

# A generic flexible and robust approach for intelligent real-time video-surveillance systems

Xavier Desurmont\*<sup>a</sup>, Jean-François Delaigle\*<sup>a</sup>, Arnaud Bastide\*<sup>a</sup>, Benoit Macq<sup>b</sup>

<sup>a</sup> Multitel A.S.B.L, Av Copernic, 1, B-7000 Mons, Belgium.

<sup>b</sup> Communications and Remote Sensing Laboratory,  
Université catholique de Louvain, Louvain-la-neuve, Belgium

## ABSTRACT

In this article we present a generic, flexible and robust approach for an intelligent real-time video-surveillance system. A previous version of the system was presented in [1]. The goal of these advanced tools is to provide help to operators by detecting events of interest in visual scenes, highlighting alarms and computing statistics. The proposed system is a multi-camera platform able to handle different standards of video inputs (composite, IP, IEEE1394 ) and which can basically compress (MPEG4), store and display them. This platform also integrates advanced video analysis tools, such as motion detection, segmentation, tracking and interpretation. The design of the architecture is optimized to playback, display, and process video flows in an efficient way for video-surveillance applications. The implementation is distributed on a scalable computer cluster based on Linux and IP network. It relies on POSIX threads for multitasking scheduling. Data flows are transmitted between the different modules using multicast technology and under control of a TCP-based command network (e.g. for bandwidth occupation control). We report here some results and we show the potential use of such a flexible system in third generation video surveillance systems. We illustrate the interest of the system in a real case study, which is the indoor surveillance.

**Keywords:** real-time, intelligent visual surveillance, computer vision, distributed architecture.

## 1. INTRODUCTION

Video security is becoming more and more important today, as the number of installed cameras can attest. Nevertheless, there is still a need for complete and generic systems that can be inserted in an existing camera network (analogic or numeric, e.g. CCTV) and still handle the various ways of video transmission (Firewire, IP, BT,...). Examples of challenging applications [2] are monitoring metro stations [3] or detecting highways traffic jam, intelligent content access, detection of loitering. The requirements for these systems are to be network-connected, multi-cameras, modular, the display must be user-friendly, the vision modules should be plug-and-play and the overall system must be highly reliable and robust.

The work reported here has both research [4][5] and industrial motivations. In this article we present a generic, flexible and robust approach for an intelligent real-time video-surveillance system. Our goals are first to obtain an efficient system that can meet strong industrial requirements and second to have a system that allows researchers to develop new vision algorithms. Such a system must for example include evaluation facilities, such as in [4]. A computer cluster based approach with fast Ethernet network connections is the innovative solution proposed to address the increasing needs of computing power at an affordable cost. The main advantages of this approach are its flexibility and its robustness. Many systems usually require big pipe and MJPEG or MJPEG2000 over ATM techniques to design the architecture of huge monitoring systems. We show here how the MPEG4 standard and a classical IP architecture allow the system to monitor a high number of cameras while ensuring small transmission delays.

The paper is organized as follows: section 2 describes the global system and its main characteristics, section 3 goes deeper in the understanding of each underlying module, section 4 is devoted to the image analysis module, section 5 illustrates the interest of the system in a real case study, which is the “people counting for indoor surveillance”. Section 6 concludes and indicates future work.

---

<sup>1</sup> {desurmont, delaigle, bastide} @multitel.be

## 2. SYSTEM OVERVIEW

The major components of the physical architecture are presented in fig. 1. Basically, the system is composed of computers connected together through a typical fast Ethernet network (100 Mb/s). The various cameras are plugged either on an acquisition card on a PC or directly on the local network hub for IP cameras. A human computer interface and a storage space are also plugged on this system. The main advantage of such an architecture is its flexibility. Future needs in computing power will be simply addressed by adding a PC in the cluster. A new camera can be plugged and configured easily. Fig. 2 shows a screen shot of the graphical user interface (GUI). The system is currently running nine cameras, some of them are indoor scenes and other ones are outdoor scenes. The right part of the picture shows the virtual VCR commands. From an end-user point of view, the GUI is one of the most important modules. It must be very easy to use but efficient.

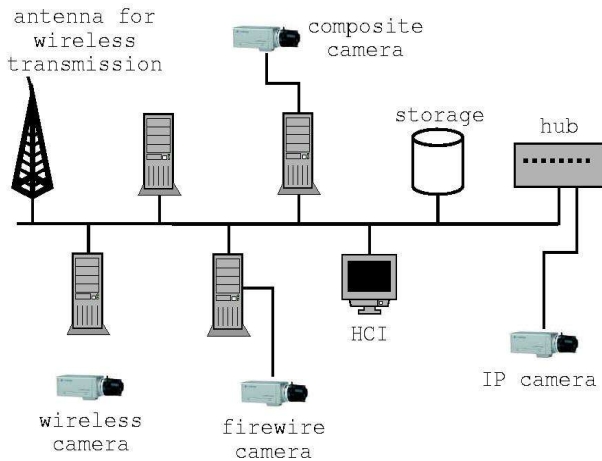


Fig.1. The physical architecture.

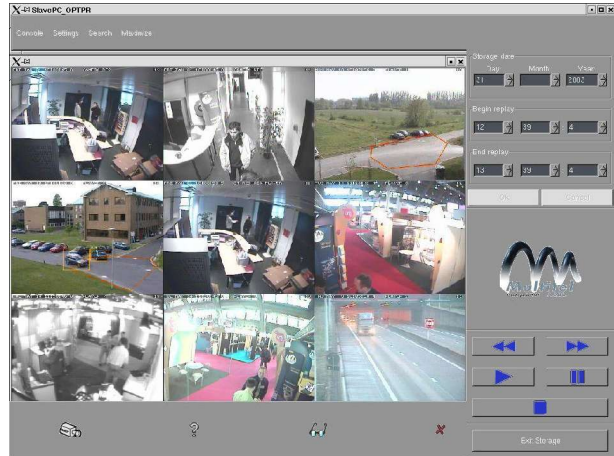


Fig.2. A view of a GUI of the system

The logical architecture has been designed in a modular way to allow a fair resource allocation over the cluster. In its implementation, each software module is dedicated to a specific task (e.g., coding, network management,...). The various treatment modules are distributed on the PC units according to configuration set-up parameters. In these parameters, the operator will define the architecture of the distributed system and moreover, he will be able to customize the action of the different modules (e.g., the vision processing units require a lot of parameters). A master process will be assigned the task of communicating the configurations parameters to all the PC units. Changes in the distribution of the different tasks between the units could be performed dynamically during the run-time of the system.

The robustness of the overall system is provided by the logical architecture. It manages the various problems that can arise: a network packet loss, a network transmission interruption, a hard drive stopping, a frame loss in the video acquisition process, etc. A powerful data management handles the distribution of information along the whole system.

### 2.1 Data management

The system's basic structure for data management between modules handles the concurrent access for sharing information for multiple producers and consumers, the network communication (TCP/IP) for multiple instances of the data (compression handling if necessary), the dynamic connection and access to the stream, the buffer to absorb peak of processing, the implicit conversion of data (e.g. images from RGB to YUV), the propagation of consumers needs to producers (e.g. if no module uses a segmentation result, the segmentation module will be advised not to produce it), the monitoring of the streams (performance, rate), the dispatching priority of data towards specific modules (it helps to process a real-time framework by decreasing latency). Fig.3 represents an example of a stream of data (control, source image, dynamic descriptors of scene, events) and a typical processing. The acquisition of image is done in each frame (25 fps), but the tracking is done half-time (and therefore the interpretation is, too).

## 2.2 Asynchronous data handling.

Video Surveillance applications need to be real-time, because of the security aspect of minimum time reactions. Therefore video-surveillance requires timing constraints for processing. Classical real-time systems usually use internal periodical loop (E.g. a new image is acquired every 40 ms and should be processed, thus the process should last less than 40 ms). If the process costs more than 40 ms for one of the stages of a pipeline (if there is no pipeline, we consider the system as a unique-stage pipeline), a delay is introduced, this delay is limited to a maximum value (e.g. 200 ms), if it exceeds this limit, the system fails (e.g. the memory buffer is full, some data is lost and processing of this data, such as integration or derivation, will fail). To prevent a possible failure, the system usually has an over-capacity, but this has the consequence of under-used most of the time (i.e. when the processing time is not maximum).

We overstep these kinds of problems by adding to each observation (image) a time-stamp, and then consider it to make all processing. (e.g. background adaptation, objects speed). Thus, if processing resources exceed the system's available resources, some information are ignored but without disturbing the integrity of the system. An example of these requirements will be presented through section 4. Table 1 summarizes these concepts.

	<i>Usual Real-Time System</i>	<i>Asynchronous Real-Time System</i>
<i>Sensors Acquisitions</i>	Periodically	When the system has time to do it
<i>Data Processing</i>	Easy to handle because the period between data is constant	Should take the time-stamp into account to perform updates, estimations, integrations, derivations
<i>Use of Processing Resources to Validate Real-Time Constraints</i>	100% . (mean processing time/ max processing time)	100%

Table 1. Differences between a usual real-time system and an asynchronous real-time system.

## 3. DESCRIPTION OF SYSTEM COMPONENTS

The various modules of the software part of the system are explained hereunder. We explain the acquisition, the coding-decoding (codec), the network and the storage modules. Section 4 will describe the image analysis module.

### 3.1. Acquisition

There are two main reasons to handle a large variety of video inputs. First, many industries cannot afford the cost of changing all their installed video surveillance equipment from CCTV to full digital cameras distributed on a network. Second, researchers have to test their video analysis tools on various inputs in order to prove their algorithms or to make them more robust. We are currently able to handle several protocols: IP (JPEG and MJPEG), IEEE1394 (raw and DV), wireless (analogic, wifi) and composite (PAL, NTSC, SECAM). A time-stamp is attached to each frame at grabbing time. This information is very useful in the subsequent processing stages as showed in 2.2. For some reasons it could be interesting to have a calibration of the camera (e.g. for multiple cameras application or when doing 3D with ground plane area). An easy and manual tool for calibration (see Fig. 4) has been developed with the same procedure as [6].

### 3.2. Codec

The goal of the coding process is to reach a good compromise between the compression ratio (bandwidth occupation) and processing resources. We propose here a MPEG4 compression scheme since it outperforms classical MJPEG encoding. The main disadvantage of MJPEG is that it does not use any temporal redundancy to increase the compression factor. In our experience MPEG4 and MJPEG compression factor are in a ratio of 1:10 for the same quality. Indeed, video-surveillance scenes are quite static when cameras are fixed. Compression methods suppressing temporal redundancy, such as MPEG4, are therefore more efficient. Nevertheless, we cannot access images independently anymore. We have to re-synchronize the flow on an I-picture (intra picture) each time a network transmission occurs. The number of I-pictures per second is determined with respect to the VCR that replays the stored sequences. Moreover, in order to limit the delay between encoding and display time, we do not encode pictures as B-pictures (bi-directional encoded pictures) to keep a causal decompression scheme. This technique allows us to transmit up to 20 CIF (352x288) video flows at 25 fps on a typical 100-base-T network. Thanks to this compression scheme, we are able to transmit video flows on new mobile phone networks (e.g. when an alarm is generated).

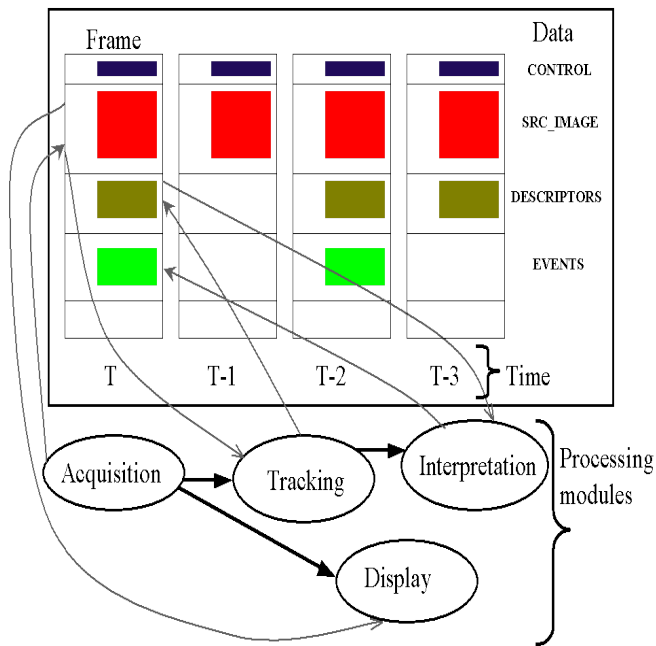


Fig. 3. Example of data management.



Fig. 4. Calibration tools used for PETS 2003 sequences.

### 3.3. Networking

Having a distributed system implies an efficient use of the bandwidth. We have seen that the various modules related to a video input can be dispatched on several computers. For example, the system can perform the acquisition on computer A, the storage on computer B and the display on computer C. We have chosen a multicasting technique to solve the bandwidth occupation problem. Each video source has an associated multicast channel. This multicast channel is accessed through an UDP connection by every module that needs the video input. Since UDP does not offer a quality of service (QoS), we have developed a protocol that can detect when a transmission failure occurs. But if we assume to use the system on a dedicated network, there are only few occurrences of transmission failures. We guarantee small delays because the network load is controlled in order to avoid buffer queuing. The decrease of the delay could be improved by passing through the ISO networks layers and use IP packets with the knowledge of its implementation on the Ethernet layer. This delay is small enough to be imperceptible for the user (i.e. less than 150 ms).

### 3.4. Storage

The storage module has to deal with the enormous quantity of data to store. It must allow a 24-hours per day storage. This module has two levels: level 0 is a classical storage process with the MPEG4 technology. This level stores a CIF video flow at 25 fps for 7 days on a 80 GB hard drive. We can further improve this number if we allow a second-pass encoder to have a constant quality stream. Up to now, we have a constant bandwidth stream. Level 1 is an intelligent storage process. It stores only interesting events that the user has defined. This level saves tremendous storage space. Moreover, it allows a fast search to retrieve a stored sequence.

## 4. IMAGE ANALYSIS MODULE

The architecture of the vision part of the system is divided in three main levels of computation that achieve the interpretation (Fig. 5): Image level (image filtering, background evaluation and segmentation), Blob level (description, blobs filtering, matching, tracking description and filtering), event level (tracking analysis, finite state machine, performance evaluations). We will only describe here the image and object levels (Fig. 6 & 7) as the event level is quite application dependant.

#### 4.1 Pre-processing and segmentation

After acquisition, the current image is filtered in the pixel domain to decrease the spatial high-frequency noise. The image is then downsized to reduce the need in computational resources of the segmentation process. Many reference image models could be used for representing the backgrounds as they are implemented in the system. However, these algorithms should satisfy the non-periodic processing framework of the system as described in 2.2. Thus the segmentation process is divided in two stages: update and estimation. As shown in Fig.6, the segmentation is fully configurable as one can choose between three types of background models (e.g. mixture of Gaussian [5], low pass temporal recursive filter[1], temporal median filter) and foreground extraction processes, even for non-background segmentation like frame differencing. For the application shown in section 5, the system will only use the mixture of Gaussians background, as it is quite robust to common noises such as monitor flatterring or branches moving in trees. The update stage will revise the mixture of Gaussian for each pixel background, and the estimation stage will extract the *a priori* foreground, filter it via morphology processing and then label objects.

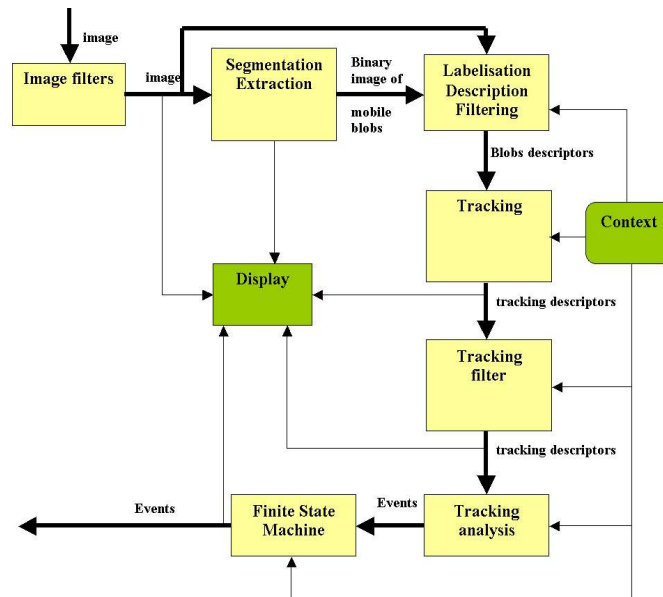


Fig.5 . Design of the vision system components.

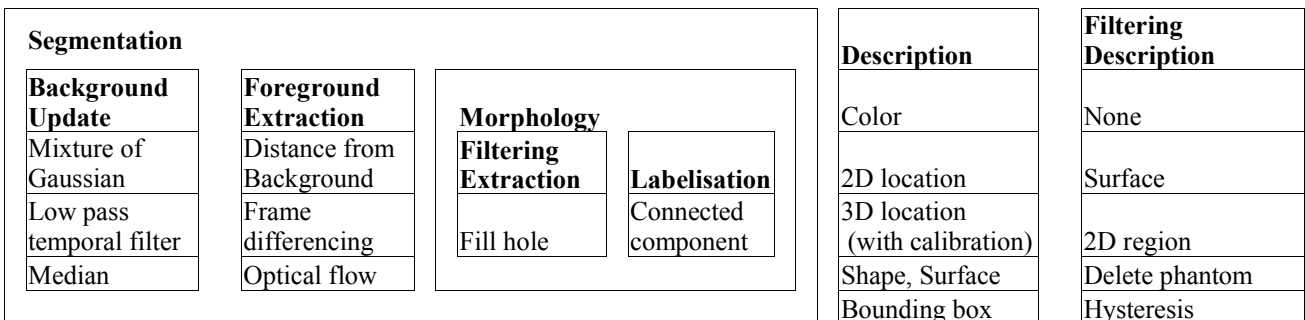


Fig. 6. Image and description level: Segmentation and Description

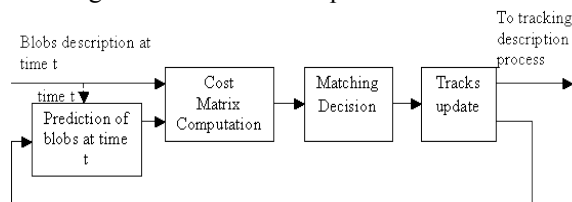


Fig. 7. Basic architecture of Tracking.

#### 4.2 Blobs description and filtering

The aim of blobs description and filtering is to make the interface between segmentation and tracking and simplify the information. The description process translates video data into a symbolic representation (i.e. descriptors). The goal is to reduce the amount of information to what is necessary for the tracking module. The description process calculates, from the image and the segmentation results at time  $t$ , the  $k$  different observed features of a blob  $i$ : 2D position in image, 3D position in the scene, bounding box, mean RGB color, 2D visual surface, inertial axis of blob shape, extreme points of the shape, etc. At this point, there is another filtering process to remove small blobs, blobs in an area of the image not considered, etc. Other model-based vision descriptors could be also integrated for specific application such as vehicle or human 3D model parameters.

#### 4.3 Tracking algorithm

The tracking part of the system is flexible and fully parametrical. The set-up should be done for a trade-off between computational resources, needs of robustness and segmentation behavior. It is divided in four steps that follow a straightforward approach: estimation, cost matrix computation, matching decisions, tracks updates. Note that there are multiple predictions and cost matrixes when the last matching decision is not unique, and there are only multiple matching decisions for some matching algorithms in MHT (multiple hypothesis tracking [6]). Fig. 7 explains briefly the architecture.

The tracking filtering is processed at the tracking description output. It is just as necessary as the other filters of the vision system. As usual the filter is used to remove the noise. At this level of processing, it can use the temporal consistency. We described above some types of filters that can be used in chain. Because the tracking description is a construction built piece by piece during the progression of the video sequence, it can process on-line or off-line. One filter detects and removes tracks that last for less than a fixed duration. This kind of noise comes when the segmentation detects noise in the image as an object. Another filter simplifies tracks by removing samples of blobs that give poor information (e.g. If the blob moves slowly). It could be seen as a dynamic re-sampling algorithm. A tracking result is shown on Fig. 8.

### 5. CASE STUDY

The developed system has a lot of high-level processing modules (detection of activity, abandoned object, missing object, people running too fast, car park management, etc.) that can be run as options in accordance with the end-user application. Here we decide to focus on one: people counting in in-door scenes. This module should be able to evaluate the number of people moving from a region of a scene to another region of the scene. It could be useful for surveillance, marketing (e.g. statistical information of entering/exiting people in a shop). The best results are obtained when using a large view camera positioned vertically and watching the observed zone. Because of segmentation and tracking, the individual movement of people is known, thus it is possible to identify an abnormal displacement in the observed zone. The set-up of this module requires the definition of the context, (i.e. a region of interest), where the tracking is working on and the counting zone, where the counting is achieved in accordance with given directions. We can also specify authorized or non-authorized displacements (e.g. to highlight the alarm when somebody tries to exit through an entrance door). See Fig. 8 for a context view.

The major problem when counting people is to ensure that there is no mistake when two people are close together. Indeed, the image segmentation process could see them as a single object. To bypass this problem, the counting module splits objects by using the historic tracking of each of them (e.g. when two people crossover as shown in Fig. 9). This algorithm is described as follows: if one object surface is superior to a defined threshold, the object tracking history is investigated, thus if this object is stemming from a merge of two objects, it will be considered as a fusion of two objects. Then the algorithm will split this object.



Fig. 8. Display of people counting module. Left: video and context, right: tracking result.

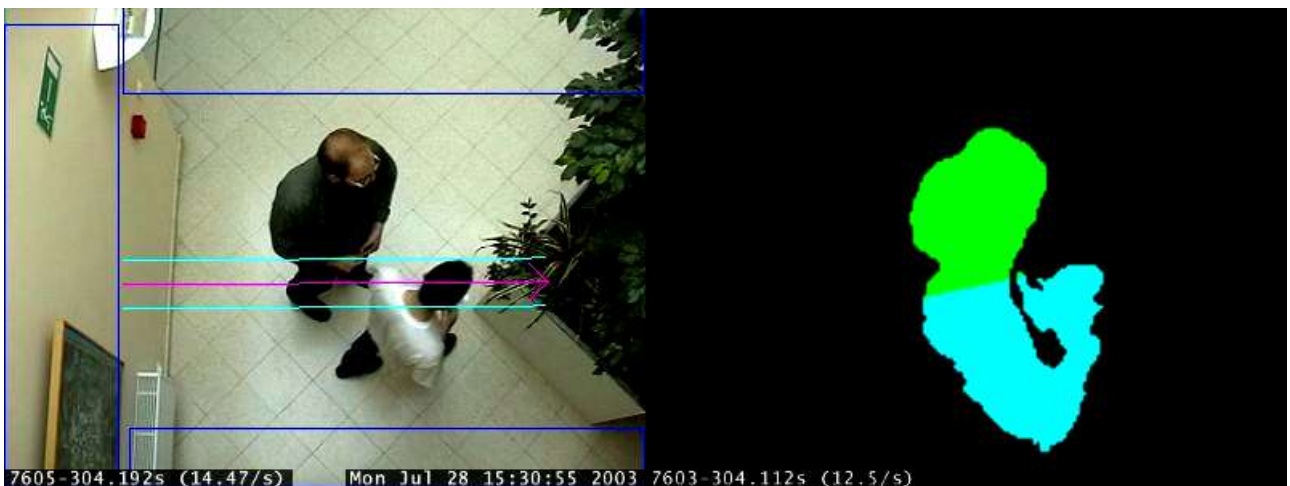


Fig. 9. Display of people counting module. Left: video and context, right: segmented blobs, separated in two parts.

## 6. CONCLUSION AND FUTURE WORK

In this paper, we have proposed an original approach for a third-generation video-surveillance platform [7] that can provide the flexibility needed by researchers and that can meet the strong efficiency requirements of industrial applications. This system is presented as being very efficient and it has been validated on various uses.

We are currently investigating new vision modules, e.g. better segmentation and tracking methods. Moreover, other extensions and improvements will be made on the global system. Future works will also integrate mobile low-bandwidth communication like GSM/GPRS to communicate parameters, results and logs over long distance.

**Acknowledgements:** this work has been granted by the Walloon Region under the FEDER project 171, the FIRST SPIN OFF program.

## REFERENCES

- [1] B. Georis, X. Desurmont, D. Demaret, J.F. Delaigle and B. Macq, "IP-Distributed computer-aided video-surveillance system", Intelligent Distributed Surveillance Systems Workshop, London, UK, February 26, 2003.
- [2] A.Cavallaro, D. Douxchamps, T. Ebrahimi and B. Macq, "Segmenting moving objects : the MODEST video object kernel", WIAMIS 2001, Workshop on Image Analysis for Multimedia Interactive Services, Tampere, Finland, May 16-17, 2001.
- [3] Cupillard F, Brémond F and Thonnat M, "Tracking groups of people for video surveillance", 2<sup>nd</sup> European Workshop on AVBS Systems.
- [4] T. Shcoepflin, C. Lau, R. Garg, D. Kim and Y. Kim, "A research Environment for Developing and Testing Object Tracking Algorithms", Proceedings of the SPIE, Electronic Imaging 2001, vol. 4310, pp. 667-675.
- [5] C. Jaynes, S. Webb, R. Steele and Q. Xiong, "An open development environment for evaluation of video surveillance systems", 3<sup>rd</sup> Int. Workshop on PETS, 1, 32-39
- [6] I.J. Cox and S.L. Hingorani, "An Efficient Implementation of Reid's Multiple Hypothesis Tracking Algorithm and Its Evaluation for the Purpose of Visual Tracking".
- [7] Marcenara L, Oberti F, Foresti L and Regazzoni C, "Distributed architectures and logical-task decomposition in multimedia surveillance systems", Video Communications Processing and Understanding for 3GSS, Proceedings of the IEEE, 89, 1419-1440.