowledge, we are using ontologies. For instance, we have built a video event ontology during the framework of the ARDA workshop series on video events, based on our experience on video surveillance [Bremond, 59]. The use of video understanding systems has been generalized all over the world leading to the need of a shared ontology on the application domains. An ontology is the set of all the concepts and relations between concepts shared by the community in a given domain. The ontology is first useful for experts of a given application domain to use video understanding systems in an autonomous way. The ontology makes the video understanding systems user-centered and enables the experts to fully understand the terms used to describe activity models. For instance, the ontology needs to be defined together with end-users. So they will be able to define themselves their scenarios of interest. Moreover, the ontology is useful to evaluate the video understanding systems and to understand exactly what types of events a particular video understanding system can recognize. This ontology is also useful for developers of video understanding applications to share and reuse activity models dedicated to the recognition of specific events. Building an ontology used as a reference for video understanding applications is particularly difficult because many developers and experts of application domains all over the world have their own ideas about how to describe human activities. The terms chosen to name the ontology concepts are taken from every day life but they have been redefined to avoid ambiguities. There are two types of ontology: the first one (user ontology) enables to address real scenarios of interest for end-users. These scenarios are usually complex and depend on the application domain. The second one (visual concept ontology) describes primitive events that can be detected by video technologies and enables to link the scenarios of interest to perceptual features effectively detectable by video technologies. We have recently extended this visual context ontology with audio event concepts [Bremond, 59].

We believe that a tool incorporating a dedicated ontology should to be provided to assist the experts in defining the scenarios that the system has to recognize. Currently, we use the ‘protégé’ software developed at Stanford University (UK) to build and browse the hierarchy of concepts used to describe the video event ontology. We are developing a second software to edit and define accurately the end-user scenarios, with the scenario components and constraints. Moreover, to help this process, we have designed a graphical tool to generate and **visualize** 3D virtual animations illustrating these scenarios [Bremond, 27 and 40].

In complement of these tools, we have developed clustering techniques to mine the frequent activities (i.e. event patterns or time series) occurring in the scene [Bremond, 47]. A

recurrent question consists in determining whether this knowledge is given a priori or is learned.

Third, if we want the system to be used, special care needs to be brought to display what has been understood to the final users. An ergonomic interface on an adapted media (e.g. immersive reality or PDA personal digital assistant), a convenient representation of the scene (e.g. virtual 3D scene, augmented reality) using an intuitive vocabulary are the necessary devices to be provided to the end-users. Besides these devices, the system needs to take into account feedback information from the end-users to be able to adapt its performance to the specific goals of each user. These topics are usually considerate secondary compared to the effectiveness of video understanding algorithms. However, we believe that these topics are important and will become critical in the future.

The whole challenge of knowledge acquisition consists in organizing all this knowledge in order to capitalize experience, share it with others and updating it along experimentations. In this chapter, we present two types of work. In the first section, we describe a tool for simulating and visualizing human behaviors. This tool, given a scenario description, generates automatically a video illustrating the scenario in an environment characterized by specific conditions. It enables experts to understand and validate the functioning of a video understanding system and the hypotheses to obtain satisfactory results. This tool has been developed during the internship of Thinh Van Vu [Bremond, 27].

In the second section, we describe preliminary work on learning event patterns (i.e. time series) corresponding to frequent activities occurring in the observed scene. The event patterns are learnt by computing the frequency of all combination of primitive events detected by a video understanding system. The learnt event patterns can complement scenarios defined by end-users, especially in applications monitoring everyday activities. This work has been done during the internship of Alexander Toshev [Bremond, 47].

9.1 Human behavior simulation for video understanding

This section presents a modelling framework for the visualization and simulation of automatic behavior analysis. The video understanding consists in recognizing pre-defined scenarios describing human behaviors from video sequences. This framework (called test framework) has (1) to visualize the computation of the interpretation and scenarios described by an expert, (2) to be flexible enough (configurable) for testing the different configurations of the understanding system and (3) to be realist enough to understand what is going on in a real scene. Another requirement of this framework is to verify the coherence between a given understanding system and the test framework, so we can establish the limit and the robustness of the understanding system. This test framework will be an efficient tool for the developers (e.g. experts in vision and in scenario recognition) and for the experts of the application domain (e.g. agents of security). To validate this framework, we have developed a test system for video understanding system. Here, we are using VSIP platform, described in chapter 5 as an example of understanding system.

For more than 20 years, the problem of 3D scene visualization has been addressed by many laboratories who study the visualization of a 3D scene from its description. For example, at the faculty of Computer Science of Toronto University [Terzoploulos, 1999], researchers generate 3D animations where many fishes and a swimmer evolve in the bottom of the sea. To visualize these animations, they have modelled the behaviors of individuals and fishes and theirs interactions in groups. In particular, they have modelled all the physical and biological rules for fish to swim, eat, reproduce and perceive other fishes. At the Computer Graphics Lab of the Swiss Technology Institute of Lausanne [Bezault et al, 1992], researchers have modelled individuals evolving in a museum, in a street and in a supermarket. They have also modelled the crowd behaviors like the reaction of people in fire situations. These laboratories have obtained many results in the domain of 3D animations from a scene description. However, there are few laboratories who study the visualization of scenarios recognized by an automatic video understanding system. For example, the Robotics Institute at Carnegie Mellon University [Collins and Kanade, 2001], computes 3D animations where a group of individuals enters/leaves the university site by taking as input the camera network surrounding the university. The goal of these animations is mainly to demonstrate the tracking of the group all around the university.

Our approach to describe human behaviors consists in defining six generic models (i.e. meta-class): scene context, camera, human body, action, scenario and scene-scenario. Using these generic models, we can construct specific models (e.g. the scenario class “meeting at a coffee machine”) described in libraries of models. Then these specific models are used to generate instances (e.g. scenario “individuals A and B meet at the coffee machine M”) to visualize what is occurring in a given real scene (corresponding to videos or scene descriptions). We also propose a description language to build all these models.

9.1.1 Visualisation and simulation for video understanding

As described in chapter 5, a video understanding system contains three principal modules: (1) individual detection, (2) individual tracking and (3) scenario (i.e. behavior) recognition. It takes its inputs from a video camera and generates recognized scenarios as output. A typical system is represented in Figure 9.1.

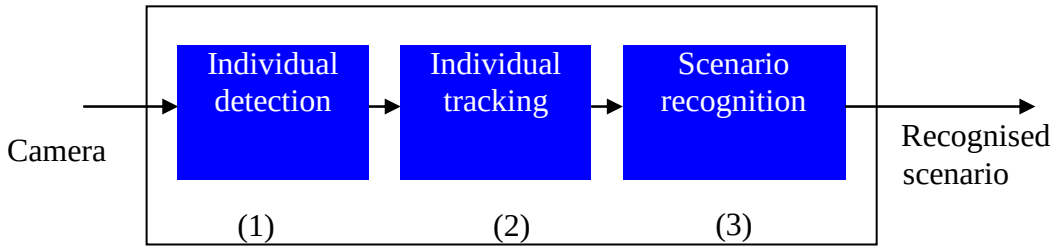


Figure 9.1: *a video interpretation system contains three principal modules.*

The test framework for a video understanding system should contain the following functionalities:

- *Visualization of scenarios recognized by an understanding system and scenarios described by an expert*. It is important for the developer (e.g. expert in vision and scenario recognition) to visualize each step of the scenario recognition process. It is also important for the experts of the application domain (e.g. agent of security in a metro) to visualize the scenarios that they describe.
- *Evaluation of the couple understanding-test system:* verify the coherence between the understanding and test system.
- *Validation of understanding system:* establish the limits and robustness of the understanding system by simulating test videos.

For this purpose, we propose to define a test framework that allows the three following tasks (see Figure 9.2):

1. *Generation of realistic 3D animations corresponding to the scenarios recognized by an understanding system*. The generation of animations needs to be flexible enough to illustrate specific steps of the understanding process. For example, to illustrate the tracking process, it is convenient to give a specific colour to each tracked individual.
2. *Comparison of two animations, one coming from the interpretation of an initial video and the other one coming from the interpretation of a new video generated by the test system and corresponding to the interpretation of the initial video*. For a scenario recognized by an understanding system from an initial video, the test system generates a first 3D animation and a second video that corresponds to the recognized scenario. We have to compare that the first animation is similar to the second one.
3. *Automatic generation of a set of videos* corresponding to a scenario described by an expert. This set of videos should illustrate the variety of all possible instances of this scenario (e.g. variety due to different locations of individuals and due to different optical effects like illumination).

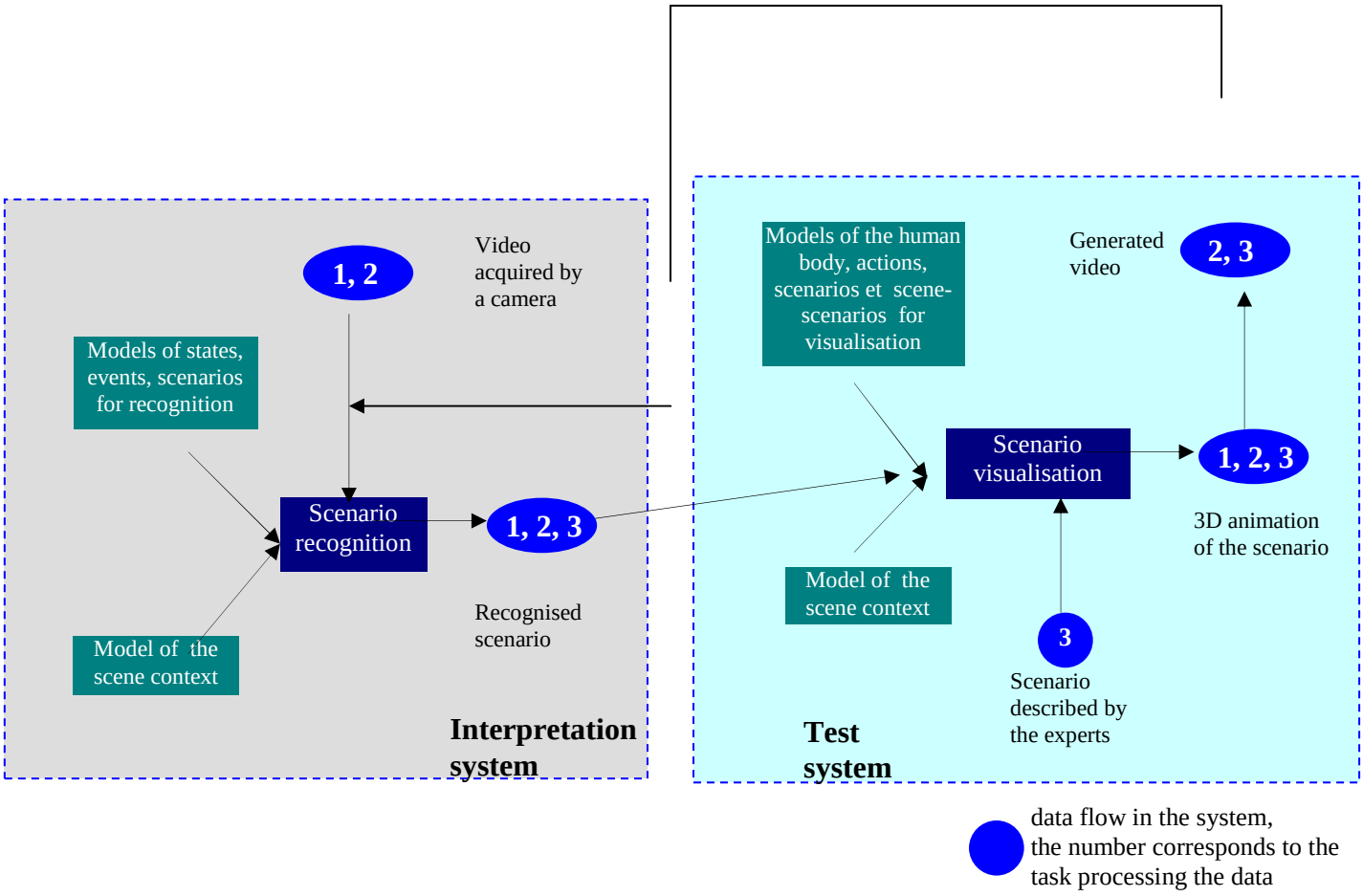


Figure 9.2: *an interpretation system and its test system.*

9.1.2 Scene context

Scene context and camera representation

The scene context contains a set of contextual information related to the environment of the scene (e.g. the 3D geometry of the scene) and used by the video interpretation and visualisation process. We use scene contexts containing the four following elements:

- a set of polygonal zones with semantic information: entrance zone, zone near the seat,…
- a set of areas of interest that gather the connected zones with the same semantic information: platform, metro tracks,…
- a set of 3D objects of the environment which principally includes the equipment (e.g. a seat, a door).

- a calibration matrix of the scene containing the extrinsic parameters of the camera (position, direction and field of view - FOV).

We represent the 3D context objects of the environment using a meta-class (generic model of 3D context object) and we represent each object type as a class of this meta-class. Thus, we have built five classes of context objects that usually appear in metro scenes: class “seat”, “trashcan”, “validation machine”, “ticket machine” and “door”. To facilitate the building process, we have defined a description language where the meta-class 3D context object enables the construction of a hierarchy of 3D context object classes. These classes contain five attributes:

- the relative co-ordinates that represent the position of the 3D object in the referential of the super part (super 3D object). For example, the leg of a chair is defined relatively to the chair.
- the angular co-ordinates of the 3D object in the object referential,
- the size of the 3D object along its referential axis,
- the sub-parts or/and geometric primitives that constitute the 3D object,
- the colour of the 3D object.

We use three types of geometric primitives: sphere, truncated cone and parallelepiped. For the truncated cone, its both sections can have different radius. The geometric primitives have the same attributes of the 3D object classes except the list of sub-parts.

Visualisation of the scene context

To visualise the scene context, we use GEOMVIEW (a free software for 3D visualisation) for visualising the polygonal zones and 3D objects. To use GEOMVIEW, it is first necessary to represent the objects of the scene context in OpenGL format (see Figure).

***Figure 9.3**: visualisation of a scene context description using GEOMVIEW.*

For the polygonal zones, there are specific methods in GEOMVIEW to display them in 3D. For the 3D context objects we first compute the co-ordinates of the geometric primitives of the objects in the scene referential: we multiply the co-ordinates of the geometric primitives by the referential transformations of all containing super 3D objects. Then we use a GEOMVIEW method that constructs the vertices and the facets of the geometric primitives.

Using this representation we have built two scene contexts for metro station including a platform, a corridor and a hall: one for Yser station in Brussels and one for Segrada-Familia station in Barcelona. Figure 9.4 shows two images from Sagrada-Familia station.

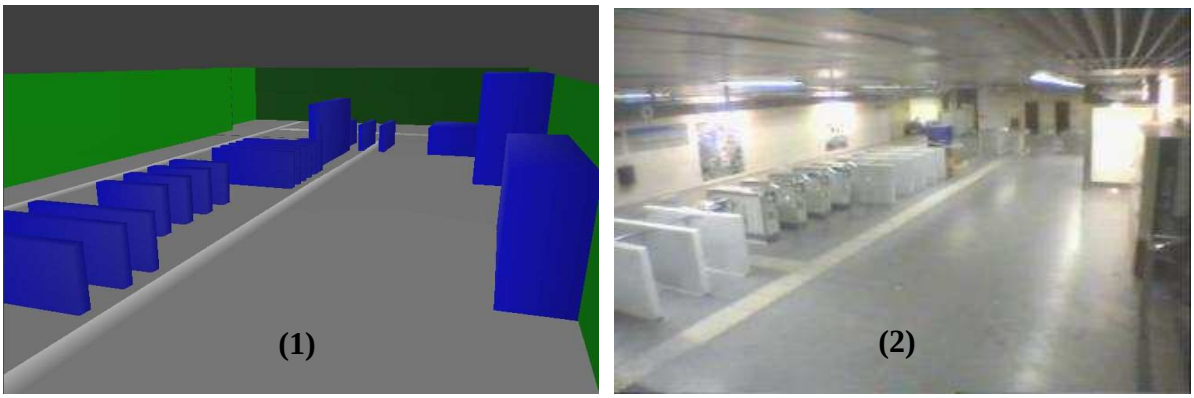


Figure 9.4: *Sagrada-Familia station in Barcelona: (1) raw image taken by a camera and (2) scene context model corresponding to this image.*

9.1.3 Human body

We use a hierarchical and articulated model for the generic model of human body parts. A human body part is composed by sub-parts or geometric primitives. These primitives are the same one used for 3D context objects: spheres, truncated cones and parallelepipeds. Figure 9.5 shows the 26 geometric primitives composing the human body. We represent classes of human body parts (and the whole human body) using a generic model similar to the generic model of 3D context object.

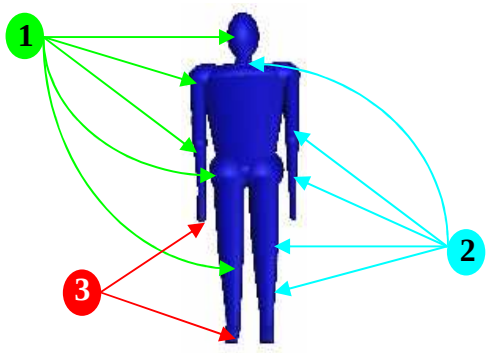


Figure 9.5: *hierarchical and articulated model of the human body using three types of primitives (1) spheres, (2) truncated cones and (3) parallelepipeds.*

In the description language we have defined 14 classes for modelling human body parts: the whole human body, the head, the arms, the legs, the neck, the shoulders, the hips, the trunk, the foot and the hand. Figure 9.6 shows the human body from different view points.

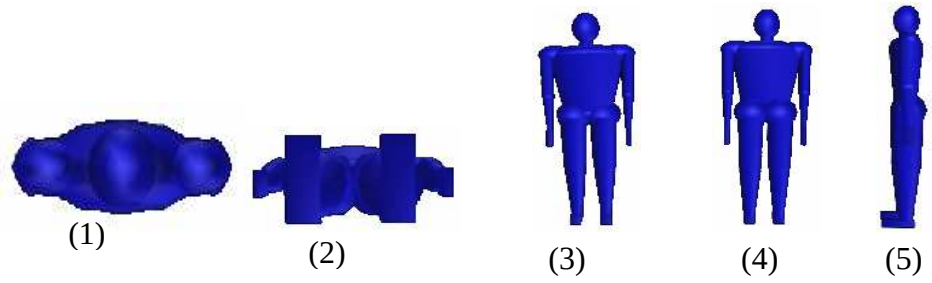


Figure 9.6*.: visualisation of the 3D model of the human body: (1) top view, (2) bottom view, (3) front view, (4) back view and (5) view from the left.*

Using these models, there are two ways of visualising the human body. First, we can visualise an individual from its description by an expert. Second, we can visualise an individual detected by an understanding system from a video sequence. In both cases, we visualise the body part by displaying the geometric primitives composing the body part through GEOMVIEW.

9.1.4 Human behaviour

Human behaviours for understanding systems

In understanding systems, the notions of state, event and scenario [Bremond, 3] are used to recognise human behaviours. A state characterises at a given instant, the situation of an individual detected by a camera. An event defines a change of state at two successive instants. A scenario defines a combination of events.

In the understanding system that we are testing, eight states are defined: posture (e.g. lying, crouching, standing), direction (e.g. towards the right, towards the left, leaving, arriving), velocity (e.g. stopped, walking, running), location with respect to a zone (e.g. inside, outside), proximity with respect to a context object (e.g. close, far), relative location with respect to another individual (close, far), relative posture with respect to a context object (e.g. sitting, any) and relative walk with respect to another individual (e.g. coupled, any). By using these eight states, eighteen events are defined: for example, the event “falling” is defined from the change of posture from “standing” to “lying”. Combining these events, several scenarios are defined such as “two persons meet at a coffee machine” for office applications and “graffiti on wall” for metro station applications. A scenario is a set of spatio-temporal constraints on the individuals of the scene, on the context objects and/or on the previously recognised sub-scenarios (or events). The temporal constraints are expressed by equations that combine the instants when the events are detected.

Human behaviours for the test framework

In the test framework, the notions of posture, action, scenario and scene-scenario are defined to visualise behaviours recognised by an understanding system or described by an expert. A posture corresponds to all body parameters of an individual to be visualised at one instant. An action characterises an individual motion when one (or several) of its body parameters change(s). Behaviours are represented by scenarios. A scenario combines the individuals of the scene and the context objects with sub scenarios which are relevant to the same activity. An elementary scenario is an action. A scene-scenario combines and instantiates all previously defined scenarios.

In our formalism, an action (or scenario) can be visualised at different speeds which indicates how many frames per second are displayed. An action (or scenario) can have a departure/arrival position which locates the individual at the beginning and the end of action (or scenario). The temporal constraints are expressed by intervals (named periods) that correspond to the duration of an action (or scenario). The interval of a sub action (or sub scenario) is defined relatively to the period of the containing action (or scenario).

Because our purpose is to conceive a test framework for automatic video understanding systems, we do not consider more precise actions such as “balance the arm” and “move the finger” which are difficult to detect by understanding systems.

Action

Generic model of actions

An action is relative to the motion of one body part (or the whole human body) which is characterised by the changes of the body part parameters. These changes are mainly rotations around the body part axis. An action is described by a *hierarchical model*: an action can be decomposed into *sub action(s)* describing the motion of sub part(s) (see Figure 9.7). In our formalism, to ease the description of actions by experts, it is possible to indicate the departure/arrival position in the case where the body part is the whole individual. There are two types of actions: periodic (e.g. “walking”) and non-periodic (e.g. “move close to”). For non-periodic actions, the period corresponds to the duration of the action. For periodic actions, the number of periods is defined in the containing action and the duration is obtained by multiplying the number of periods times the action period.

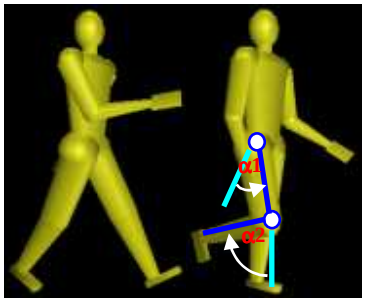


Figure 9.7: *in the action “walking” during interval [t1,t2], the right leg rotates with angle α1 around the hip; and in its sub action “the right leg up”, the lower part of the leg rotates with angle α2 around the knee.*

To represent actions, we propose a generic model with the following attributes:

- the concerned part of human body.
- the fixed part of human body on the ground (see above the visualisation of an action).
- the global period of the action.
- the variation of angles of rotation around the part referential.
- the speed of the action.
- the departure/arrival position (optional, used only when the part is the whole individual).
- the list of sub actions with:
- their relative period,
- the concerned sub part of human body,
- the variation of angles of rotation around the sub part referential.

Visualisation of actions

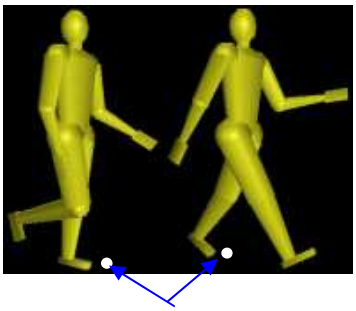
An action is visualised by displaying the individual performing the action at regular instants. In the case where the test framework visualises the actions recognised by an understanding system, the individual posture to be visualised is obtained by the posture detected by the understanding system. Therefore the test framework just needs to display the individual where it has been detected. If the visualisation frequency is greater than the frequency of the input video, then it is necessary to interpolate linearly the intermediary positions of the individual. Knowing the global position of the individual, we calculate the vertices of the geometric primitives of the individual body in the scene referential and display the primitives through GEOMVIEW in the same way with the visualisation of 3D context objects.

In the case where the test framework takes as input the actions modelled by an expert, we visualise an action in three steps:

1. *calculation* of the current posture from the previous instant. By using the posture of the previous instant and the angular variations of the action, we calculate the new angular co-ordinates of each sub part of the body at the current instant. From the new angular co-ordinates, we can calculate the new vertices of the primitives of each body part by multiplying their co-ordinates by the referential transformation matrix. This transformation matrix is defined for each body part and enables to compute co-ordinates in body party referential to co-ordinates in its containing body part referential. By this way we obtain the vertices of the body part defined relatively to the global position of the individual. These new co-ordinates define the new posture of the individual in the individual referential.
2. *calculation* of the global position of the individual. To calculate all positions of the individual, we make the following hypothesis: at each moment, there is a fixed point of a body part on the ground (see Figure 9.8.). Currently, the actions we are interested in are actions where the individual has a fixed part on the ground (e.g. “walking”, “running”). In the near future, we are planning to extend our formalism to handle actions such as “jumping above a barrier”. To calculate the global position, we first compute the distance between two successive fixed

points on the ground (if the fixed point of the action has changed since last instant). Second, we compute the motion of the referential point of the individual relatively to the current fixed point. These two points (referential/fixed points) are defined by the expert. By applying the transformation corresponding to this motion to the vertices of primitives defining the individual, we obtain the new co-ordinates of these vertices that correspond to the current posture of the individual. There are other approaches to calculate the position of an individual from its motion description. In [L. Bezault et al, 1992], the authors describe the motion by mathematical equations (based on experimental data) and calculate the position of individuals by solving the equation system.

3. *visualisation:* after computing the geometric primitives of the human body relatively to the new global position of the individual, we display all the primitives with GEOMVIEW.



fixed point during the interval [100, 150]

Figure 9.8.: *one of the fixed points while the individual is walking.*

Scenario

Generic model of scenarios

A scenario combines the individuals of the scene and the context objects which are relevant to the same activity with more elementary sub scenarios. An elementary scenario is an action that corresponds to the motion of the whole human body of the involved individuals. The model of scenarios is defined as the model of actions. It is a hierarchy of sub scenarios. Each sub scenario is ordered in time thanks to intervals (called periods) that correspond to the duration of the sub scenarios defined relatively to the global period of the main scenario. Unlike actions, a scenario has an attribute corresponding to the list of actors and context objects involved in the scenario. At the level of scenarios, an actor (or a context object) is represented by a variable that corresponds to the role of the actor in the scenario.

Visualisation of scenarios

We visualise a scenario in three steps. First, we link all actors and context objects of the scene involved in the scenario to the variables defined in the actions composing the scenario. Second, we order these actions in time: for each action, we calculate its duration

(start and end point) relatively to the scenario period, defining when the action is active (is displayed). Third, at each instant, we display all actors involved in active actions using GEOMVIEW. Figure 9.9 presents the visualisation of the scenario “two persons meet at a coffee machine” between the instants 80 and 240.

Scene-scenario

Generic model of scene-scenarios

A scene-scenario combines and instantiates all previously defined scenarios. To represent a scene-scenario we use a generic model that has five attributes:

- The scene context includes the list of context objects involved in the scene. The expert describing the scene can change the default attributes of the context objects (e.g. their colour).
- The virtual camera information that corresponds to the viewpoint from where the 3D animation is visualised. This information includes the 3D position, the direction and the field of view (FOV) of the camera.
- The list of actors involved in the scene with their initial position, size, posture and colour. If this information is not provided, default values are used.
- A set of scenarios occurring in the scene. For each scenario, we first specify which actor corresponds to which role defined in the scenario and we also specify the scenario period relatively to the global period of the scene-scenario.
- The visualisation speed of the scene.

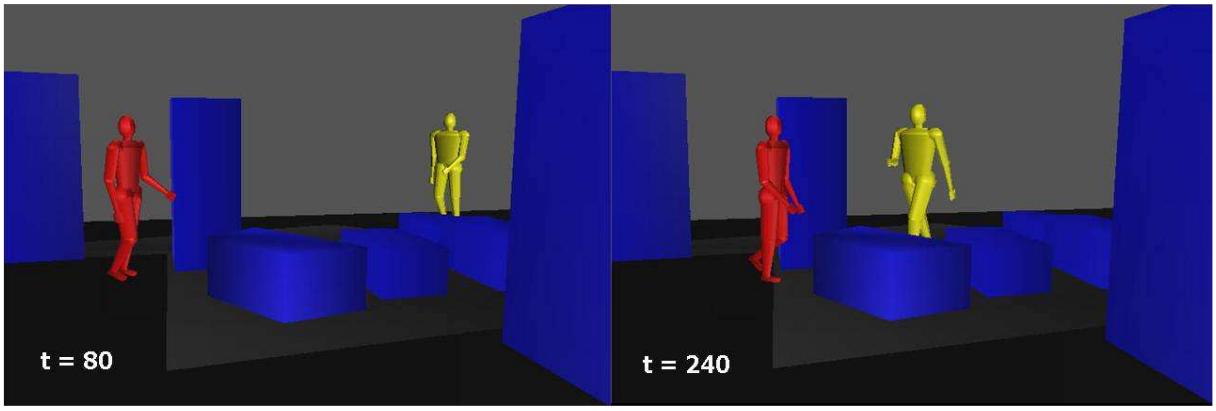


Figure 9.9: *visualisation of the scenario “two persons meet at a coffee machine” between the instants 80 and 240.*

Visualisation of scene-scenarios

We display a scene-scenario in three steps. First, we initialise and connect the actors and the context objects to the scenarios defined in the scene. Second, we calculate the parameters of the virtual camera of GEOMVIEW. Third, we display all active scenarios

composing the scene at each instant. The visualisation frame rate can be specified either at the level of the scene-scenario or at the level of the scenarios or actions.

9.1.5 Results

To validate this framework, we have developed a test system for VSIP, the video understanding system taken as an example of understanding system. Thanks to this test system, we have realised three 3D animations that visualise the results of VSIP from real videos of metro taken for the ADVISOR European project. Figure shows (1) an image that illustrates the individuals detected by VSIP and corresponds to the output of VSIP and (2) an image that illustrates the 3D animation generated by the test system (named “animation 1”). At the time, VSIP was not able to detect the posture and the orientation of the individuals (front view, lateral view). By default, the 3D animation shows the front view of the individuals.

Thanks to the test system, we have also realised seven 3D animations from scene-scenarios described by an expert and then generated the corresponding videos taken from the view point of the real camera.

Moreover we were able to verify the coherence between the understanding and the test system. For that, we have first generated a 3D animation (named “animation 1”) corresponding to a recognised scenario. Then we have generated a video from “animation 1”, processed this second video by the understanding system and generated a second 3D animation. As shown on Figure these two animations are almost identical which indicates that the understanding system does not make any difference between real videos and videos generated by the test system.

9.1.6 Discussion on simulation for video understanding

We have proposed a framework for the visualisation and the simulation of video understanding systems. Thanks to this framework, we were able to build a test system that generates the 3D animations corresponding to scenarios recognised by a video understanding system, or scenarios described by an expert. To realise this framework, we have defined six original models for modelling the virtual camera, the visualisation of the scene geometry, the human body, the actions, the scenarios and the scene-scenarios of individuals evolving in the scene.

These encouraging results open many perspectives. We are planning three main extensions of the framework. First, we plan to add functionalities to help the developer (e.g. expert of vision or scenario recognition) to understand the influence of algorithm parameters

setting, and to test the robustness of the interpretation. It will be interesting to generate test videos (animations) with noise phenomena (e.g. shadow) for simulating more realistically the input video of understanding systems. Second, we plan to extend the description language for the expert of the application domain (e.g. security agent) to be able to describe more complex scenarios and to visualise scenario variations, for example with respect to the variation of actor location.

Finally, we would like to define a unified framework using the same models for the understanding and the test system (e.g. models of individual, action and scenario).

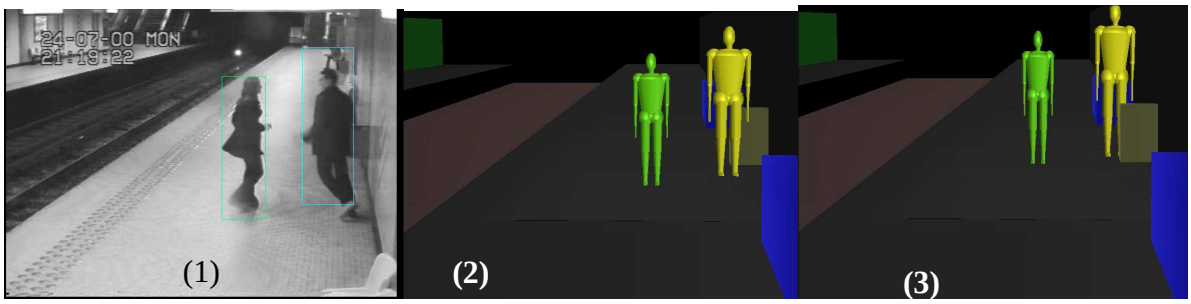


Figure 9.10*: illustration of the test system results: (1) detection of individuals corresponding to the output of VSIS, (2) generation of a first animation and a second video corresponding to the output of VSIS and (3) generation of a second animation corresponding to the second video processed a second time by VSIS.*

9.2 Frequent Composite Event Discovery in Videos

In this section, we propose a method for discovery of composite events in videos. The algorithm processes a set of primitive events such as simple spatial relations between objects obtained from a tracking system and outputs frequent event patterns which can be interpreted as frequent composite events. We use the APRIORI algorithm (called also association rules) from the field of data mining for efficient detection of frequent patterns. We adapt this algorithm to handle temporal uncertainty in the data without losing its computational effectiveness. It is formulated as a generic framework in which the context knowledge is clearly separated from the method in form of a similarity measure for comparison between two video activities and a library of primitive events serving as a basis for the composite events.

The problem of video event recognition has been studied in chapter 7. In most of the approaches explicit models of events are used which are either created manually or learned from labelled data. In this work we focus on the problem of detecting frequent complex activities without a model. In this work an event is a spatio-temporal property of an object in a time interval or a change of such a property, as described in chapter 7. An example of an event at a parking lot is ’vehicle on the road’ (see Figure. 9.11(a)). For the recognition of

events an algorithm was selected from the video understanding VSIP platform described in chapter 5. The events are formally defined in an event description language which allows us to define complex events in terms of simpler ones and in this way to build hierarchical structures of events. For instance, in a parking lot a complex event is a *'parking manoeuvre'* which consists of *'a vehicle on the road'*, *'a vehicle on the parking road'*, *'vehicle on a parking place'* and '*person coming out from the vehicle'*. The simplest events at the bottom of this description are referred as primitive while the complex ones are called composite. In order to provide a complete description of a domain an extensive library of events is needed containing a formal description for each possible behavior. To simplify the library and decrease the deployment efforts we retain in this library only the most simple and general events which are the primitive ones. These are simple spatial relations like *'object in a zone'* or *'object near another object'*. The frequent composite events are deduced in an automatic way from the set of all detected primitive events. For the latter task we adapt the data mining APRIORI algorithm ([1]) which uses the so called APRIORI property: the sub-patterns of frequent patterns are also frequent. Therefore, starting with short patterns we count the occurrence only of those patterns whose sub-patterns were marked as frequent in a previous step. In this way the search space is reduced. A pseudo code can be found in Algorithm (1):

APRIORI(S, l, s_{th})
INPUT: A set of states S = {s_1 . . . s_n}.
OUTPUT: All l-pattern classes $C^{(l)}$ with size s($C^{(l)}$) greater than a given size s_{th}.
1: natural i ← 2;
pattern set P_i ← Ø;
K_1 ← {{s_1} . . . {s_n}}
2: **while** i ≤ l **do**
3: P_i ← CREATENEXTLENGTHPATTERNS(K_{i-1}, i)
4: K'_i ← COMBINEPATTERNSINTOCLASSES(P_i)
5: K_i ← RETAINFREQUENTPATTERNS(K'_i, s_{th})
6: i ← i + 1
7: **end while**
Return K_l

In line 3 all pairs of patterns of length $i - 1$ from the previous iteration which have exactly $i - 2$ mutual elements are merged into patterns of length i. In the next step equal patterns are combined in classes. Finally, only the classes with a size greater than a given size threshold s_{th} are retained.

***Figure 9.11**: (a) Visualization of the event 'vehicle on the parking road'. (b) Manually defined zones in the scene.*

The difficulties of a direct application of the above algorithm in the domain of video event analysis arise from the uncertainty of the data: there are no equal but only similar occurrences of the same behavior type. Precisely, we have to answer two questions:

- How do we measure the number of occurrences of an activity (its frequency) and how the frequency of a sub-activity is related to the frequency of an activity (line 5)?
- How do we decide which patterns of events represent the same event type (line 4)?

The answer to these questions depends on the way we compare event patterns. Therefore, a similarity measure is necessary which evaluates to which extend two patterns represent the same activity. Using this similarity as a basis we can answer in a domain-independent way the above questions. In particular, the number of occurrences of a behavior and thus implicitly its frequency can be defined in a soft manner taking into account the similarity. Additionally, the APRIORI property does not hold in case of similarity because sub-patterns of patterns can be less similar than the patterns themselves. Therefore, we formulate in the next section a WEAK-APRIORI property which decreases the frequency threshold for shorter patterns in order to prevent losing sub-patterns of frequent patterns and thus to guarantee their detection in the merge step in line 3. The second point - combination of patterns into classes - uses an entropy-based clustering algorithm.

The domain knowledge is provided in two forms: besides the similarity we must specify a library of generic primitive event types. The occurrences of these events are the input to the algorithm and therefore serve as a basis for the composite events which can be detected.

We present a generic framework for high-level frequent event discovery. The context knowledge is clearly separated from the algorithm and thus makes the approach applicable in different domains for which primitive events and similarity can be defined. By solving the above issues we guarantee robust event detection in noisy environments. Moreover, the algorithm outputs a set of composite events which are hierarchically ordered and thus generates a clear event structure.

9.2.1 Related Work

Although the research in the field of unsupervised event detection and learning is at its beginning there are several approaches studied. One of the most widely used techniques is to learn in an unsupervised manner the topology of a Markov model. [Brand and Kettnaker, 2000] use 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 (HMM). This leads to a concentration of the transitional probabilities just on several states which correspond in most of the cases to meaningful events. Another approach is based on variable length Markov models which can express the dependence of a Markov state on more than one previous states [Galata et al, 2002]. While 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 et al, 2003]. The advantage of the above techniques for topology learning of Markov models is that they work in a completely unsupervised way. Additionally, they can be used after the learning phase to recognize efficiently the discovered events. On the other hand, these methods deal with simple events and are not capable of creating concept hierarchies. The states of the Markov models do not also correspond always to meaningful events. Another method was proposed by [Magee et al, 2004] who use inductive logic programming to generalize simple events. Although being promising this system was developed only for simple interactions without taking into account any temporal relations. For low-level event detection and learning several standard techniques were also used. [Walter et al, 2001] learn gestures by extracting and clustering prototypes of gesture components from trajectories of hand movements using the k-means algorithm. The authors use the Minimum Description Length principle to determine the optimal number of gesture components. All these approaches perform well in the case of the problem they are specified for but cannot be generalized. The data mining community has been studying the task of frequent pattern extraction for several decades. However, more emphasis is put on the computational effectiveness than on the robustness against noise. There are only a few approaches coping with some specific problems arising from uncertainty: temporal variability is addressed in [Sun et al, 2003]. Unfortunately, they do not propose a general way of dealing with different types of uncertainty.

9.2.2 A Model of Frequent Patterns

In this section we introduce some basic notation. We are interested in reoccurring structures in a primitive event set, expecting that these structures correspond to meaningful complex activities:

Definition 1: A *m-event pattern p* is a set of primitive events with cardinality m: $p = \{e_1 \ldots e_m\}$. The set of *all m-event patterns is denoted by* $P^{(m)}$. A *sub-pattern* of a pattern *p* is a pattern *p'* whose events are contained in *p*.

Although a pattern p is just a set, it describes implicitly a structure through the relations between its events. Additionally, a similarity measure is needed in order to express the degree of similarity between patterns:

Definition 2: *A similarity measure* $\mathrm{sim}^{(m)}(p_1, p_2)$ of order *m* between two *m*-patterns p_1 and p_2 is a mapping $\mathrm{sim}^{(m)} : P^{(m)} \times P^{(m)} \rightarrow [0, 1]$, where $P^{(m)}$ is the set of all *m*-patterns.

The concrete form of this mapping depends on the domain and therefore it is not possible to require further properties. Using the above notions of patterns and similarity we can build pattern classes.

Definition 3: A *m-pattern class* $C^{(m)}$ of order m is a set of m-event patterns. A *subclass* of a class *C* is a class *C'* whose patterns are sub-patterns of patterns in *C*.

We expect that a pattern class stands for a behavior type and thus contains examples of this behavior which should be similar to each other. This property is guaranteed through the class building step (line 4 in Algorithm 1) which is realized as clustering maximizing the weighted entropy of the class (see next section). A frequent subclasses of a class stands also for a frequent shorter activity which should be described additionally if it occurs more frequently than activity represented by the class itself. Using the APRIORI-algorithm (1) from section 1 we can maintain links from subclasses to classes formed in the next loop of the algorithm and retain the subclasses containing more examples than the number of examples in the class. In this way a hierarchical description of composite events can be achieved. Based on the above similarity we can define (in a domain independent way) a measure for the membership of a pattern in a pattern class and the pattern class size:

Definition 4: A class similarity measure $\mathrm{sim}^{(m)}$ (p, C) between a m-event pattern p and a pattern class C of order m is a mapping $\mathrm{sim}^{(m)} : \mathrm{P}^{(m)} \times \mathrm{K}^{(m)} \rightarrow [0, 1]$:

$$sim^{(m)}(p,C) \stackrel{def}{=} \frac{1}{|C|} \sum_{p' \in C} sim^{(m)}(p,p') \qquad (9.1)$$

where P(m) is the set of all m-patterns and K(m) is the set of all pattern classes of order m.

Definition 5: The size s(C) of a pattern class C is the sum of the similarities between all patterns in C to C:

$$s(C) \stackrel{def}{=} \sum_{p \in C} sim(p,C) \qquad (9.2)$$

A large class size means both a large number of examples of the behavior represented through the class (large number of summands in equation (9.2)) and also high degree of representativeness of the behavior through its examples (high value of the summands in equation (9.2)). In the case that we use an equality instead of a similarity the above class size

corresponds to the number of the patterns in the class: s(C) = |C|. The class size can be viewed also to be proportional to frequency of the event type described by the class.

9.2.3 Weak-Apriori Property

Using the above notation, the APRIORI property cited in the introduction and used as a basis for the algorithm can be expressed as follows:

$$s(C^{(m-1)}) \geq s(C^{(m)}),$$

where $C^{(m-1)}$ is a subclass of $C^{(m)}$. In the case of similarity instead of equality the APRIORI property does not have to be always valid: it can be violated by m-patterns which have (m − 1)-sub-patterns that are not so similar as the patterns itself. A reason may be a pair of strongly dissimilar events in the sub-patterns whose impact on the total similarity between the patterns loses strength with increasing pattern length. This property can hold for all pairs of patterns in a class and thus leads to a smaller size of its subclasses. In this case the APRIORI property will not hold.

A remedy is a different version of the APRIORI property which requires a weaker bound on the size of the subclass:

Definition 6: WEAK-APRIORI-Property: For all sub-classes $C^{(m-1)}$ of a pattern class $C^{(m)}$ holds:

$$s(C^{(m-1)}) \geq g(m)s(C^{(m)}) - f(m)|C^{(m-1)}|$$

where g(m) and f(m) are positive functions of the class order m. The functions g(m) and f(m) serve as a correction for the smaller size of a subclass. The concrete form of f(m) and g(m) depends on the similarity measure and an instantiation is given in equation (9.3) for the similarity defined there. Provided the property in Definition 6 holds for a similarity measure we can see that Algorithm 1 correctly recognizes all l-classes with size at least s^{th} if we use in line 5 in the i^{th} loop for the size of a class C the following dynamic threshold ($i \in [2, l]$):

$$s_{th}^{(i)}(|C|) = s_{th} \prod_{k=i+1}^{m} g(k) - |C| \sum_{k=i+1}^{m} f(k) \prod_{j=i+1}^{k-1} g(j) \qquad (9.3)$$

This threshold results from the WEAK-APRIORI property by propagating the class size decrease from step l till step i. It guarantees that we will not miss in earlier loops of the algorithm any subclasses which can be used to construct and consequently detect all classes of order l with size at least s_{th}. A proof of this fact can be found in [Toshev, 2005].

9.2.4 Pattern Clustering

The objective of line 4 in Algorithm (1) is to classify the generated patterns into clusters which correspond to the same activity type. Precisely, this clustering step must result in large coherent classes which are also clearly distinguishable from each other.

We propose an entropy-based agglomerative hierarchical clustering method [Li et al, 2004]. Starting with classes containing only one pattern, in each successive step we merge those two classes C_i and C_j whose merge leads to the highest increase of a utility function $U(C_i, C_j)$. This function is based on the weighted entropy $H_w(C)$ of a class C which is defined as the product of the class size s(C) and the class entropy H(C). The entropy H(C) is defined by interpreting a class as a random variable with values equal to the patterns and probabilities $P_C(\cdot)$ of those values proportional to the class similarities:

$$P_C(p) \stackrel{def}{=} \frac{sim(p,C)}{\sum_{p' \in C} sim(p',C)} = \frac{sim(p,C)}{s(C)} \tag{9.4}$$

where $p \in C$. From the above definitions follows:

$$H_w(C) \stackrel{def}{=} s(C)H(C) = \sum_{p \in C} sim(p,C) \log\left(\frac{s(C)}{sim(p,C)}\right) \tag{9.5}$$

A class of high quality is characterized by a large value of the weighted entropy: large class size indicates a lot of mutually similar patterns in the class and large class entropy indicates good coherence and lack of outliers.

During the clustering we merge classes if this step leads to an increase of the weighted entropy of the new class compared with the old classes:

$$U(C_i, C_j) \stackrel{def}{=} H_w(C_i \cup C_j) - H_w(C_i) - H_w(C_j) > 0 \tag{9.6}$$

We iterate the above merge step until no further increase can be achieved. In this case we have hopefully all patterns describing the same activity type in one class.

9.2.5 Similarity Measure

In this section we describe informally a similarity measure between video events. As an application we use videos recorded at a parking lot divided into zones (see Figure 9.11(b)) in which vehicles and persons are tracked. The system recognizes the events *'an object being in a zone'* and *'an object close to another object'*. These events can be described completely by a tuple of attributes: event name, object types, zone name and start/end time.

A similarity measure compares event patterns using the attributes of the primitive events. We distinguish between symbolic and numeric attributes. In the above domain examples for the first attribute types are event name, object type, and zone name; examples for the second type are event duration, event start/end time. The former are in most of the cases unordered and can be compared only for equality while the latter must be treated with a soft comparison function:

$$C_{symb}(x,y) \overset{def}{=} \begin{cases} 1, x = y \\ 0, otherwise \end{cases}$$

$$C_{num}(x,y) \overset{def}{=} e^{-\frac{(x-y)^2}{axy}}$$

where $x, y \in R$ and $a \in R^+$ is a parameter. The usage of the denominator xy makes the function more sensitive to differences between small values. This corresponds to the assumption that attributes with small values are more susceptible to changes.

Another taxonomy of the attributes is based on their usage. Some attributes can be compared directly like event names, object types, zone name, and event duration. In other cases we must evaluate an attribute in relation with attributes of other events: comparing event start/end times directly does not make sense but only in the context of another event in order to express the temporal relation between them. Based on this distinction between attributes we define the *attribute-structure similarity*:

$$sim_{a-s}(p_i, p_j) \overset{def}{=} w_a attr(p_i, p_j) + w_s struct(p_i, p_j)$$

with $w_a, w_s \geq 0$, $w_a + w_s = 1$, which compares not only the properties of the primitive events in a separate manner through its *attribute similarity* $attr(p_i, p_j)$ but also compares the structures of the patterns expressed in terms of the temporal relations between the events in a pattern formulated as *structure similarity* $struct(p_i, p_j)$. Using the above two notions we combine direct comparison with principles of analogy reasoning.

The above components of the similarity must be defined manually for each domain. Hereby, the attributes of a primitive event $e_k^{(i)}$ of a *m*-pattern p_i, $k = 1 \ldots m$, are compared with the attributes of exactly one event $e_k^{(j)}$ of the other m-pattern p_j using an appropriate compare function: symbolic attributes are compared only about equality; for numeric attributes x and y we use $e^{-\frac{(x-y)^2}{axxy}}$. The similarity between patterns is the average of the similarities between all primitive events:

$$attr(e_k^{(i)}, e_k^{(j)}) \stackrel{def}{=} \prod_{a^{(i)} \in D(e_k^{(i)})} C(a^{(i)}, c_{i,j}(a^{(i)}))$$

$$attr(p_{i,}, p_j) \stackrel{def}{=} \frac{1}{m} \sum_{k=1}^{m} attr(e_k^{(i)}, e_k^{(j)})$$

where $c_{i,j}(a) \in D(e_k^{(j)})$ is the corresponding attribute in $e_k^{(j)}$ to an attribute *a* from $e_k^{(i)}$. The *structure similarity* compares the temporal relations of the primitive events. The temporal relation between $e_k^{(i)}$ and $e_l^{(i)}$ from p_i can be compared with the temporal relation between $e_k^{(j)}$ and $e_l^{(j)}$ from p_j as follows:

$$struct(e_k^{(i)}, e_l^{(i)}, e_k^{(j)}, e_l^{(j)}) \stackrel{def}{=} C_{num}(dist(e_k^{(i)}, e_l^{(i)}), dist(e_k^{(j)}, e_l^{(j)}))$$

where dist(·,·) compares the temporal distance between events:

$$dist(e_i, e_j) \stackrel{def}{=} \begin{cases} d(e_i, e_j), & |d(e_i, e_j| \leq |d(e_j, e_i)| \\ d(e_j, e_i), & otherwise \end{cases}$$

$D(e_i, e_j) = b(e_i) - e(e_j)$ with $b(e)$ and $e(e)$ start and end time of an event *e*. Finally:

$$struct(p_i, p_j) \stackrel{def}{=} \frac{1}{m(m-1)} \sum_{k,l=1, k \neq l}^{m} struct(e_k^{(i)}, e_l^{(i)}, e_k^{(j)}, e_l^{(j)})$$

Configuration		Sample pattern 1		Sample pattern 2	
Pert.	Noise	length	rank	length	rank
5	0%	5	1	6	2
	25%	5	1	6	1
	50%	5	1	6	2
10	0%	5	1	6	2
	25%	5	2	5	1
	50%	5	1	6	1

Table 9.1. *Results with synthetic data. For each configuration of perturbation variance, noise portion, and sample pattern the rank and the length of the most meaningful detected patterns are displayed.*

It can be shown that the attribute-similarity similarity satisfies the WEAK-APRIORI property [Toshev, 2005]:

$$s(C^{(m-1)}) \geq \underbrace{(1-\frac{1}{m-1})}_{g(m)} s(C^{(m)}) - \underbrace{(\frac{w_a}{m-1}+\frac{2w_s}{m-2})}_{f(m)} |C^{(m-1)}|$$

With increasing pattern length *m* the bound on the right side of the above inequality converges towards $s(C^{(m)})$ because $g(m) \overset{m\to\infty}{\to} 1$ and $f(m) \overset{m\to\infty}{\to} 0$. Hence, the violation of the initial APRIORI property decreases with increasing pattern length and so the computational effectiveness of the initial algorithm is preserved.

9.2.6 Evaluation

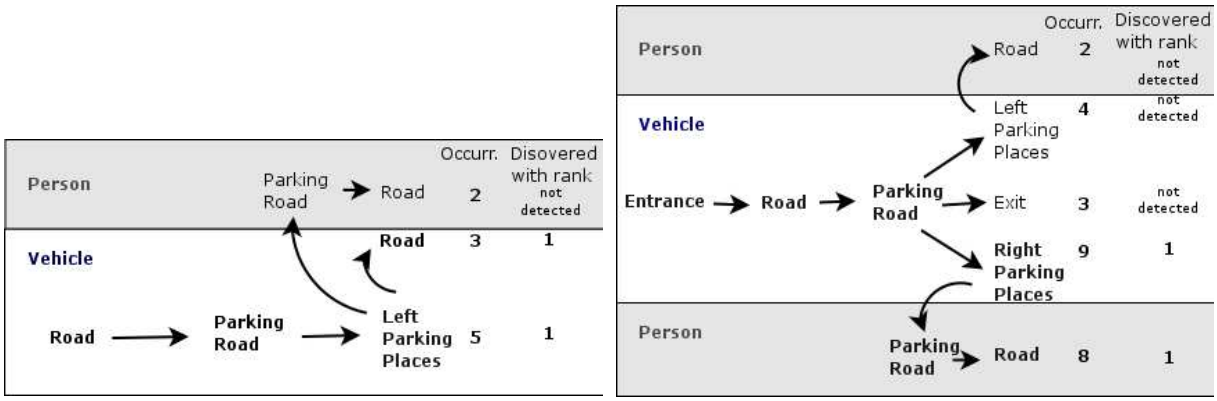
We test our approach on two types of sets containing frequent composite events: synthetic data and data from the parking lot monitoring domain. In both cases we describe the data manually and compare the most frequent patterns found by the algorithm with this description. We assess (i) which sub-patterns of the expected event patterns were recovered and (ii) what is their frequency compared with the frequency of the other detected patterns. The latter aspect is quantified in form of a rank: a pattern has rank k if it is the k^{th} most frequent.

The synthetic data was generated from two manually created sample patterns of length 6. For each sample pattern 6 test sets were created as follows. 5 copies of each of these patterns were perturbed and randomly positioned in a time interval of 15000 time frames. Precisely, the start/end times were perturbed with a Gaussian noise with variance equal to 5 and 10 time frames and mean 0 and thus resulting in two sets for each sample pattern. Additionally, noisy events were added to each set whose portion of all events equals to 0%, 25% or 50%. The resulting sets and the results of the experiments for each set are displayed in table 9.1 and show that in all test runs at least a 5-subpattern of the optimal 6-pattern was recovered. This pattern was in 65% of the cases the most frequent and in the remaining cases the second frequent. In the case when the expected pattern was the second frequent, the most frequent pattern was caused by events which coincidentally form patterns. The performance of the algorithm was stable even in the worst case of high perturbation and 50% noise.

In order to evaluate the technique in a real situation we apply it to the parking lot domain. We process approximately 4 hours video from two days resulting in approximately 200 hundred primitive events representing approximately 20 composite events divided into two sets, one set for each day. These sets are presented in Figure 9.12 together with the results. In both cases the most frequent complex events were detected and they had in both cases rank 1. These events correspond to the manoeuvre parking and thus the most natural activity in the domain was detected.

The reasons for not obtaining a pattern of full length are strong perturbations in some cases and imperfect tracking which splits sometimes one primitive event into several due to lost objects. The computational cost reduces rapidly with each iteration of the APRIORI algorithm: in each step beyond the 3rd one less than 1% of all possible patterns were taken into account. This shows the effectiveness of the WEAK-APRIORI property in the case of the

attribute-similarity measure: the bound becomes more restrictive with increasing pattern length for which the number of possible patterns increases. On a machine with a 3.4 GHz Intel Pentium CPU the algorithm needed between 30 and 90 minutes for each run.



(a) BOREL PARKING 11 03 (b) BOREL PARKING 21 03

Figure 9.12: *Manually created description of the data and results. Each flow displays a sequence of primitive events of 'vehicle or person in a zone' with the zone name and object type given. The occurrence refers to data descriptions. The discovered complex events are marked green with their rank to the right.*

9.2.7 Discussion on Event Discovery

In this section, we have presented an approach for detecting frequent composite events using the APRIORI algorithm from the data mining field. We were able to adapt this algorithm to handle uncertainty without losing its computational attractiveness. It discovers clear composite events and structures them hierarchically. The proposed method is built as a general framework for which context knowledge in form of a similarity measure and a generic library of primitive events must be specified. In the future we would like to investigate other similarity measures based, for example, on probabilities. In a similarity we can incorporate not only uncertainty of the temporal attributes but also of the remaining attributes such as labels, for examples. Another topic is the analysis of the detected frequent patterns in order to create a compact and expressive model of the whole data. A different topic is to improve the performance of the method. This can be achieved through better implementation but also through integrating the operation from the line 3 in Algorithm (1) as another merge step in the clustering from line 4: we can create longer patterns by combining classes whose patterns have large number of overlapping states. In this way the whole algorithm can be represented as a clustering.

9.3 Conclusion on Knowledge Acquisition

In this chapter, we have presented two tools for acquiring a priori knowledge and more precisely the scenarios to be recognised. The first tool, by simulating and visualizing 3D animations helps end-users to define their scenarios of interest. It has been successfully used to model several scenarios for two applications. This tool can also be useful to understand the functioning of the whole video understanding system and to help tuning system parameters. However, the tool is not user-friendly and mature enough to be fully operational and to simulate the whole diversity of the real world.

The second tool aims at learning the frequent combinations of primitive events called event patterns. These event patterns correspond to the frequent activities occurring in the observed scene and are for end-users potential scenarios of interest. This tool is useful for pre-defining everyday activities, especially in monitoring applications. However, the learnt frequent scenarios are not always interesting and scenarios of interest are not necessary frequent. Thus, this work needs to be refined to include, for instance, contextual information and end-users feedback through an interactive interface to guide the extraction of scenarios of interest.

All these tools on knowledge acquisition for video understanding systems are useful and will become critical soon, in order to develop 24/7 video understanding systems functioning on large scales. However, these works are still at a preliminary stage of development. Therefore, more works are expected to flourish in this knowledge acquisition domain.

Chapter 10

Conclusion and Perspectives

This document aims at summarising past experiences on scene understanding systems, focusing on main achievements and underlying new trends for the near future. We have first studied this domain following the three worlds involved by the scene understanding process: perceptual world, physical world and semantic world.

The **perceptual world** includes all features (e.g. colour, edge, 2D shape, 3D trajectory, sound, contact information) describing a scene and in particular the physical objects evolving in the scene. This world is characterised by its uncertainty and redundancy. Here, the scene understanding process is mainly to compensate for missing or misplaced data, to filtered outliers (i.e. inconsistency with the majority of data), to remove aberrant data (i.e. out of range), to cluster similar information (using for instance, statistical models) and to categorise combinations of features. To explore the perceptual world, in the chapter 5, on **vision programs**, we have described the large variety of video processing programs and the diversity of videos depicting a scene. We have explained how VSIP platform can help building scene understanding systems, based on two types of programs: (1) generic programs for the main video understanding tasks and for common video characteristics, (2) advanced programs for specific tasks and for handling particular situations. Nevertheless, object detection and extracting perceptual features will stay an open issue for still a long period of time, in particular in real world applications containing challenging situations, such as moving cameras, crowd, and limited processing capacities.

To improve the perception of dynamic 3D scenes, we are planning to work towards three directions. First, we want to compute better perceptual features for characterising the objects of interest. For instance, we aim at exploring new visual features (e.g. feature points such as KLT (Kanade Luca Tomasi), local 2D descriptors such as HOG (Histogram of Gradients) or the ones described by Viola and Jones) and other sensor features (e.g. environment and physiological sensors, audio, radar features). In this objective, we have started to supervise a PhD Nadia Zouba [Bremond, 55] on the combination of heterogeneous sensor network for homecare applications. A second direction consists in designing robust algorithms for characterising the object of interest shape in order to infer their postures and gestures. Thanks to the proposed new perceptual features and previous work [Bremond, 10], these algorithms should be independent from the camera view point and still effective in complex situations (e.g. static and dynamic occlusions, moving background, crowd, interactions between objects of interest and interactions with contextual objects). We will explore these new types of algorithms with PhD M.B. Kaaniche who is studying gesture recognition in complex scenes. The third direction is to establish under which hypotheses the algorithms are valid, and to understand their limits. This topic is currently under process with PhD A. T. Nghiem who is studying the relationships between algorithm parameters and algorithm performance. The objective is then to optimise the use of perception algorithms and their combinations. Based on this algorithm characterisation, the advanced research axes proposed in the introduction can be fruitfully explored. Moreover, we believe that pursuing these other axes is important for two reasons: first, in some specific conditions (e.g. structured environment), the challenge of the perceptual world can be solved and second, some applications do not require perfect vision results (i.e. perfect object detection and tracking). These other research axes constitute strong active trends in scene understanding.

The **physical world** contains all the physical objects of the real world, especially the ones in motion. Reasoning in this world is near the logical inferences performed by human beings (i.e. common sense). It includes geometric, physical laws, spatio-temporal and logic. Here, the scene understanding process aims at maintaining a coherent and multi-modal representation of the real world throughout time. To study the physical world, in the chapter 6 **on maintaining 3D coherency throughout time**, we have reviewed several works. These works are organized following two directions: coherency throughout time and coherency of multi-modal information in the 3D space. These works have presented some solutions to bridge the gap between signal and semantic levels. The key issues are:

- Building a common knowledge representation for combining all information describing the scene.
- Modelling and managing the uncertainty and the incompleteness of data and models characterizing the scene and its dynamics.

These works have described some efficient information fusion algorithms that can lead to a coherent understanding of the observed scene. Despites these success, there are still some limitations in particular situations, depending on the amount of uncertainty and lack of information characterising the input data.

Nevertheless, thanks to a precise formalism combining uncertain information coming from heterogeneous sensors, we are able in several cases to understand the evolution of the mobile objects in the scene and so to recognise successfully the events characterising the behaviours of these objects of interest. For us, a natural trend in this information fusion domain consists first in extending the common knowledge representation to combine in an easy way all information coming from different sources. This trend also consists in modelling all type of uncertainty and incompleteness for both data and scene models. Therefore, a current work is to integrate the largest diversity of sensors to get a complete multi-modal

perception of the scene. This objective is addressed in two different aspects by PhD B. Bui [Bremond 43] on people categorisation using optical cells and video cameras and by PhD N. Zouba [Bremond, 55] on multi-sensor scene understanding.

The **semantic world** gathers all types of events, relations and concepts related to the activities occurring in the scene. This world is mostly symbolic and linked to the application domain. Reasoning in this world consists in specific causal and logic inferences and verifying spatio-temporal constraints which make sense only in a particular domain with a specific objective. For the semantic world, in the chapter 7 on **Event Recognition**, we have presented work on the two types of approach for recognising events: numerical and symbolic. The numerical approaches are well adapted to events closely related to vision features involving few actors (mostly one actor with or without interactions with his/her environment). However, their effectiveness depends on the training stage as the parameters are learned with training video sequences containing positive and negative event samples. Moreover, developing these recognition algorithms is not straightforward and required a strong expertise in vision. They are relatively easy to implement but their tuning for a particular application is a main problem. In addition, they suffer from a difficult modelling stage of events involving multiple actors (i.e., objects of interest). The reason is that the combination of events is often exponential given the number of actors, leading to the collection of a huge training set. Finally, their description is not declarative and it is often difficult to understand how they work (especially for Neuronal Networks). In consequence, it is relatively difficult to modify them or to add a priori knowledge.

Therefore, new learning mechanisms need to be defined to ease the construction of event recognition algorithms: to select the training video sets and to learn the structure and the parameters of the network. In this objective, for instance a first goal is to propose a set of generic primitive event concepts and to link them with specific algorithms characterised by explicit knowledge, in particular on the scene context (e.g. narrow corridor). For instance, these event concepts can express changes in object shape (e.g. bending) and/or (e.g. zigzagging) trajectory. The challenge is to describe this knowledge in a declarative way and to link it to a description of the observed real scene and of the application objectives. This topic will be partially studied by PhD M.Zuniga who is working on learning primitive event concepts related to shapes and by PhD Thi Lan Le working on event concepts related to trajectories. A second goal is to learn the semantics of the observed dynamic 3D scene based for instance on the statistic analysis of object trajectories. For that, in the IST CARETAKER project we will learn the main people journeys in a subway network, we will cluster people trajectories into meaningful categories and we will learn the topology of the observed scene through the massive and long-term recording of perceptual data coming from a network of cameras and microphones.

Symbolic approaches are well adapted to model complex temporal events involving multiple actors. Despite a large number of potential combinations of events to be explored, well designed algorithms can still recognise events in real-time. As we have shown in chapter 7, when good vision results are obtained, the symbolic event recognition algorithms can recognise all scenarios. An intuitive language has been also defined to help the end-users to describe their scenarios of interest. The main problem of symbolic approaches is the mechanism to handle the errors of vision algorithms. Most of the time, the recognition algorithms take the hypothesis that vision algorithms do not make errors and generate perfect tracked objects. Moreover, modelling the scenarios specified by the end-users is an error

prone process (i.e. models of scenario are often vaguely defined) at the scenario recognition level, which can be time consuming, especially in case of monitoring all everyday activities.

Therefore, we are planning to work on two main improvements. First, managing the uncertainty of vision features (in particular the lost of tracked objects), is a crucial point. To reach this objective, we are planning to extend our Scenario Description Language for modelling explicitly the uncertainty of vision features. Together with this language extension, we would like to propose mechanisms to propagate this uncertainty through all the layers of the event recognition process. These improvements will be explored through the ITEA SERKET project whose goal is to enhance security technologies. The second improvement consists in learning the scenario models of interest. This issue becomes essential while dealing with video monitoring applications and with a large amount of scenario models. This research direction is related to knowledge acquisition and is detailed below in the corresponding section.

After studying the issues and perceptive of the scene understanding process, in these perceptual world, physical world and semantic world, we have addressed problems more related to scene understanding **systems**; once we have shown that a computer program can understand a scene in few situations, how this processing can be generalized in most real cases. I believe that new trends in scene understanding rely on this generalising process, on the mechanisms to acquire and capitalise knowledge and on making systems adaptable, user-centered and in the same time fully autonomous. We have addressed these topics through two directions: (1) evaluation and learning knowledge of systems and (2) knowledge acquisition through end-user interactions.

Concerning **Evaluation and Learning,** in the chapter 8, we have presented several works on performance evaluation of video understanding algorithms and on learning techniques for parameter tuning.

On evaluation, we have described a methodology and a tool for evaluating the performance of video understanding systems. Given a video data base, ground truth and a set of metrics, the user can adapt the tool to evaluate a specific video understanding task (e.g. object tracking), or specific scene conditions (e.g. sudden illumination changes). We have also presented an algorithm to evaluate automatically the quality of a segmentation program for object detection in order to re-compute dynamically the reference image when it is necessary.

On learning knowledge of systems, we have described an algorithm to learn automatically the parameters of a segmentation program, by computing a characterisation of the illumination conditions of a given scene. This is the first stage towards the dynamic configuration of video understanding systems.

These evaluation and learning mechanisms are at a preliminary stage, but are necessary to obtain an effective video understanding system, operational 24/7 at a large scale. Therefore, more efforts still need to be done. In particular, an appropriate formalism needs to be defined to make these mechanisms seamlessly integrated in new scene understanding systems. Moreover, for parameter tuning, we aim at characterising precisely and exhaustively all algorithm parameters, their dependencies between themselves and between a characterisation of the input data (i.e. videos) and their impact on system performance. Given this characterisation and a set of reference input data associated with ground-truth, we will be

able to automatically and dynamically configurate any scene understanding system. This is the subject of PhD A. T. Nghiem.

Concerning **knowledge acquisition** through end-user interactions**,** in the chapter 9, we have presented two tools for acquiring a priori knowledge and in particular the scenarios to be recognised. The first tool, by simulating and visualizing 3D animations helps end-users to define their scenarios of interest. It has been successfully used to model several scenarios in two applications. This tool can be also useful to understand the functioning of the whole video understanding system and to help tuning system parameters. However, the tool is not user-friendly and mature enough to be fully operational and to simulate the whole diversity of the real world. Research in this domain is still for us an appealing topic. Especially, we are planning to work on a generic and complete formalism to describe realistic dynamic 3D scenes (to be visualised as virtual scenes) as imagined by end-users.

The second tool aims at learning the frequent combinations of primitive events called event patterns. These event patterns correspond to the frequent activities occurring in the observed scene and are for end-users potential scenarios of interest. This tool is useful for pre-defining everyday activities, especially in monitoring applications. However, the learnt frequent scenarios are not always interesting and scenarios of interest are not necessary frequent. Thus, we are planning to refine this tool to include, for instance, contextual information and end-users feedback through an interactive interface to guide the extraction of scenarios of interest. In the same way, we want to conceive tools to visualise and explore the event space structured through the computed event patterns. Another trend consists for us to reduce the processing time of the clustering techniques by taking benefit of new data mining algorithms. This topic will be partially studied through the IST CARETAKER project.

All these topics on evaluation, learning and on knowledge acquisition for video understanding systems are sensitive and will become critical soon, in order to develop 24/7 video understanding systems working on a large scale. However, these works are still at a preliminary stage of development, waiting for the main scene understanding process to be sufficiently mastered. Therefore, more works are expected to flourish in these learning and knowledge acquisition domains.

Bibliography

Doctorate thesis

[1] François Bremond, « Environnement de résolution de problèmes pour l'interprétation de séquences d'images (Problem solving environment for image sequence interpretation) », Université de Nice-Sophia Antipolis, INRIA. Octobre 1997.

Book chapters

[2] N. Chleq , F. Brémond and M. Thonnat. "Image Understanding for Prevention of Vandalism in Metro Stations". Dans The Kluwer International Series in Engineering and Computer Science". Serie "Advanced Video-Based Surveillance Systems", edited by Carlo S. Regazzoni, Gianni Fabri and Gianni Vernazza. Kluwer Academic Publishers. pp106-116. 1999.

[3] Frederic Cupillard, Francois Bremond and Monique Thonnat, Tracking Group of People for Video Surveillance, in *"The Kluwer International Series in Computer Vision and Distributed Processing" Serie "Video-Based Surveillance Systems"*, edited by Paolo Remagnino,Graeme A. Jones, Nikos Paragios and Carlo S. Regazzoni. Kluwer Academic Publishers. pp89-100, 2002.

International journals with a reviewing committee

[4] F. Brémond et M. Thonnat. Issues of representing context illustrated by video-surveillance applications. Dans *International Journal of Human-Computer Studies* (1998) 48, 375-391.

[5] F. Brémond et M. Thonnat. Tracking Multiple Nonrigid Objects in Video sequences. Dans *IEEE Transactions On Circuits and Systems for VideoTechnology*, Vol. 8. No. 5, Sep 1998.

[6] G. Médioni, I. Cohen, F. Brémond, S. Hongeng et R. Nevatia. Event Detection and Analysis from Video Streams. Dans *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23. No. 8, pp. 873-889, Aug 01.

[7] T. Vu, F. Brémond and M. Thonnat. Human behaviour visualisation and simulation for automatic video understanding. *Journal of* WSCG (Winter School of Computer Graphics), Volume 10, No.1-3, 2002, pp485-492 ISSN1213-6972.

[8] S. Hongeng, R. Nevatia and F. Bremond. Video-Based Event Recognition: Activity Representation and Probabilistic Recognition Methods. In *Computer Vision and Image Understanding (CVIU)*, Volume 96, Issue 2 , November 2004, Pages 129-162, Special Issue on Event Detection in Video 2004.

[9] Alberto Avanzi, Francois Bremond, Christophe Tornieri and Monique Thonnat, Design and Assessment of an Intelligent Activity Monitoring Platform, *in EURASIP Journal on Applied Signal Processing*, special issue in *"Advances in Intelligent Vision Systems: Methods and Applications"*, 2005:14, pp.2359-2374.

[10] B. Boulay, F. Brémond and M. Thonnat, Applying 3D Human Model in a Posture Recognition System. *In Pattern Recognition Letter* , Special Issue on vision for Crime Detection and Prevention, Vol.27, No15, pp. 1788-1796, Nov 2006.

[11] F. Brémond, M. Thonnat and M. Zuniga, Video Understanding Framework For Automatic Behavior Recognition. *In Behavior Research Methods*, 38(3), pp. 416-426, 2006.

[12] F. Fusier, V. Valentin, F. Brémond, M. Thonnat, M. Borg D. Thirde and J. Ferryman, Video Understanding for Complex Activity Recognition. *In the Machine Vision and Applications Journal, 2006,* to be published.

[13] B. Georis, F. Brémond and M. Thonnat, Real-Time Control of Video Surveillance Systems with Program Supervision Techniques. *In the Machine Vision and Applications Journal, 2006,* to be published.

International conferences with a reviewing committee

[14] F. Brémond et M. Thonnat. A contex representation for surveillance systems. Dans *proc. Of the Workshop on Conceptual Descriptions from Images at the European Conference on Computer Vision* (ECCV), Cambridge (U.K.), April 1996.

[15] J.C. Ossola, F. Brémond et M. Thonnat. A Communication Level in a Distributed Architecture for Object Recognition. Dans *proc. of the 8th International Conference on Systems Research Informatics and Cybernetics* (Intersymp'96). Baden-Baden, 14/18 August 1996.

[16] F. Brémond et M. Thonnat. Issues in representing context illustrated by scene interpretation applications. Dans *proc. of the International and Interdisciplinary Conference on Modeling and Using Context* (CONTEXT-97), Rio de Janeiro, February 1997.

[17] F. Brémond et M. Thonnat. Analysis of human activities described by image sequences. Dans *proc. of the 10th International FLAIRS Conference*, Florida, May 1997.

[18] F. Brémond et M. Thonnat. Tracking multiple non-rigid objects in a cluttered scene. Dans *proc. of the 10th Scandinavian Conference on Image Analysis* (SCIA), Lappeenranta (Finland), June 1997.

[19] F. Brémond et M. Thonnat. Recognition of scenarios describing human activities. Dans *proc. of the International Workshop on Dynamic Scene Recognition from Sensor Data* , Toulouse (France), June 1997.

[20] F. Brémond et M. Thonnat. Object tracking and scenario recognition for video-surveillance. Dans *the poster session of the 15th International Joint Conference on Artificial Intelligence* (IJCAI), Nagoya (Japan), August 1997.

[21] F. Brémond et G. Médioni.Scenario Recognition in Airborne VideoImagery. Dans *Workshop of Computer Vision and Pattern Recognition* (CVPR) on Interpretation of Visual Motion, Santa Barbara (USA), June 1998.

[22] F. Brémond and G. Medioni. Scenario Recognition in Airborne Video Imagery. Dans *Proceedings of DARPA Image Understanding Workshop*, pages 211-216, Monterey, CA, 1998.

[23] S. Hongeng, F. Brémond and R. Nevatia. Representation and Optimal Recognition of Human Activities. Dans IEEE *Proceedings of Computer Vision and Pattern Recognition*, pages 818-825, Hilton Head Island, SC, 2000.

[24] S. Hongeng, F. Brémond and R. Nevatia. Bayesian Framework for Video Surveillance Applications. Dans *Proceedings of the International Conference on Pattern Recognition*, pages 164-170, Barcelona, Spain, 2000.

[25] F. Cupillard, F. Brémond and M. Thonnat. Tracking groups of people for visual surveilllance. Dans *2nd European Workshop on Advanced Video-based Surveillance Systems*, Sep 4 , 2001 Kingston upon Thames, UK.

[26] A. Avanzi, F. Brémond and M. Thonnat. Tracking Multiple Individuals for Video Communication. Dans *Proc. of the IEEE Signal Processing Society, International Conference on Image Processing*, Thessaloniki, Greece, oct 2001.

[27] T. Vu, F. Brémond and M. Thonnat. Human Behaviors Visualization and Simulation for Automatic Video Understanding. Pour *the International Conference in Central Europe on Computer Graphics* (WSCG'2002). En préparation.

[28] Vu, F. Brémond and M. Thonnat, Temporal Constraints for Video Interpretation, The *15-th European Conference on Artificial Intelligence ECAI'2002, W9: Modelling and Solving Problems with Constraints*, Lyon, France, 21-26 July 2002.

[29] T. Vu, F. Brémond and M. Thonnat, Video surveillance: human behaviour representation and on-line recognition, The *Sixth International Conference on*

Knowledge-Based Intelligent Information & Engineering Systems (KES'2002), Podere d'Ombriano Crema Italy, 16, 17 & 18 September 2002.

[30] F. Cupillard, F. Brémond and M. Thonnat, Group Behavior Recognition With Multiple Cameras, *In IEEE Proc. of the Workshop on Applications of Computer Vision* - ACV2002, Orlando USA, Dec 3,4 2002.

[31] F. Cupillard, F. Brémond and M. Thonnat, Behaviour Recognition for Individuals, Groups of people and Crowd, *In IEE Proc. of the IDSS Symposium - Intelligent Distributed Surveillance Systems*, London, UK, 26 February 2003.

[32] C. Tornieri, F. Brémond and M. Thonnat, Reference Image for a Visual Surveillance Platform, *In IEE Proc. of the IDSS Symposium - Intelligent Distributed Surveillance Systems*, London, UK, 26 February 2003.

[33] T. Vu, F. Brémond and M. Thonnat, Automatic Video Interpretation: A Recognition Algorithm for Temporal Scenarios Based on Pre-compiled Scenario Models, The *3rd International Conference on Computer Vision Systems* - ICVS 2003, Graz, Austria, April 2003.

[34] N. Moënne-Loccoz, F. Brémond and M. Thonnat, Recurrent Bayesian Network for the Recognition of Human Behaviors from Video, The *3rd International Conference on Computer Vision Systems* - ICVS 2003, Graz, Austria, April 2003.

[35] T. Vu, F. Brémond and M. Thonnat, Automatic Video Interpretation: A Novel Algorithm for Temporal Scenario Recognition. The *Eighteenth International Joint Conference on Artificial Intelligence* (IJCAI'03), Acapulco, Mexico, 9-15 August 2003.

[36] B. Boulay, F. Brémond and M. Thonnat, Human Posture Recognition in Video Sequence. The *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS)*, Nice, France on October 11-12 2003.

[37] B. Georis, F. Brémond, M. Thonnat and B. Macq, Use of an Evaluation and Diagnosis Method to Improve Tracking Performances . The *3rd IASTED International Conference on Visualization, Imaging and Image Proceeding VIIP 03* , September 8-10 2003, Benalmadera, Spain.

[38] B. Georis, M. Mazière, F. Brémond and M. Thonnat, A Video Interpretation Platform Applied to Bank Agency Monitoring. The *Intelligent Distributed Surveillance Systems Workshop*, London, UK, February 23th, 2004.

[39] F. Cupillard, A. Avanzi, F. Brémond and M. Thonnat, Video Understanding for Metro Surveillance. The *IEEE ICNSC 2004 in the special session on Intelligent Transportation Systems*, Taiwan, March 2004.

[40] J. Bannour, B. Georis, F. Brémond and M. Thonnat, Generation of 3D Animations from Scenario Models. The *IASTED 4rd International Conference on Visualization, Imaging and Image Processing*, Marbella, Spain, September 6-8th, 2004.

[41] M. Borg, D. Thirde, J. Ferryman, F. Fusier, F. Brémond and M. Thonnat, An Integrated Vision System for Aircraft Activity Monitoring. The *IEEE PETS2005 workshop* in Breckenridge, Colorado, on January 7 2005.

[42] M. Borg, D. Thirde, J. Ferryman, F. Fusier, V. Valentin, F. Brémond and M. Thonnat, Video Event Recognition for Aircraft Activity Monitoring. The *8th International IEEE Conference on Intelligent Transportation Systems (ITSC´05)*, in Vienna, Austria on the 13th to 16th of September 2005.

[43] B. Bui, F. Brémond, M. Thonnat and JC. Faure, "Shape Recognition Based on a Video and Multi-Sensor System". The *IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS 2005)*, Como, Italie, on September 15-16, 2005.

[44] D. Thirde, M. Borg, J. Ferryman, F. Fusier, V. Valentin, F. Brémond and M. Thonnat, "Video Event Recognition for Aircraft Activity Monitoring". The *IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS 2005)*, Como, Italie, on September 15-16, 2005.

[45] M. Borg, D. Thirde, J. Ferryman, F. Fusier, V. Valentin, F. Brémond and M. Thonnat and J. Aguilera and M. Kampel, "Visual Surveillance for Aircraft Activity Monitoring". The *Proceedings of the Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Beijing, China, 15-16th October 2005.

[46] J. Ferryman, M. Borg, D. Thirde, F. Fusier, V. Valentin, F. Brémond and M. Thonnat and J. Aguilera and M. Kampel, "Automated Scene Understanding for Airport Aprons". The *Proceedings of 18th Australian Joint Conference on Artificial Intelligence*, Sidney, Australia, 5-9 December 2005.

[47] Toshev, F. Brémond and M. Thonnat, "An A priori-based Method for Frequent Composite Event Discovery in Videos". The *Proceedings of 2006 IEEE International Conference on Computer Vision Systems*, New York, USA, January 5-7, 2006.

[48] B. Georis, F. Brémond and M. Thonnat, "Evaluation and Knowledge Representation Formalisms to Improve Video Understanding". The *Proceedings of 2006 IEEE International Conference on Computer Vision Systems*, New York, USA, January 5-7, 2006.

[49] M. Borg, D. Thirde, J. Ferryman, F. Fusier, V. Valentin, F. Brémond and M. Thonnat, "A Real-Time Scene Understanding System for Airport Apron Monitoring". The *Proceedings of 2006 IEEE International Conference on Computer Vision Systems*, New York, USA, January 5-7, 2006.

[50] MB.Kaâniche, F. Brémond and M. Thonnat, « Monitoring Trichogramma Activities from Videos : An adaptation of cognitive vision system to biology field ». The *International Cognitive Vision Workshop*, (ICVW'2006) in conjunction with the 9th European Conference on Cognitive Vision (ECCV'2006), Graz, Austria, May 07-13, 2006.

[51] T. Vu, F. Brémond, G. Davini, M. Thonnat, Q.C. Pham, N. Allezard, P. Sayd, J.L. Rouas, S. Ambellouis and A. Flancquart, Audio Video Event Recognition System for Public Transport Securityn. The IET conference on *Imaging for Crime Detection and Prevention* (ICDP 2006), London, Great Britain, June 13-14, 2006.

[52] C. Carincotte, X. Desurmont, B. Ravera, F. Bremond, J. Orwell, S.A. Velastin, J.M. Odobez, B. Corbucci, J. Palo, J. Cernocky, Toward generic intelligent knowledge extraction from video and audio: the EU-funded CARETAKER project. The IET conference on *Imaging for Crime Detection and Prevention* (ICDP 2006), London, Great Britain, June 13-14, 2006.

[53] M. Zúñiga, F. Brémond and M. Thonnat. Fast and reliable object classification in video based on a 3D generic model. The *3rd International Conference on Visual Information Engineering* (VIE 2006), pp.433-440, Bangalore, India, September 26-28, 2006.

[54] B. Zhan, P. Remagnino, S. Velastin, F. Brémond and M. Thonnat, Matching gradient descriptors with topological constraints to characterise the crowd dynamics. The *3rd International Conference on Visual Information Engineering* (VIE 2006), pp.441-440, Bangalore, India, September 26-28, 2006.

[55] Nadia Zouba, François Bremond, Monique Thonnat and Van Thinh Vu. Multi-sensors Analysis for Everyday Elderly Activity Monitoring. In the *SETIT 2007, 4th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications* - Tunis, TUNISIA, March 25-29, 2007.

Reports and national conferences with a reviewing committee

[56] F. Brémond. Les attributs multivalués dans les systèmes de représentation de connaissances à objets; l'exemple du raisonnement spatial. *Master's thesis, Ecole Normale Supérieure de Lyon,* September 1992.

[57] F. Brémond et M. Thonnat. Interprétation de séquences d'images et incertitude. Dans *proc. sur la Logique Floue et ses Applications* (LFA), Nancy, December 1996.

[58] M. Thonnat, A. Avanzi and F. Brémond. Analyse de scènes de bureaux pour la communication video. Dans *proc. de Traitement et Analyse, d'Images Méthodes et Applications* (TAIMA'01), oct 2001, Hammamet (Tunisie).

[59] F. Brémond, N. Maillot, M. Thonnat and T. Vu, Ontologies For Video Events. *INRIA Research Report RR-5189,*, May 2004.

References

[Advisor, 2003] Advisor (2003). http://wwwsop.inria.fr/orion/ADVISOR/index.html.

[Agarwal and Triggs, 2006] A. Agarwal and B. Triggs. A local basis representation for estimating human pose from cluttered images. In Asian Conference on Computer Vision, January 2006.

[Al-Hames et al., 2005] M. Al-Hames, A. Dielmann, D. Gatica-Perez, S. Reiter, S. Renals, G. Rigoll, and D. Zhang. Multimodal integration for meeting group action segmentation and recognition. In MLMI, 2005.

[Allen, 1981] J. F. Allen. An interval-based representation of temporal knowledge. In Proceedings of IJCAI, 1981.

[Andrade et al., 2006a] E. Andrade, S. Blunsden, and R. Fisher. Performance analysis of event detection models in crowded scenes. In Workshop on Towards Robust Visual Surveillance Techniques and Systems at Visual Information Engineering 2006, pages 427–432, Bangalore, India, September 2006.

[Andrade et al., 2006b] E. L. Andrade, S. Blunsden, and R. B. Fisher. Hidden markov models for optical flow analysis in crowds. In International Conference on Pattern Recognition, Hong Kong, August 2006.

[Andrade et al., 2006c] E. L. Andrade, S. Blunsden, and R. B. Fisher.Modelling crowd scenes for event detection. In International Conference on Pattern Recognition, Hong Kong, August 2006.

[Avitrack, 2006] Avitrack (2006). http://www.avitrack.net.

[Ayers and Shah, 2001] D. Ayers and M. Shah. Monitoring human behavior from video taken in an office environment. Image and Vision Computing, 19:833–846, October 2001.

[Barnard et al., 2003] M. Barnard, J. M. Odobez, and S. Bengio. Multi-modal audio-visual event recognition for football analysis. In IEEE Workshop on Neural Networks for Signal Processing (NNSP), Toulouse, France, September 2003.

[Barron et al., 1994] Barron, J., Fleet, D., and Beauchemin, S. (1994). Performance of optical flow techniques. International Journal of Computer Vision, 12(1):42–77.

[Bashir and Porikli, 2006] F. Bashir and F. Porikli. Performance evaluation of object

detection and tracking systems. In PETS, 2006.

[Baumberg and Hogg, 1994] Baumberg, A. and Hogg, D. (1994). An efficient method for contour tracking using active shape models. In Proceedings of the IEEE Workshop on Motion of Non-Rigid and Articulated Objects, pages 194–199, Austin, USA.

[Berclaz et al., 2006] J. Berclaz, F. Fleuret, and P. Fua. Robust people tracking with global trajectory optimization. In Conference on Computer Vision and Pattern Recognition, 2006.

[Bergen et al., 1992] Bergen, J., Burt, P., Hingorani, R., and Peleg, S. (1992). A three frame algorithm for estimating two-component image motion. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 14(9):886–896.

[Beucher and Bilodeau, 1994] Beucher, S. and Bilodeau, M. (1994). Road segmentation and obstacle detection by a fast watershed transformation. In Intelligent Vehicle Symposium ’94, pages 296–301, Paris, France.

[Beucher and Lantuejoul, 1979] Beucher, S. and Lantuejoul, C. (1979). Use of watersheds in contour detection. In Proceedings of the International Workshop on Image Processing, Real-Time Edge and Motion Detection/ Estimation, Rennes, France.

[Bezault et al, 1992] Laurent Bezault, Ronan Boulic, Nadia Magnenat-Thalmann, Daniel Thalmann: An interative Tool for the Design of Human Free-Walking trajectories, *Computer Animation ’92*, pp.87-104.

[Black and Ellis, 2001] Black, J. and Ellis, T. (2001). Multi camera image tracking. In Ferryman, J., editor, Proceedings of the 2nd IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS’01), Kauai, Hawaii.

[Borg et al., 2005] Borg, D. J. Thirde, J. M. Ferryman, K. D. Baker, J. Aguilera, and M. Kampel. Evaluation of object tracking for aircraft activity surveillance. In The Second Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS 2005), Beijing, China, October 2005.

[Brand, 1999] Brand, M. (1999). Pattern discovery via entropy minimization. In Deckerman, D. and Whittaker, J., editors, Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics, Fort Lauderdale, Florida, USA. Morgan Kaufmann Publishers Inc.

[Brand and Kettnaker, 2000] Matthew Brand and Vera Kettnaker: Discovery and Segmentation of Activities in Video. IEEE Trans. on Pattern Analysis and Machine Intelligence, 22:8 (2000), pp. 844 - 851.

[Brand et al., 1997] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In CVPR, 1997.

[Brdiczka et al., 2006] O. Brdiczka, J. Maisonnasse, P. Reignier, and J. L. Crowley.Extracting activities from multimodal observation. In KES2006 10th

International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Bournemouth, UK, October 2006.

[Buxton, 2002] H. Buxton. Learning and understanding dynamic scene activity. In ECCV Generative Model Based Vision Workshop, Copenhagen, 2002.

[Buxton and Gong, 1995] Buxton, H. and Gong, S. (1995). Visual surveillance in a dynamic and uncertain world. Artificial Intelligence, 78(1- 2):431–459.

[Castel et al., 1996] C. Castel, L. Chaudron, and C. Tessier. 1st order ccubes for the interpretation of petri nets: an application to dynamic scene understanding. In Proceeding TAI'96, Toulouse, France, 1996.

[Cavallaro, 2005] A. Cavallaro. Event detection in underground stations using multiple heterogeneous surveillance cameras. In International Symposium on Visual Computing, Intelligent Vehicles and Autonomous Navigation, Lake Tahoe, Nevada, December 2005.

[Cavallaro et al., 2005a] A. Cavallaro, E. Salvador, and T. Ebrahimi. Shadow-aware object-based video processing. IEE Vision, Image and Signal Processing, 152:14– 22, August 2005.

[Cavallaro et al., 2005b] A. Cavallaro, O. Steiger, and T. Ebrahimi. Tracking video objects in cluttered background. IEEE Transactions on Circuits and Systems for Video Technology, 15(4):575– 584, April 2005.

[Chleq and Thonnat, 1996] Nicolas Chelq and Monique Thonnat, Real-time image sequence interpretation for video-surveillance applications. International conference on Image Processing (ICIP'96). Proceeding IEEE ICIP'96. Vol 2. pp 801-804. Lausanne, Switzerland. September 1996.

[Coimbra et al., 2003] Coimbra, M., Davies, M., and Velastin, S. (2003). Pedestrian detection using MPEG2 motion vectors. In Izquierdo, E., editor, Proceedings of the International Workshop for Image Analysis for Multimedia Interactive Services (WIAMIS'03), pages 164–169, London, United Kingdom.

[Cohn et al., 2006] A. G. Cohn, D. C. Hogg, B. Bennett, V. Devin, A. Galata, D. R. Magee, C. Needham, and P. Santos. Cognitive vision: Integrating symbolic qualitative representations with computer vision. Cognitive Vision Systems: Sampling the Spectrum of Approaches, Lecture Notes in Computer Science, pages 221–246, 2006.

[Collins and Kanade, 2001] Robert T. Collins and Takeo Kanade, Robotics Institute Carnegie Mellon University: Multi-camera Tracking and Visualisation for Surveillance and Sports, *Proc. Of the Fourth International Workshop on Cooperative Distributed Vision*, 22-24/03/2001.

[Collins et al., 1999] Collins, R., Lipton, A., and Kanade, T. (1999). A system for video surveillance and monitoring. In Proceedings of the American Nuclear Society 8th International Topical Meeting on Robotics and Remote System, pages 25–29.

[Collins et al., 2000] Collins, R., Lipton, A., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., Burt, P., and Wixson, L. (2000). A system for video surveillance and monitoring: VSAM final report. Technical report, CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University.

[Cootes et al., 1995] Cootes, T., Taylor, C., Cooper, D., and Graham, J. (1995). Active shape models: their training and applications. Computer Vision and Image Understanding, 61(1):38–59.

[Coue et al., 2006] C. Coue, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessiere. Bayesian occupancy filtering for multitarget tracking: an automotive application. Journal of Robotics Research, 25(1):19–30, January 2006.

[Cox and Higorani, 1996] Cox, I. and Higorani, S. (1996). An efficient implementation of reid's mht algorithm and its evaluation for the purpose of visual tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(2):138–150.

[Cristani et al., 2006] M. Cristani, M. Bicego, and V. Murino. Audio-visual event recognition in surveillance video sequences. IEEETransactions on Multimedia, 2006.

[Crowley, 2006a] J. L. Crowley. Cognitive Vision Systems, chapter Things that See: Context-Aware Multi-modal Interaction. Springer Verlag, March 2006.

[Crowley, 2006b] J. L. Crowley. Situation models for observing human activity. ACM Queue Magazine, May 2006.

[Cucchiara et al., 2005] R. Cucchiara, A. Prati, and R. Vezzani. Making the home safer and more secure through visual surveillance. In Symposium on Automatic detection of abnormal human behaviour using video processing of Measuring Behaviour, Wageningen, The Netherlands, 2005.

[Cucchara et al., 2004] R. Cucchiara, C. Grana, A. Prati, G. Tardini, and R. Vezzani. Using computer vision techniques for dangerous situation detection in domotics applications. In IEE Intelligent Distributed Surveillance Systems (IDSS04), pages 1–5, London, UK, February 2004.

[Cucchiara et al., 2003a] R. Cucchiara, A. Prati, and R. Vezzani. Domotics for disability: smart surveillance and smart video server. In 8th Conference of the Italian Association of Artificial Intelligence -Workshop on Ambient Intelligence, pages 46–57, Pisa, Italy, September 2003.

[Cucchiara et al., 2003b] Cucchiara, R., Grana, C., Piccardi, M., and Prati, A. (2003). Detecting moving objects, ghosts and shadows in video streams. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 25(10):1337–1342.

[Cucchiara et al., 2001] R. Cucchiara, C. Grana, G. Neri, M. Piccardi, and A. Prati. The sakbot system for moving object detection and tracking. Video-based Surveillance Systems: Computer Vision and Distributed Processing (Part II -Detection and Tracking), pages 145–158, 2001.

[Dalal, 2006] N. Dalal. Finding People in Images and Videos. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble, France, July 2006.

[Dalal et al., 2006] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In European Conference on Computer Vision, 2006.

[Davis et al., 2005] L. S Davis, D. Harwood, and V. D. Shet. Vidmap: Video monitoring of activity with prolog. In AdvancedVideoand Signal-Based Surveillance(AVSS), Como, Italy, September 2005.

[Davies and Velastin, 2005] A. C. Davies and S. A. Velastin. Progress in computational intelligence to support cctv surveillance systems. International Scientific Journal of Computing, 4(3):76–84, 2005.

[Dee and Hogg, 2005] H. M. Dee and D. C. Hogg. On the feasibility of using a cognitive model to filter surveillance data. In IEEE International Conference on Advanced Video and Signal-Based Surveillance, Como, Italy, 2005.

[Desurmont et al., 2006a] X. Desurmont, J. B. Hayet, J. F. Delaigle, J. Piater, and B. Macq. Trictrac video dataset: Public hdtv synthetic soccer video sequences with ground truth. In Workshop on Computer Vision Based Analysis in Sport Environments (CVBASE), pages 92–100, 2006.

[Desurmont et al., 2006b] X. Desurmont, I. Ponte, J. Meessen, and J. F. Delaigle. Nonintrusive viewpoint tracking for 3d for perception in smart video conference. In Three-Dimensional Image Capture and Applications VI,San Jose, CA USA, January 2006.

[Desurmont et al., 2006c] X. Desurmont, R. Sebbe, F. Martin, C. Machy, and J. F. Delaigle. Performance evaluation of frequent events detection systems. In Ninth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, New York, June 2006.

[Dimitrijevic et al., 2005] M. Dimitrijevic, V. Lepetit, and P. Fua. Human body pose recognition using spatio-temporal templates. In ICCV workshop on Modeling People and Human Interaction, Beijing, China, October 2005.

[Doermann and Mihalcik, 2000] Doermann, D. and Mihalcik, D. (2000). Tools and techniques for video performance evaluation. In Proceedings of the International Conference on Pattern Recognition (ICPR’00), pages 167–170, Barcelona, Spain.

[Dong and Pentland, 2006] W. Dong and A. Pentland. Multi-sensor data fusion using the influence model. In Body Sensor Networks Workshop, Boston, MA, April 2006.

[Dousson and Duong, 1999] Dousson, C. and Duong, T. (1999). Discovering chronicles with numerical time constraints from alarm logs for monitoring dynamic systems. In Dean, T., editor, Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI’99), pages 620–626, Stockholm, Sweden. Morgan Kaufmann Publishers Inc.

[Dousson, 1994] C. Dousson. Suivi d'evolutions et reconnaissance de chroniques. These Universite Paul Sabatier de Toulouse, September 1994.

[Dousson et al., 1993] C. Dousson, P. Gaborit, and M. Ghallab. Situation recognition: Representation and algorithms. In The 13rd IJCAI, pages 166–172, August 1993.

[Dousson and Ghallab, 1994] C. Dousson and M. Ghallab. Suivi et reconnaissance de chroniques. Revue d'intelligence artificielle, Vol.8, N1:29–61, 1994.

[Du et al., 2006] W. Du, J. B. Hayet, J. Piater, and J. Verly. Collaborative multi-camera tracking of athletes in team sports. In Workshop on Computer Vision Based Analysis in Sport Environments (CVBASE), pages 2–13, 2006.

[Du and Piater, 2006] W. Du and J. Piater. Multi-view tracking using sequential belief propagation. In Asian Conference on Computer Vision, pages 684–693, Hyderabad, India, 2006. Springer-Verlag.

[Elgammal et al., 2000] Elgammal, A., Harwood, D., and Davis, L. (2000). Non-parametric background model for background subtraction. In Vernon, D., editor, Proceedings of the 6th European Conference on Computer

[Ellis, 2002] Ellis, T. (2002). Performance metrics and methods for tracking in surveillance. In Ferryman, J., editor, Proceedings of the 3rd IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS'02), pages 26–31, Copenhagen, Denmark.

[Enficiaud et al., 2006] A. Enficiaud, B. Lienard, N. Allezard,R.Sebbe, S.Beucher,X.Desurmont, P. Sayd, and J. F. Delaigle. Clovis -a generic framework for general purpose visual surveillance applications. In IEEE International Workshop on Visual Surveillance (VS), 2006.

[Etiseo, 2005] Etiseo (2005). http://www.etiseo.net.

[Fejes and Davis, 1997] Fejes, S. and Davis, L. (1997). Detection of independent motion using directional motion estimation. Technical report, CS-TR-3815, CAR-TR-866, University of Maryland.

[Ferryman et al., 2000] J. M. Ferryman, S. Maybank, and A. Worrall. Visual surveillance for moving vehicles. International Journal of Computer Vision, 37(2):187– 197, June 2000.

[Foresti et al., 2005] G. L. Foresti, C. Piciarelli, and L. Snidaro. Trajectory clustering and its applications for video surveillance. In Advanced Video and Signal-Based Surveillance (AVSS), Como, Italy, September 2005.

[Foresti, 2003] G. L. Foresti. Advanced neural-based systems for 3d scene undertsanding. Intelligent Systems: Techniques and Applications, pages 237–268, 2003.

[Foresti and Micheloni, 2003] G. L. Foresti and C. Micheloni. A robust feature tracker for active surveillance of outdoors scenes. Electronic Letters on Computer Vision and

Image Analysis, 1(1):21–34, 2003.

[Foresti et al., 2003] G. L. Foresti, C. Micheloni, L. Snidaro and C. Marchiol. Face detection for visual surveillance. In 12th International Conference on Image Analysis and Processing (ICIAP03), pages 115–119, Mantova, Italy, September 2003.

[Forest et al., 2002] G. L. Foresti, G. Giacinto, and F. Roli. Detecting dangerous Behaviors of Mobile Objects in Parking Areas. Kluwer Academic Publishers, 2002.

[Foresti and Regazzoni, 2002] G. L. Foresti and C. S. Regazzoni. Multisensor data fusion for driving autonomous vehicles in risky environments. IEEE Transactions on Vehicular Technology, 51(5):1165 –1185, September 2002.

[Friedman and Russell, 1997] Friedman, N. and Russell, S. (1997). Image segmentation in video sequences: a probabilistic approach. In Proceedings of the 13th Conference of Uncertainty in Artificial Intelligence, pages 175 – 181, San Francisco, USA.

[Galata et al., 2002] Aphrodite Galata, Anthony Cohn, Derek Magee, and David Hogg: Modeling Interaction Using Learnt Qualitative Spatio-Temoral Relations and Variable Length Markov Models. Proc. of the 15th European Conference on Artificial Intelligence, 2002, pp. 741 – 745

[Galata et al., 2001] A. Galata, N. Johnson, and D. Hogg. Learning variable length markov models of behaviour. Computer Vision and Image Understanding, 81(3):398–413, 2001.

[Gatica-Perez, 2005] D. Gatica-Perez. Tracking people in meetings with particles. In Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), 2005.

[Gavrila and Munder, 2007] D. M. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. International Journal of Computer Vision (est: February-March 2007 issue), 2007.

[Gelgon et al., 2005] M. Gelgon, P. Bouthemy, and J. P. Le-Cadre. Recovery of the trajectories of multiple moving objects in an image sequence with a pmht approach. Image and Vision Computing Journal, 23(1):19–31, 2005.

[Georis, 2006] Georis, Benoît (2006). Program Supervision Techniques for Easy Configuration of Video Understanding Systems. PhD. of the Université Catholique de Louvain, Faculté des Sciences Appliquées, Département d'Electricité, Laboratoire de Télécommunication et Télédetection Janvier 2006.

[Gerber et al., 2002] R. Gerber, H. Nagel and H. Schreiber. Deriving Textual Descriptions of Road Traffic Queues from Video Sequences. The 15-th European Conference on Artificial Intelligence (ECAI'2002), Lyon, France, 21-26 July 2002, pp.736-740.

[Ghallab, 1996] Malik Ghallab, On Chronicles: Representation, On-line Recognition and Learning. 5th International Conference on Principles of Knowledge Representation and Reasoning (KR'96), Cambridge (USA), 5-8 November 1996, pp.597-606.

[Gomila and Meyer, 2001] C. Gomila and F. Meyer. Tracking objects by graph matching of image partition sequences. In 3rd IAPR-TC15 Workshop on Graph-Based Representations in Pattern Recognition, pages 1–11, Ischia, Italy, May 2001.

[Gong and Hung, 2005] S. Gong and H. S. Hung. Detecting and quantifying unusual interactions by correlating salient motion. In Advanced Video and Signal-Based Surveillance (AVSS), Como, Italy, September 2005.

[Gong and Xiang, 2005]S. Gong and T. Xiang. Relevance learning for spectral clustering with applications on image segmentation and video behaviour profiling. In Advanced Video and Signal-Based Surveillance (AVSS), Como, Italy, September 2005.

[Gong and Xiang, 2003]S. Gong and T. Xiang. Recognition of group activities using dynamic probabilistic networks. In The 9th International Conference on Computer Vision, Nice, France, October 2003.

[Gourier et al., 2006] N. Gourier, J. Maisonnasse, D. Hall, and J. L. Crowley. Head pose estimation on low resolution images. In CLEAR 2006, South Hampton, UK, April 2006.

[Greenhill et al., 2002] D. Greenhill, P. Remagnino, and G. A. Jones. Video Based Surveillance Systems -Computer Vision and Distributed Processing, chapter VIGILANT: Content-Querying of Video Surveillance Streams, pages 193–204. Kluwer Academic Publishers, 2002.

[Guler and Farrow, 2006] S. Guler and M. K. Farrow. Abandoned object detection in crowded places. In PETS, 2006.

[Hall et al., 2006] D. Hall, R. Emonet, and J. L. Crowley. An automatic approach for parameter selection in self-adaptive tracking. In International Conference on Computer Vision Theory and Applications (VISAPP), Setubal, Portugal, February 2006.

[Hall et al., 2005] D. Hall, J. Nascimento, P. Ribeiro, E. Andrade, P. Moreno, S. Pesnel, T. List, R. Emonet, R. B. Fisher, J. Santos-Victor, and J. L. Crowley. Comparison of target detection algorithms using adaptive background models. In 2nd Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS), pages 113–120, Beijing, October 2005.

[Hall et al., 2004] D. Hall, F. Pelisson, O. Ri□, and J. L. Crowley. Brand identification using gaussian derivative histograms. Machine Vision and Applications, in Machine Vision and Applications, 16(1):41–46, 2004.

[Haritaoglu et al., 2000] Haritaoglu, I., Harwood, D., and Davis, L. (2000). W4: Real-time surveillance of people and their activities. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8):809–830.

[Herbulot et al., 2006] Herbulot A., Jehan-Besson S., Duffner S., Barlaud M., Aubert G., Segmentation of vectorial image features using shape gradients and information

measures, in Journal of Mathematical Imaging and Vision, Volume 25, Issue 3, Pages 365-386, Octobre 2006.

[Howell and Buxton, 2002] A.J. Howell and H. Buxton. Active vision techniques for visually mediated interaction. Image and Vision Computing, 2002.

[Huang et al., 2005a] C. L. Huang, E. L. Chen, and P. C. Chung. Fall detection using modular neural networks and back-projected optical flow. In The 12th International Conference on Neural Information Processing (ICONIP 2005), 2005.

[Huang et al., 2005b] C. L. Huang, E. L. Chen, and P. C. Chung. Using modular neural networks for falling detection. In The 10th Conference on Artificial Intelligence and Application (TAAI2005), 2005.

[Hung et al., 2002] Y. P. Hung, Y. P. Tsai, and C. C. Lai. A bayesian approach to video object segmentation via merging 3d watershed volumes. In Proceedings of International Conference on Pattern Recognition (ICPR02), Quebec, Canada, August 2002.

[Intille and Bobick, 2001] S. Intille and A. Bobick. Recognizing planned, multi-person action. Computer Vision andImageUnderstanding, 81(3), 2001.

[Ivanov and Wren, 2006] Y. Ivanov and C. R. Wren. Toward spatial queries for spatial surveillance tasks. Technical Report TR2006-051, Mitsubishi Electric Research Laboratories, May 2006.

[Ivanov and Bobick, 2000] Y. Ivanov and A. Bobick. Recognition of visual activities and interactions by stochastic parsing. IEEETrans.Patt.Anal.Mach. Intell., 22(8):852–872, 2000.

[Javed et al., 2003] Javed, O., Rasheed, Z., Shafique, K., and Shah, M. (2003). Tracking across multiple cameras with disjoint views. In Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV'03), pages 952–957, Nice, France.

[Jodogne and Piater, 2005] S. Jodogne and J. Piater. Learning, then compacting visual policies. In 7th European Workshop on Reinforcement Learning, Naples, Italy, 2005.

[Johnson and Hogg, 1996] Johnson, N. and Hogg, D. (1996). Learning the distribution of object trajectories for event recognition. Image and Vision Computing, 14(8):609–615.

[Kaewtrakulpong and Bowden, 2001] Kaewtrakulpong, P. and Bowden, R. (2001). An improved adaptive background mixture model for real-time tracking with shadow detection. In Proceedings of the 2nd European Workshop on Advanced Video Based Surveillance Systems (AVBSS'01), Kingston, United Kingdom.

[Kale et al., 2005] A. Kale, N. Cuntoor, B. Yegnanarayana, A. N. Rajagoplan, and R. Chellappa. Using Appearance Matching in Optical and Digital Techniques of Information Security, chapter Gait-based Human Identification. Springer, 2005.

[Kautz and allen, 1986] H. A. Kautz and J. F. Allen. Generalized plan recognition. In Proceeding of the5th AAAI, pages 32–37, Philadelphia, Pennsylvania, 1986.

[Khalaf and Intille, 2001] Khalaf, R. and Intille, S. (2001). Improving multiple tracking using temporal consistency. Technical report, HouseN: the MIT Home of the Future, Massachusetts Institute of Technology.

[Kowalski and Sergot, 1986] R. Kowalski and M. Sergot. A logic-based calculus of events. NewGeneration Computing, 4:67–95, 1986.

[Krahnstoever et al., 2006] N. Krahnstoever, P. Tu, T. Sebastian, A. Perera, and R. Collins. Multi-view detection and tracking of travelers and luggage in mass transit environments. In PETS, 2006.

[Kumar and Mukerjee, 1987] K. Kumar and A. Mukerjee. Temporal event conceptualization. In The 10th IJCAI, pages 472–475, 1987.

[Kuno et al., 1996] Kuno, Y., Watanabe, T., Shimosakoda, Y., and Nakagawa, S. (1996). Automated detection of human for visual surveillance systems. In Proceedings of the International Conference on Pattern Recognition (ICPR'96), pages 865–869.

[Laptev, 2006] I. Laptev. Improvements of object detection using boosted histograms. In BMVC, number 3, pages 949–958, Edinburgh, UK, 2006.

[Laptev et al., 2007] I. Laptev, B. Caputo, and T. Lindeberg. Local velocity-adapted motion events for spatio-temporal recognition. Computer Vision and Image Understanding, 2007.

[Larlus and Jurie, 2006] D. Larlus and F. Jurie. Latent mixture vocabularies for object categorization. In British Machine Vision Conference, 2006.

[Lazarevic et al., 2006] N. Lazarevic-McManus, J. R. Renno, D. Makris, and G. A. Jones. Designing evaluation methodologies: the case of motion detection. In Ninth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, New York, June 2006.

[Lesire and Tessier, 2005] C. Lesire and C. Tessier. Particle petri nets for aircraft procedure monitoring under uncertainty. In 26thInternational Conference onApplicationand Theory of Petri Nets and Other Models of Concurrency (ATPN), Florida, USA, June 2005.

[Li et al., 2006] L. Li, R. Luo, R. Ma, W. Huang, and K. Leman. Evaluation of an ivs system for abandoned object detection on pets 2006 datasets. In PETS, 2006.

[Lim et al., 2005] S. N. Lim, A. Mittal, L. S. Davis, and N. Paragios. Fast illumination-invariant background subtraction using two views: Error analysis, sensor placement and applications. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Diego, CA, June 2005.

[Lipton et al., 1998] Lipton, A., Fujiyoshi, H., and Patil, R. (1998). Moving target classification from real-time video. In Proceedings of the IEEE Workshop on Application of Computer Vision, pages 8–14, Princeton, New Jersey, USA.

[Lv et al., 2006] F. Lv, X. Song, B. Wu, V. K. Singh, and R. Nevatia. Left luggage detection using bayesian inference. In PETS, 2006.

[Magee et al., 2004] D. R. Magee, C. J. Needham, P. Santos, A. G. Cohn, D. C. Hogg: Autonomous learning for a cognitive agent using continuous models and inductive logic programming from audio-visual input. Anchoring Symbols to Sensor Data, 2004 AAAI Workshop, 2004, pp. 17-24.

[Manohar et al., 2006] V. Manohar, M. Boonstra, V. Korzhova, P. Soundararajan, D. Goldgof, R. Kasturi, S. Prasad, H. Raju, R. Bowers, and J. Garofolo. Pets vs. vace evaluation programs: A comparative study. In PETS, 2006.

[Marcon et al., 2005] M. Marcon, M. Pierobon, A. Sarti, and S. Tubaro.Clustering of human actions using invariant body shape descriptor and dynamic time warping. In Advanced Video and Signal-Based Surveillance (AVSS), Como, Italy, September 2005.

[Mariano et al., 2002] Mariano, V., Min, J., Park, J.-H., Kasturi, R., Mihalcik, D., Doermann, D., and Drayer, T. (2002). Performance evaluation of object detection algorithms. In Proceedings of the International Conference on Pattern Recognition (ICPR'02), pages 965–969, Québec, Canada.

[Mark and Gavrila, 2006] W. Mark and D. M. Gavrila. Real-time dense stereo for intelligent vehicles. IEEE Transactions on Intelligent Transportation Systems, 7(1):38– 50, March 2006.

[Martinez del Rincon et al., 2006] J. Martinez del Rincon, J. E. Herrero-Jaraba, J. R. Gomez, and C. Orrite-Urunuela. Automatic left luggage detection and tracking using multi-camera ukf. In PETS, 2006.

[Mathes and Piater, 2006] T. Mathes and J. Piater. Robust non-rigid object tracking using point distribution manifolds. In 28th Annual Symposium of the German Association for Pattern Recognition (DAGM), 2006.

[McKenna et al., 2000] McKenna, S., Jabri, S., Duric, Z., Rosenfeld, A., and Wechsler, H. (2000). Tracking groups of people. Computer Vision and Image Understanding, 80(1):42–56.

[Micheloni and Foresti, 2003] C. Micheloni and G. L. Foresti. Fast good feature selection for wide area monitoring. In IEEE Conference on Advanced Video and Signal Based Surveillance, pages 271–276, Miami Beach, Florida, July 2003.

[Middendorf and Nagel, 2000] Middendorf, M. and Nagel, H.-H. (2000). Vehicle tracking using adaptive optical flow estimation. In Ferryman, J., editor, Proceedings of the 1st IEEE Workshop on Performance Evaluation of Tracking and Surveillance (PETS'00), pages 38–45, Grenoble, France.

[Mittal and Davis, 2003] Mittal, A. and Davis, L. (2003). M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene. International Journal of Computer Vision, 51(3):189–203.

[Mittal et al., 2003] A. Mittal, L. Zhao, and L. Davis. Human body pose estimation by shape analysis of silhouettes. In IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Orlando, Florida, July 2003.

[Moeslund et al., 2006] T. Moeslund, A. Hilton, V. Krüger. A survey of advances in vision-based human motion capture and analysis. Computer Vision and Image Understanding, Volume 104, Issue 2-3, Pages 90-126, 2006.

[Mohr and Henderson, 1986] R. Mohr and T. C. Henderson. Arc and path consistency revisited. Research Note,Artificial Intelligence, 28:225–233, 1986.

[Moisan and Thonnat, 2000] Moisan, S. and Thonnat, M. (2000). What can program supervision do for program reuse. IEE Proceedings - Software, 147(5):179–185.

[Monnet et al., 2003] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background modeling and subtraction of dynamic scenes. In IEEE International Conference on Computer Vision (ICCV), Nice, France, October 2003.

[Munder and Gavrila, 2006] S. Munder and D. M. Gavrila. An experimental study on pedestrian classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(11):1863–1868, 2006.

[Nagel, 2000] Nagel, H.-H. (2000). Image sequence evaluation: 30 years and still going strong. In Sanfeliu, A. and et al., J. V., editors, Invited Lecture in Proceedings of the 15th International Conference on Pattern Recognition (ICPR2000), Barcelona, Spain, volume 1, pages 149–158.

[Needham et al., 2005] C. J. Needham, P. E. Santos, D. R. Magee, V. Devin, D. C. Hogg, and A. G. Cohn. Protocols from perceptual observations. Artificial Intelligence, 167(1-2):103–136, 2005.

[Nevatia et al., 2004] R. Nevatia, J. Hobbs, and B. Bolles. An ontology for video event representation. In IEEE Workshop on Event Detection and Recognition, June 2004.

[Nixon et al., 2005] M. S. Nixon, T. N. Tan, and R. Chellappa. Human Identification Based on Gait. Springer, December 2005.

[Nokel, 1989] K. Nokel. Temporal matching: Recognizing dynamic situations from discrete measurements. In The 11st IJCAI, pages 1255–1260, Detroit, Michigan, USA, 1989.

[Noor et al., 2006] H. Noor, S. H. Mirza, Y. Sheikh, A. Jain, and M. Shah. Model generation for video based object recognition. In ACM MM, Santa Barbara, CA, USA, 2006.

[Numerical, 2002] Numerical Recipes Example Book (C++)- The Art of Scientific Computing - de William T. Vetterling, William H. Press, Saul A. Teukolsky, Brian P. Flannery - Mathematics - 2002 - 330 pages, 28 2002.

[Nummiaro et al., 2002] Nummiaro, K., Koller-Meier, E., and Gool, L. V. (2002). A color-based particle filter. In Pece, A., editor, Proceedings of the 1st International Workshop on Generative Model-Based Vision (GMBV'02), pages 53–60, Copenhagen, Denmark.

[Ogata et al., 2006] T. Ogata, W. Christmas, J. Kittler, and S. Ishikawa. Improving human activity detection by combining multi-dimensional motion descriptors with boosting. In Proceedings of the 18th International Conference on Pattern Recognition, volume 1, pages 295–298. IEEE Computer Society, August 2006.

[Olguin and Pentland, 2006] D. O Olguin and A. S. Pentland. Human activity recognition: Accuracy across common locations for wearable sensors. In 10th International Symposium on Wearable Computing (Student Colloquium), Montreux, Switzerland, October 2006.

[Oliver et al., 2000] N. M. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modeling human interactions. IEEETransactions onPatternAnalysis and MachineIntelligence, 22(8):831–843, 2000.

[Oren et al., 1997] Oren, M., Papageorgiou, C., Sinha, P., Osuna, E., and Poggio, T. (1997). Pedestrian detection using wavelet templates. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR'97), pages 193–199, Puerto Rico, USA. IEEE Computer Society Press.

[Ozuysal et al., 2006] M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua. Feature harvesting for tracking-by-detection. In European Conference on Computer Vision, pages 592–605, 2006.

[Panini and Cucchiara, 2003] L. Panini and R. Cucchiara. A machine learning approach for human posture detection in domotics applications. In Proceedings of International Conference on Image Analysis and Processing (ICIAP 2003), pages 103–108, Mantova, Italy, September 2003.

[Parameswaren and Chellappa, 2005] V. Parameswaren and R. Chellappa. Human action-recognition using mutual invariants. Computer Vision and Image Understanding, 98:295–325, September 2005.

[Piater and Crowley, 2001] Piater, J. and Crowley, J. (2001). Multi-modal tracking of interacting targets using gaussian approximations. In Ferryman, J., editor, Proceedings of the 2nd IEEE Workshop on Performance Evaluation of Tracking and Surveillance (PETS'01), Kauai, Hawaii.

[Pinhanez and Bobick, 1997] Claudio Pinhanez et Aaron Bobick. Human Action Detection Using PNF Propagation of Temporal Constraints. M.T.T Media Laboratory Perceptual Section Technical Report No. 423, 04/1997.

[Plankers and Fua, 2001] R. Plankers and P. Fua. Tracking and modeling people in video sequences. Computer Vision and Image Understanding, 81(3):285–302, March 2001.

[Rao et al., 2002] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. International Journal of Computer Vision, 50(2):203–226, 2002.

[Remagnino et al., 2006] P. Remagnino, N. D. Monekosso, and S. A. Velastin. Ambient intelligence. Journal of Japan Society for Fuzzy Theory, February 2006.

[Remagnino et al., 2004] P. Remagnino, A. I. Shihab, and G. A. Jones. Distributed intelligence for multi-camera visual surveillance. Pattern Recognition, Special Issue on Agent-based Computer Vision, 37(4):675–689, April 2004.

[Renno et al., 2003] J. R. Renno, P. Remagnino, and G. A. Jones. Learning surveillance tracking models for the self-calibrated ground plane. Acta Automatica Sinica, Special Issue on Visual Surveillance of Dynamic Scene, 29(3):381–392, 2003.

[Rota, 2001] N. Rota. Contribution a la reconnaissance de comportements humains a partir de sequences videos.These, INRIA-Universite de Nice Sophia Antipolis, October 2001.

[Rota and Thonnat, 2000] Rota, N. and Thonnat, M. (2000). Activity recognition from video sequences using declarative models. In Proceedings of the 14th European Conference on Artificial Intelligence (ECAI'00), pages 673–680, Berlin, Germany.

[Saga and Buxton, 2004] K. Sage and H. Buxton. Joint spatial and temporal structure learning for task based control. In International Conference on Pattern Recognition, 2004.

[Salvador et al., 2004] E. Salvador, A. Cavallaro, and T. Ebrahimi. Cast shadow segmentation using invariant colour features. Computer Vision and Image Understanding, 95(2):238–259, August 2004.

[Scalzo and Piater, 2006] F. Scalzo and J. Piater. Unsupervised learning of dense hierarchical appearance representations. In International Conference on Pattern Recognition, 2006.

[Shah, 2003] M. Shah. Understanding human behavior from motion imagery. Machine Vision and Applications Journal, 14:210–214, September 2003.

[Shanahan, 1990] M. P. Shanahan. Representing continuous change in the event calculus. In Proceedings of ECAI, pages 598–603, 1990.

[Sheikh and Shah, 2005] Y. Sheikh and M. Shah. Bayesian modelling of dynamic scenes for object detection. IEEE Transactions on PAMI, October 2005.

[Shi and Tomasi, 1994] Shi, J. and Tomasi, C. (1994). Good features to track. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR'94), pages 593–600, Seattle, USA. IEEE Computer Society Press.

[Siebel and Maybank, 2002] Siebel, N. and Maybank, S. (2002). Fusion of multiple tracking algorithms for robust people tracking. In Heyden, A., Span, G., Nielsen, M., and Johansen, P., editors, Proceedings of the 7th European Conference on Computer Vision (ECCV'02), Lecture Notes in Computer Science, pages 373–387, Copenhagen, Denmark. Springer.

[Smith et al., 2005] P. Smith, M. Shah, and N. V. Lobo. Integrating multiple levels of zoom to enable activity recognition. Computer Vision and Image Understanding, 2005.

[Snidaro et al., 2003] L. Snidaro, R. Niu, P. K. Varshney, and G. L. Foresti. Automatic camera selection and fusion for outdoor surveillance under changing weather conditions. In IEEE Conference on Advanced Video and Signal Based Surveillance, pages 364–369, Miami Beach, Florida, July 2003.

[Sun et al., 2003] Xingzhi Sun, Maria E. Orlowska, and Xue Li: Introducing Uncertainty into Pattern Discovery in Temporal Event Sequences. Proc. of the 3th IEEE International Conference on Data Mining, 2003, pp. 299 - 306.

[Taj et al., 2006] A. M. Taj, E. Maggio, and A. Cavallaro. Multi-feature graph-based object tracking. In Classification of Events, Activities and Relationships (CLEAR) Workshop, Southampton, UK, April 2006. Springer.

[Terzopoulos, 1999] D. Terzopoulos: Artificial life for computer graphics, *Communications of the ACM*, 42(8), August, 1999, 32-42.

[Tessier, 2003] C. Tessier Towards a commonsense estimator for activity tracking. In AAAI Spring symposium, number 03, Stanford University, USA, 2003.

[Thirde et al., 2006] D. Thirde, L. Li, and J. Ferryman. An overview of the pets 2006 dataset. In PETS, 2006.

[Urtasun et al., 2006] R. Urtasun, D. Fleet, and P. Fua. 3d people tracking with gaussian process dynamical models. In Conference on Computer Vision and Pattern Recognition, June 2006.

[Vaswani et al., 2003] N. Vaswani, A. K. Agrawal, Q. Zheng, and R. Chellappa. Computer Vision beyond the Visible Spectrum, chapter Moving Object Detection and Compression in IR sequences. Springer, 2003.

[Vaswani et al., 2005] N. Vaswani, A. K. Roy-Chowdhury, and R. Chellappa. Shape activity: A continuous-state hmm for moving/deforming shapes with application to abnormal activity detection. IEEE Transactions on Image Processing, 14:1603 – 1616, October 2005.

[Veerarghavan et al., 2005] A. Veeraraghavan, A. K. RoyChowdhury, and R. Chellappa. Matching shape sequences in video with applications in human movement analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27, December 2005.

[Velastin et al., 2005] S. A.Velastin, B. A. Boghossian, B. Lo, J. Sun, and M. A. Vicencio-Silva. Prismatica: Toward ambient intelligence in public transport environments. IEEE

Transactions on Systems, Man, and Cybernetics -Part A, 35(1):164–182, January 2005.

[Velastin et al., 2006] S. A. Velastin, B. A. Boghossian, and M. A. Vicencio-Silva.A motion-based image processing system for detecting potentially dangerous situations in underground railway stations. Transportation Research Part C: Emerging Technologies, 14(2):96–113, 2006.

[Velastin and Remagnino, 2005] S. A. Velastin and P. Remagnino. Intelligent Distributed Surveillance Systems. The Institution of Electrical Engineers (IEE), 2005.

[Viola and Jones, 2004] P. Viola and M. J. Jones. Robust real-time face detection. International Journal of Computer Vision, 57(2):137–154, 2004.

[Viper, 2002] http://lamp.cfar.umd.edu/media/research/viper/.

[VSAM, 1997] http://www.cs.cmu.edu/~vsam/vsamhome.html

[Vu, 2004] Thinh Van VU, Scénarios temporels pour l'interprétation automatique de séquences vidéos (Temporal Scenario for Automatic Video Interpretation). PhD., Université de Nice Sophia Antipolis, UFR Science Ecole Doctorale STIC, Département d'Informatique, Octobre 2004

[Walter et al., 2001] Michael Walter, Alexandra Psarrou, and Shaogang Gong: Data Driven Model Acquisition using Minimum Description Length. Proc. of the British Machine Vision Conference, 2001, pp. 673 - 683.

[Wang et al., 2003] Wang, L., Hu, W., and Tan, T. (2003). Recent developments in human motion analysis. Pattern Recognition, 36(3):585–601, 2003.

[Wilson and Bobick, 2001] A. Wilson and A. Bobick. Hidden markov models for modeling and recognizing gesture under variation. Int'l J. of Pattern Recognition and Artificial Intelligence, 15(1):123–160, 2001.

[Wren et al., 1997] Wren, C., Azarbayejani, A., Darell, T., and Pentland, A. (1997). Pfinder: Real-time tracking of the human body. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(7):780–785.

[Xiang and Gong, 2006a] T. Xiang and S. Gong. Beyond tracking: Modelling activity and understanding behaviour. International Journal of Computer Vision, 67(1):21– 51, 2006.

[Xiang and Gongb, 2006] T. Xiang and S. Gong. Incremental visual behaviour modelling. In IEEE Visual Surveillance Workshop, pages 65–72, Graz, May 2006.

[Zaidenberg et al., 2006] S. Zaidenberg, O. Brdiczka, P. Reignier, and J. L. Crowley. Learning context models for the recognition of scenarios. In In 3rd IFIP Conference on Artificial Intelligence Applications and Innovations (AIAI) 2006,Athens, Greece, June 2006.

[Zhai and Shah, 2006] Y. Zhai and M. Shah. Visual attention detection in video sequences using spatiotemporal cues. In ACM MM, Santa Barbara, CA, USA, 2006.

[Zhang et al., 2005] D. Zhang, D. Gatica-Perez, S. Bengio, and I. McCowan. Semi-supervised adapted hmms for unusual event detection. In EEE CVPR, 2005.

[Zhang and Gong, 2006] J. Zhang and S. Gong. Beyond static detectors: A bayesian approach to fusing long-term motion with appearance for robust people detection in highly cluttered scenes. In IEEE Visual Surveillance Workshop, pages 121–128, Graz, May 2006.

[Zhao and Nevatia, 2004] T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. Computer Vision and Pattern Recognition, 2:406–413, 2004.

[Zhou and Chellappa, 2005] S. Zhou and R. Chellappa. Handbook of Image and Video Processing, chapter Face Recognition from Still Images and Videos. Academic Press, 2005.

[Ziliani and Cavallaro, 2001] F. Ziliani and A. Cavallaro. Image analysis for video surveillance based on spatial regularization of a statistical change detection. Real-Time Imaging, Special Issue on: Video Processing and Communications in Real-Time Video-Based Surveillance Systems, 7(5):389–399, October 2001.

[Ziliani et al., 2005] F. Ziliani, S. Velastin, F. Porikli, L. Marcenaro, T. Kelliher, A. Cavallaro, and P. Bruneaut. Performance evaluation of event detection solutions: the creds experience. In IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS 2005), Como, Italy, September 2005.