UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

ONTOLOGY-GUIDED SEMANTIC VIDEO DESCRIPTION USING FEEDBACK BETWEEN SIGNAL PROCESSING STAGES

Juan Carlos San Miguel Avedillo
Supervisor: José María Martínez Sánchez

-TRABAJO FIN DE MASTER-

Departamento de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Noviembre 2008

ONTOLOGY-GUIDED SEMANTIC VIDEO DESCRIPTION USING FEEDBACK BETWEEN SIGNAL PROCESSING STAGES

Juan Carlos San Miguel Avedillo

Supervisor: José María Martínez Sánchez

email: {Juancarlos.Sanmiguel, JoseM.Martinez}@uam.es

Video Processing and Understanding Lab

Departamento de Ingeniería Informática

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Noviembre 2008

Abstract

In this work, the improvement of the semantic description extraction process from video sequences using contextual information (from a specific application domain) and feedback strategies between different processing stages is proposed. The analysis of video-surveillance sequences is proposed as a case of study. Contextual information is applied by using an event detection ontology and the feedback schemes are proposed to provide a re-evaluation of the unknown detected events. The results show that combining the use of contextual information and feedback strategies improve the computational efficiency and the correct event detection rate.

Acknowledgements

First I would like to thank my supervisor, Prof. Dr. José María Martínez Sánchez, for their support and advice during this work.

I also wish to thank all members of Video Processing & Understanding Lab for their support and friendship. Each one has contributed to finish this work in some way. In particular, without Alvaros's help, I would have found the strange world of ontologies much more painful.

I would also thank Prof. Dr Jesús Bescós Cano for his advices during this work.

I also would like to thank Sonsoles for her love, support and patience that made everything easier for me.

Finally, I would like to thank my family for always being there supporting me. Without them I would never come up to this stage.

Juan Carlos San Miguel Avedillo
November 2008

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In recent years, much of research effort in video event analysis has been focused on extracting semantic descriptions from video sequences (number and type of objects, relations between them, events,...etc.). Most of this previous research takes as good the results obtained from previous analysis stages like object segmentation, object tracking and object recognition (like people, cars, buildings, ...) which are traditionally supposed independent between them. Nevertheless, the remarkable dependence that the results present with the application domain and the close dependence of the results of an algorithm with respect to other subsequent algorithms (for example a poor segmentation makes recognition very difficult), are generally accepted facts[1].

In extending the semantic (or high-level) interpretations from video sequences, one of the challenges is to exploit expectations derived from high-level structures to improve the low-level processing stages. In this context, the use of complex feedback strategies can enhance the performance of the lower levels of processing for feature extraction and consequently the semantic information extracted will be more accurate. The feedback strategies are processes whereby some proportion of the output signal of a system is passed to the input. The general idea is to compute the error signal between the image data and the hypothesized model and use this error signal to accept or reject the hypothesis. If the current hypothesis is rejected, the error signal is used to generate a new hypothesis and to tune the parameters of the analysis algorithms to perform a more accurate detection. Finally, the corresponding error signal is computed and the feedback process begins again until the hypothesis is accepted. For example, feedback strategies can be used to analyze the scene with a different degree of detail, which implies applying different techniques to extract image information and different degrees of precision in segmenting the objects of interest[2]. Other studies (like [3]) show that feedback enables higher levels of processing to tailor the general-purpose pixel-level segmentation.

The use of context information about how a video sequence has been captured or the location of some objects in the scene generally allows the improvement of the quality of the results from analysis and recognition stages. Traditionally, this contextual information has been incorporated by manual adjustment of the algorithm's parameters or implicitly in the structure and design of the algorithms. In the last years, the research community is proposing alternatives based on the use of formal descriptions of the content modeled with ontologies. The ontologies had been used successfully for semantic video description and for searching/automatic indexing in multimedia databases (e.g., [4]). The use of ontologies for representing the contextual information has caused the development of knowledge-based vision systems[5]. These systems try to find solutions for two basic problems: how to build and represent the object ontology and how to use the ontology for improving the analysis stage results.

1.2 Objectives

The main objective of the work presented in this document is to study the effect of the application of contextual information in the video analysis process and to use this information to provide feedback between analysis stages with the purpose of improving the performance of the analysis, interpretation and semantic adaptation processes operating with time constraints (real-time). To achieve this objective, a new visual analysis methodology has to be designed. It will allow to customize the visual analysis procedures using the contextual knowledge explicitly represented. Therefore, the presented work will have the following objectives:

- To study the state of the art in the following topics: visual analysis systems, representation of the knowledge in computer vision using ontologies and feedback strategies between processing stages.
- To design a visual analysis framework integrating common techniques for semantic information extraction or event recognition.
- To design and develop a generic ontology to represent the contextual information associated to the detection of video events. Later, this generic ontology will be particularized to detect events in the video surveillance domain, adapting the semantic analysis techniques to ontology concepts.

- To design and develop different feedback schemes using the partial output results of the analysis stages to achieve two objectives: the use of generic analysis algorithms (that will be customized, by the use of the ontology, depending on the context of the application domain) and an improvement of the system performance (due to the correlation between concepts and stages).
- To evaluate the quality of the final results using objective techniques, as well as its computational execution cost. Furthermore, this objective includes the design of an appropriate evaluation methodology and a dataset to evaluate the techniques proposed.

1.3 Document Structure

The structure of the document is as follows:

- Chapter 1. This chapter presents the motivation and the objectives of the work presented.
- Chapter 2. This chapter presents an overview of the literature related to the work presented in this document.
- Chapter 3. This chapter presents the ontology design and development for the representation of the contextual information. This ontology is particularized in the underground video-surveillance domain.
- Chapter 4. This chapter presents the framework and the techniques used for the visual analysis task. These techniques are mainly focused on foreground object extraction, object tracking, object classification and event detection.
- Chapter 5. This chapter presents the feedback schemes developed to increase the overall system performance. These feedback schemes are focused on the improvement of the proposed framework in the previous chapter.
- Chapter 6. This chapter presents the dataset used, the evaluation process and some experimental results. Furthermore, a comparison of the system results with/without using the feedback path is presented.
- Chapter 7. This chapter summarizes the main achievements of the work, discusses the obtained results and provides suggestions for future work.

At the end, five appendices list further details. They describe the complete ontology developed in this document (Appendix A), the ViPER toolkit used for the evaluation process (Appendix B), some color conversion schemes used in this work (Appendix C), the mean-shift and the PCA algorithms (Appendix D) and some mathematical issues related with the foreground detection stage (Appendix E).

Chapter 2

State Of The Art

This chapter gives an overview of previous work that has been done in the scope of the study presented in this document. In the next sections, we describe the areas of visual analysis (section 2.1), knowledge representation (section 2.2) and feedback strategies used in visual analysis (section 2.3).

2.1 Video Analysis

As we can see in Figure 2.1, the processing stages of an automated video analysis system can be listed as foreground object detection, object tracking, object classification and event or action recognition[6]. The following sections describe the related work found in the literature on each of these fields.

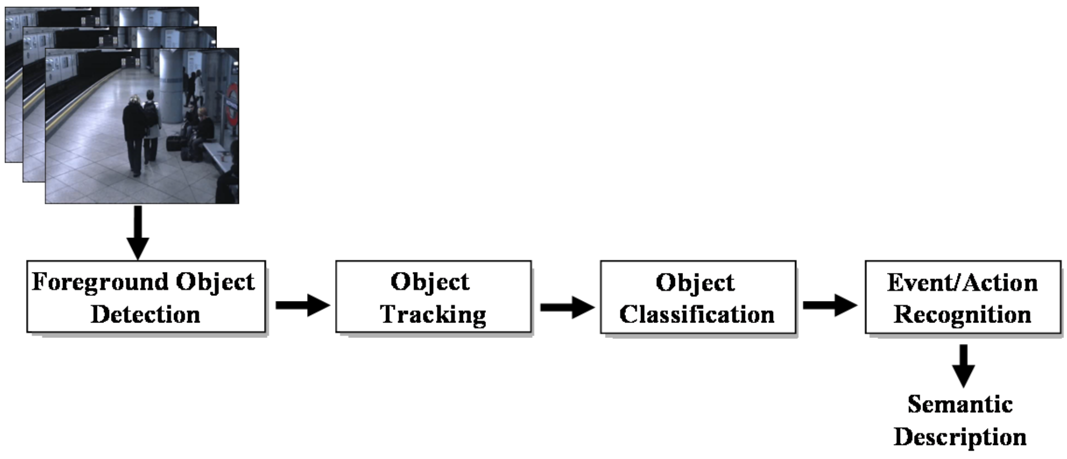


Figure 2.1: Generic video analysis system

2.1.1 Foreground object Detection

Foreground object detection or segmentation is the first basic step of almost every video surveillance system. It identifies meaningful components of an image providing a focus of attention for the subsequent processing stages. In the surveillance domain, the relevant objects are extracted by analyzing the motion of the scene [6]. Classifications of moving object segmentation algorithms vary significantly in the literature and no consistent classification can be found. Most of them are either ambiguous or not complete. For example, in [7] the algorithms are classified into four categories: 3-D segmentation, motion-based segmentation, spatio-temporal segmentation and joint motion estimation and segmentation. Another survey [8] distinguishes two main categories: motion-based and spatio-temporal. Among motion-based techniques, there are two subgroups based on the dimension of motion models employed (2D or 3D). The spatio-temporal category includes algorithms that use the spatial information to rectify and improve the temporal segmentation results. A simplified version of these two categories is shown in Figure 2.2.

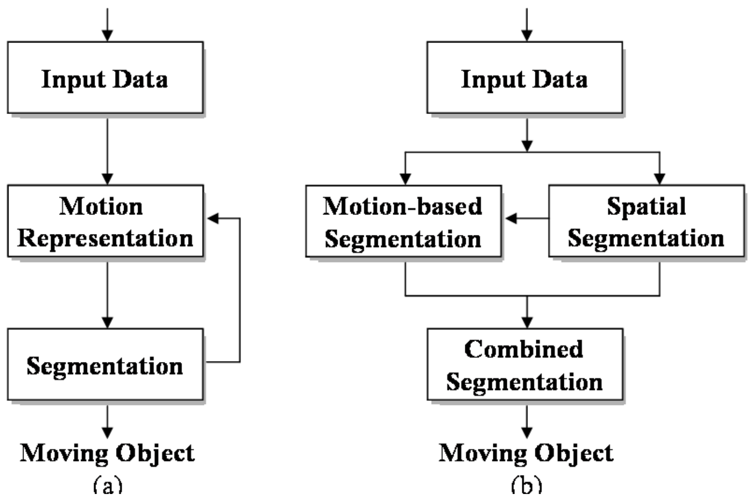


Figure 2.2: Classification of segmentation categories: (a) simplified motion-based segmentation and (b) simplified spatio-temporal segmentation

In the case of 2D segmentation using motion-based information, the most popular approaches include temporal differencing [9], optical flow [10] and background subtraction [11]. Moving object segmentation is a difficult task with some significant problems that need to be handled well for obtaining a robust video analysis system. They are:

- How to represent the motion/spatial information of the scene and combine them?

- How to pre- and post-process the video in order to eliminate the noise artifacts?
- How to deal with dynamic environmental conditions such as illumination changes, shadows, highlights and complex backgrounds?

In the following subsections, the state-of-art in moving object segmentation is overviewed focusing in the use of background subtraction techniques and the application of some post-processing stages (noise removal, shadows/highlight detection,...) to improve the accuracy of the moving object extracted.

2.1.1.1 Background subtraction

Background subtraction is a commonly used technique for motion segmentation in scenes captured by a static camera that tries to detect the moving regions that do not belong to the background of the scene[12]. The basic scheme of this method is shown in Figure 2.3. This method is based on the pixel-by-pixel difference between the current image It and the reference background image Bt at time t. Then, the pixels where the difference is above a threshold τ are classified as foreground as it is shown in the following equation:

$$F_t \Longleftrightarrow |I_t - B_t| > \tau \tag{2.1}$$

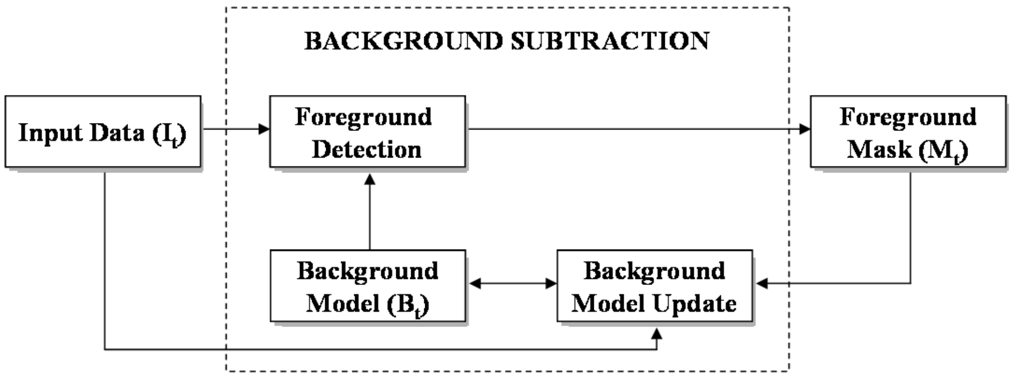


Figure 2.3: Basic background subtraction scheme

According to recent surveys like [13][14][15][16], there are three issues that characterize background subtraction methods: model representation, model initialization and model adaptation. The first describes the kind of model used to represent the background; the second one takes into account the initialization of this model, and the third one relies to the mechanism used for adapting the model to the background changes (e.g. illumination changes).

The simplest background models suppose that the camera is fixed and the background is relatively constant and therefore propose modeling the background as a single static image. The most simple solutions include running average or the mean/median of the last N frames [17]. The main disadvantages of these simple approaches is that they do not provide information to choose the correct threshold τ used in the subtraction process and they do not provide a rigorous statistical description of the background. Furthermore, some of these approaches may require high memory resources (like the median-based approach in which the storage requirements can be high if a large buffer is needed to model the variation of the pixel values). When the background is not constant and presents complex movement, more tolerant models are required. These methods can be classified into parametric or non-parametric.

Parametric methods estimate the background pixels distribution as a known distribution, providing also a way to calculate the threshold. For example, to cover small variations in the background pixels we can use the mean intensity μ and variance σ of each pixel [18]. In this case the threshold is calculated proportionally to the variance of each pixel (e.g. $k = 3$):

$$|x_t - \mu| < k\sigma \tag{2.2}$$

In this approach the use of a k is less critic than using a fixed threshold. For updating the background model, a scheme called Gaussian Running Average (GRA) is used as shown in the following equation:

$$\begin{aligned} \mu_{t+1}^2 &= \alpha F_t + (1-\alpha)\mu_t \\ \sigma_{t+1}^2 &= \alpha(F_t - \mu)^2 + (1-\alpha)\sigma_t^2 \end{aligned} \tag{2.3}$$

This approach is very efficient in terms of memory consumption and robustness to noise. Another interesting approach is proposed in [19]. In this paper a transformation of the three-dimensional RGB space to a two-dimensional space called angle-module is presented (truncated-cone method). Then the foreground maps are calculated by thresholding the angle and the module and the background model (angle and modules) is updated by using a running average scheme.

The previously proposed solutions can fail if the pixel variations become more significant and non-uniform (complex backgrounds). Therefore more robust background models have to be designed. In [11], the background pixels distribution is estimated by using a finite Mixture of Gaussians (MoG) . The parameters of this model are then k mean values, k covariance matrices and k scaling factors to weight the relevance of each Gaussian. The main problem of this method is the selection of the value of k, that is, the number of Gaussians.

Non-parametric methods try to obtain a more accurate detection by estimating the density function of the pixel's distribution making use of recent pixel intensity values. In [20] the probability density function that a pixel will have a certain intensity (belonging to the background) is non-parametrically estimated using the kernel estimator [21]. This method is called Kernel Density Estimation (KDE). Another non-parametric method is presented in [15]. The background model consists of one histogram per pixel with the last N values. Then the mean shift algorithm is used to find the maximum local points of the histogram (a detailed description of the mean-shift algorithm can be seen in Appendix D.1). The decision taken for segmenting into foreground or background is performed by evaluating the probability density function previously calculated:

$$PDF(x_t) < Th \Rightarrow x_t \in foreground \tag{2.4}$$

This model presents a good performance dealing with complex backgrounds but the required resources and computational cost are high. Another interesting approach is described in [22]. Principal Component Analysis (PCA) is applied to a sequence of N frames for computing the *eigenbackgrounds* (a detailed description of the PCA algorithm can be seen in Appendix D.2). The basic idea is to use PCA to keep the eigenvectors that have most information (and belong to the background). Then, for each new frame, the projection of the new frame in the eigenvectors sub-space and the reconstruction are calculated. The difference between the original and reconstructed frame is used to extract the foreground objects.

For concluding this section, a more detailed discussion of the algorithms previously described can be seen in [15].

2.1.1.2 Shadow detection

In the moving object segmentation process, moving cast shadows [23] are usually misclassified as a part of the moving object making the upper levels, such as object classification, to perform inaccurate. Several shadow detection algorithms have been proposed and they can be classified by their use of chromaticity information [24][17][19], edge information [25], stereo information (or geometrical properties)[26] and a combination between them [27][28][29].

In [24] each pixel is represented by a color model that separates brightness from the chromaticity component. A given pixel is classified into four different categories (background, shaded background or shadow, highlighted background and moving foreground object) by calculating the distortion of brightness and chromaticity between the background and the current image pixels. In [17] the colour independence property in the HSV colour space is used to detect shadows and highlights. It is observed that the *hue* and *saturation* components of a background pixel only change within a certain limit even if they are covered by a shadow or highlight. However, the hue components on pixels with saturated or poor illumination are usually unstable. A similar approach is presented in [19] where RGB color space is used to extract the chromaticity information. In other approach [25], it is assumed that the shadow often appears around the foreground object and they try to detect shadow by extracting moving edges. An efficient method to deal with shadows using stereo information is presented in the W4S system [26]. In W4S, stereo information is used to detect shadows, sudden illumination changes and complex occlusion cases. [29] proposes a fusion scheme exploiting the information of colour, texture, neighborhood and temporal consistency to detect shadows efficiently and adaptively. Stauder et al. [27] relied on the brightness, edge and shading information to detect moving cast shadows in a textured background. A similar approach to [17] is employed in [28] where the HSV colour space is combined with the extraction of moving edges.

2.1.1.3 Noise removal

As it is well known[19], the scheme introduced previously for detecting moving objects can produce inaccurate foreground masks when the background model is not able to handle correctly intensity variations of the input video signal. These variations introduce noise in the output foreground pixel masks and the following processing stages can be highly affected and can reduce their performance drastically. In order to obtain better results, noise removal is a key issue. Some simple (but effective) algorithms are used to deal with this problem: morphological operators and Connected Component Analysis (CCA).

Morphological operators usually work on binary images by using a structuring element and a set operator (intersection, union, etc). The structuring element determines the area of the noise to be eliminated. The basic operations of interest in this context are erosion and dilation [30]. Although the application of these operators eliminate noise in the binary mask, the contours of the blobs are modified reducing the accuracy of the following object recognition algorithms applied. More recently, new morphological operators have been proposed[31] that preserve the contour of the blobs in the binary image.

Connected component analysis (CCA) groups the pixels in an image based on pixel connectivity. For binary masks, the method typically used is a growing region algorithm [30]. It tries to identify the connected components (with value 1) by analyzing the value of the pixels and their neighbors as described below :

Algorithm 1 Connected Component Analysis

```
1. Pixels of the binary image are scanned
2. If the pixel under consideration is a foreground pixel (having value 1):
    (a) Scan values of neighbors pixels (using 4 or 8 connectivity)
        i. If one of the pixels is labeled, this label is copied as the label of the
           current pixel.
    (b) If none of the neighbors has a label, current pixel is given a new label
3. All pixels on the binary image are scanned considering the rules defined in Step
   2. As a result, all isolated groups of pixels are given a distinct label as a result
   of the algorithm (Figure 2.4).
```

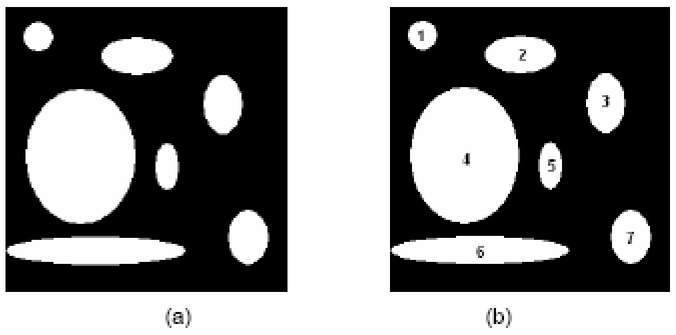


Figure 2.4: Connected component labeling on a binary image

CCA is a powerful tool that gives important information about the objects in the change mask. It provides the area of a region, the number of moving objects in the scene and the bounding boxes associated (width, height and center). These results are very crucial in such an automated video analysis system. Indeed, the tracking algorithm is based on such information.

2.1.2 Object tracking

After the moving object segmentation process, the problem of establishing a correspondence between object masks in consecutive frames should be solved. Tracking refers to predicting and establishing the position and orientation of an object in an image sequence. In this document, we formulate object tracking as a region correspondence problem, given the foreground segmentation results previously obtained.

The object tracking process can involve any moving object in the scene without recognizing the type of the object (*precategorical* tracking[32]) or recognizing the specific object to track usually employing a model (*categorical* tracking[32]). Thus, there are two main approaches to solve the tracking problem:

- **Based on the analysis of the pixel's content**. These techniques segment the regions of the image where the movement is produced and then, they do the tracking process by analyzing the pixel's content. Examples of this approach are methods based on the analysis of differential movements [33] or on the optical flow analysis [34].
- **Based on object models**. These techniques imply to localize, in each image in the video sequence, the position of the specific moving object (for which we have a model). The object models are built "by hand", induced from a sequence of examples or dynamically acquired from the moving object. The model-based methods include: correlational methods [35], methods based on correspondence [36], filtering methods [37], methods based on deformable contours [38] and methods based on predictive filtering [39].

2.1.3 Object classification

Object classification is an important analysis stage for getting semantic descriptions of the analyzed video sequences. The objective is the identification of the different objects that can appear on the video sequence. In the context of video event detection, one of the key issues is to identify the people that appear in the video sequence as they are mainly the agents performing the actions to be detected. As a result of this fact, this state of the art is centered in people detection and classification.

In the case of people detection the larger and more unpredictable freedom of movement poses additional complications to the classic moving object detection problem. Existing algorithms can be divided depending on their use of motion or static features (such as shape, color and texture) to detect people. Motion-based algorithms are usually employed after a tracking stage and they typically use the periodicity of human motion as a strong cue for people detection [40]. Static feature classification algorithms can be classified depending on whether they analyze contours (silhouettes) or regions. Most of current static-feature people detectors belong to the contour analysis family. In many cases, they apply a training phase for determining a set of silhouettes representative of common human poses. Some examples are stick figure [41] and silhouette models [42]. A different approach proposed in [43] does not keep full silhouettes but distinctive segments of them referred to as edgelets. Others schemes are directly based on information extracted from the analyzed silhouettes. For example, the technique in [26] determines the pose and location of body parts from the silhouette of a person. Iterative ellipse fitting is proposed in [44]. Alternatively, other even simpler blob features like aspect ratio, bounding box or blob area are used [9]. Finally, there are algorithms that combine the motion and static features to get the advantages of both approaches [45][11].

2.1.4 Event Detection

One of the main purposes of an automated video analysis system is to understand what is happening in the sequence being analyzed. Event detection is probably the ultimate purpose of a fully automated surveillance system. Even though it is quite important and useful to recognize an activity, it is not easy to define the type of features that are interesting and meaningful in each context. The methods proposed in the literature can be classified depending on the type of approach followed and features used.

Depending on the type of model used, the proposed methods can be divided into deterministic and statistical.

- **Deterministic approaches** model each variable/parameter with a defined value (or sequence of numbers) under a fixed number of restrictions. They use a priori knowledge to model the events to recognize and this knowledge usually corresponds to rules defined by experts from the application domain. These techniques are easily understandable since they are based on constraints which are defined in a declarative way. However, they do not take into account the numerical variation of data. In consequence, it is difficult to model the real-world variety. For instance, they can use finite state automata [46], Petri nets [47], or logic programming [48]. In order to obtain more flexibility, probabilities can be used to weight the rules defined by the experts [49]. Another approach is presented in [50] to optimize the temporal constraints resolution by ordering in time the sub-events of the event to be recognized. A very efficient algorithm which follows this approach takes advantage of a pre-compiling stage of event models to recognize in real-time complex events involving a large number of physical objects [51][52].

- **Statistical approaches** rely on learning. These approaches do not know the models with precision and therefore they are described with statistical approaches. Sometimes, statistical approaches have the advantage of being more simple and robust to changes in the model parameters. For instance, HMMs (Hidden Markov Models) [53], NNs (Neural Networks) [54][55] and Bayesian Inference [56], are a first class of techniques frequently used to recognize events in video streams. Dynamic Bayesian Networks (DBN), of which Hidden Markov Models (HMM) are a special case, have proved robustness to data variations in the input streams and good performance with respect to generalization, as shown by their frequent use in activity recognition [22][57]. A drawback, however, is that the amount of labeled training data needed. To handle this issue, several modeling techniques such as multi-streams HMM and layered-HMM [22][57] have been proposed to reduce the complexity by breaking the problem into a different layers responsible of recognizing sub-events (lower layer) or the complex event (higher layer) from the sub-event probabilistic output streams.

Depending on the underlying features used, the proposed methods for video event recognition can be divided into three sub-categories:

- **those based on 2-D image features**. For instance, the authors of [58] recognize human actions at a large distance (at small resolutions) by using a motion descriptor based on optical flow measurements in a spatio-temporal volume. Other authors [59] use the projections of the 3D body shape/contour into the 2D image plane to detect actions. In [60], the idea of spatial interest points is extended into the spatio-temporal domain. These new points often reflect interesting events that can be used for a compact representation of video data as well as for its interpretation
- **those based on 2-D trajectories of tracked objects**. In this category the approaches differ on the granularity of the object tracking phase. For example, in [61] there is a tracking stage for the hands, legs and the whole body to detect the actions performed by them. Other approaches, like [56], present the recognition of events based on a tracking stage of the blobs extracted from the scene and a classification stage to distinguish between different objects.

These approaches based on blob tracking analysis have a reduced expressiveness, limiting the number of events to detect.

- **those based on 3-D body pose** acquired from motion capture or 3-D pose tracking system. For example in [62] the 3D space is decomposed into a set of feature spaces where each feature corresponds to the motion of a single joint or combination of related multiple joints. For each feature, the dynamics of each action class is learned with one HMM and then action recognition is computed using a weak classifier.

2.2 Knowledge representation and ontologies

2.2.1 Introduction

Knowledge can be defined as an organized collection of data designated to solve a specific task. This collection is based on a conceptualization and in this document, it includes objects and other entities that are assumed to exist in an area of interest (scene) and the relationships between them (actions or events). This conceptualization is an abstract view of the domain of interest, that we want to represent with a purpose. Subsequently, a knowledge representation structure has to be selected to represent these concepts. At this point, the most popular choices are: relational data models and ontologies.

The relational data model describes the logic structure of the data and its application. One of the most important models of this category is the entity-relationship model. This model describes schematically the possible instances of the concepts represented. These instances represent the data used by the final application. Many extensions have been added to the entity-relationship model to enrich the data in a semantic way. One common extension is to use a class-subclass hierarchy. Despite of this, the data model has limitations in its design: it presents only a view of the world and it is not reusable (and it is evident that one concept can have different interpretations depending on the context).

On the other hand, ontologies [63] allow to manage more easily the knowledge information. An ontology is a set of concepts and relations between them shared by the community experts of a given domain (like video surveillance domain). Moreover, they add expressiveness and reasoning capabilities. Ontologies make systems user-centered and enable experts to fully understand the terms used to describe the knowledge models. They provide an easy way to capture the domain knowledge and make it usable by people and automatic systems. They facilitate the interoperability between systems providing a standard shared knowledge for each specific domain. At last, it has to be highlighted that ontologies have to be build, refined, adapted and integrated by domain experts.

2.2.2 Ontologies for video events

Recently, the use of ontologies for representing the a priori knowledge (or context information) has been proposed in the video analysis research community [64]. In each domain, the determination of the number and type of the objects that can appear in the scene allows to link the visual analysis process with the specific domain ontologies (that is, descriptions of possibly detectable objects, relationships between objects,...). This approach is currently obtaining promising results [1][65]. Moreover, the ontology is useful to evaluate scene understanding systems, to understand exactly what types of events a particular system can recognize, and to share and reuse models dedicated to the recognition of specific events.

The use of the ontologies for representing the contextual information has caused the development of knowledge-based vision systems [5]. These systems have to deal with two basic problems: how to build and represent the object ontology and how to use the ontology for improving the analysis stage results.

For solving the first problem, most of the proposals define the components and the ontology structure in a manual way (e.g. [5][66][67]), obtaining the low level attributes by extracting visual descriptors from available training sequences. Commonly, a generic ontology is defined for the analysis problems and so many specific ontologies as specific domains. The different proposals represent their ontologies with formal languages usually employed in Artificial Intelligence and Semantic Web Project [68] for knowledge representation. For example, in [66][69] RDFS (Resource Description Framework Schema) is used, whilst in [5] the choice is DL (Description Logic). Recently VERL, an ontology representation language for describing events in video sequences [70][71], has been proposed as an initiative for dinamizing the use of ontologies in all the areas of video processing.

The solutions taken to solve the second problem are more heterogeneous in the approaches proposed. For example, the analysis process is specified in [69] by using a set of logic rules for inference expressed in F-Logic format. As an alternative, in [66] a genetic algorithm is proposed to find the optimum matching between the ontology entities and a set of regions initially extracted automatically from the images. Additionally, in [67] they use an ontology for defining a Dynamic Bayesian Network and they train it using some video sequences with the aim of identifying the entities defined in the ontology by using visual descriptors.

2.3 Feedback in computer vision systems

Nowadays, it is known that most of the basic steps of video signal processing (segmentation, tracking, recognition,...) are highly related, e.g., segmentation takes into account the movement continuity between consecutive frames (that is, tracking), and tracking takes into account some criteria about connectivity, compactness, aspect relation,etc., that definitely are more or less complex models of what we want to track (that is, what we want to recognize). Unfortunately, the single-pass strategy (or the use of the forward path, see Figure 2.5) of the hypothesis-verify paradigm becomes inadequate as it easily fails in case of not enough quality data due to bad initial data (noise, camera tampering,...), inadequate application of algorithms,.... For improving the high-level interpretations, one of the challenges is to exploit expectations derived from high-level (or mid-level) structures to improve mid or low-level processing (see Figure 2.5). The improvement of the relationships between processing steps or the use of a feedback path of partial results can be expected to improve overall performance as demonstrated in [72]. This work has been one of the first to demonstrate with specific experiments in the domain of urban traffic that high-level hypotheses about intended vehicle behavior could in fact be used to influence the tracking unit and thus improve tracking under occlusion.

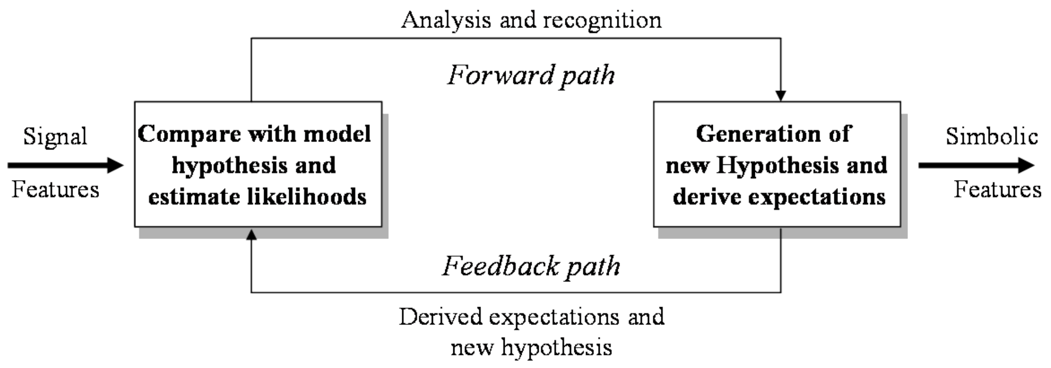

Analysis and recognition
Forward path
Signal
Features
Compare with model hypothesis and estimate likelihoods
Generation of new Hypothesis and derive expectations
Simbolic
Features
Feedback path
Derived expectations and new hypothesis

Figure 2.5: Representation of the forward and feedback path

In the literature work addressing expectation-guided image analysis using the forward path exists [56][73][74][75] but to our knowledge few video analysis systems with a generic architecture have been proposed which allow to feedback expectations from high levels of abstraction to image analysis procedures at low-level. In the past years, some methods like [76][77][3] can be interpreted as using high-level feedback to guide pixel-level foreground segmentation. These type of methods modify their pixel-level background modeling only in response to specific types of high-level processing, such as the detection of people or illumination changes. None of them describe how to generalize the feedback influence to arbitrary forms of high-level analysis.

In the last years, some approaches have emerged that describe in detail the feedback path. In [78][79][80][81], the authors extend the idea of the single-pass framework by employing complex feedback strategies for more robust hypothesis generation and verification. A generic control strategy of activities is presented and focused on the stages of the object detection and recognition tasks.

In the moving object segmentation stage, there is some relevant work. In [3], it is defined a generic framework for introducing feedback from high-level analysis modules into low-level stages. The foreground segmentation model is extended in a general way to incorporate corrective guidance from analysis concerned with higher level primitives such as image regions or object semantics. The benefits obtained are various: quick adaptation into the background model of sudden illumination changes, improvement of the segmentation of objects, and exclusion of repetitive movements. The proposed framework enables to use general-purpose pixel-level segmentation to the specific definition of "foreground" pertinent to an application. In [2] a feedback scheme is presented to couple with segmentation errors due to noise. Specifically, the model uses a decomposition strategy in description levels or models (for the objects expected to detect) to enable the feedback of information between adjacent levels. In this work, the detection of humans is proposed as a case study. The approach used at object level to re-feed the blob level is based on the decomposed model verification and it is translated, at blob level, into a parameter configuration that affects the segmentation process. The parameter optimization for segmentation of video objects is also discussed in [82] as a way of providing feedback to low-level analysis. The modules used in this proposal are an object filter and a genetic algorithm that learns the best parameter configuration accumulatively (based on the number of objects detected and some feedback information). More recently, [83] combines the use of feedback and low-level scene ontology to present a knowledge-based framework for video analysis which exploits relationships among analysis stages. The authors test the proposed framework in a foreground object extraction application, where temporal frame difference and background subtraction are used as inputs, and a basic low-level ontology suited for these inputs defines the classes in the scene description (at pixel level) and set constraints to their relationships.

The use of feedback in the object recognition process is discussed in [84]. The recognition task in variable illumination environments is modeled as an optimization problem with the quality of object recognition being the goal function. The feedback path is performed from the recognition phase to the detection phase varying the detection template. A multilevel Markov random field (MRF) is adopted to model both the detection and recognition processes. Then an optimization phase is performed using the MRF structure.

Nowadays, the incorporation of tracking analysis in complex feedback schemes is starting and some works can be found in the current literature. [85] calculates performance measure of tracking results that are used in a feedback loop to evaluate the goodness of the segmentation/tracking in order to adjust the weights assigned to each low-level analysis stage. [86] presents a statistically consistent method for incorporating feedback from high-level motion models (tracking analysis) to modify the adaption behavior (background model). The idea is to use knowledge from the models of the background and from the tracker to improve the overall performance. This share of both models is based on formulating the background maintenance problem as inference in a continuous state Hidden Markov Model, and combining it with a similarly formulated object tracker in a multi-chain graphical model framework. Another approach similar to the previous one is presented in [87]. In this paper a detection feedback mechanism is performed which deals with objects deposited/removed into/from the scene. This method avoids any slow moving or stopping objects being absorbed as well as the so-called "ghost effect" caused by adaptive background learning, as the feedback loop enables the selective update of the background.

Chapter 3

Ontology for semantic video analysis

3.1 Introduction

The proposed ontology defines all the knowledge needed to represent video events for automatic semantic description of video sequences. This knowledge involves the different objects that appear in the scene, the different events/actions that may happen, the system capabilities and the possible responses to the different events observed.

An ontology structure is used to design the concepts relative to video events. The ontology has been structured in two parts: the basic concepts and their domain-based extensions. The basic concepts define the common fundamentals for the creation of new domain ontologies for the extraction of semantic information from video sequences. Then, some extensions of the basic concepts (based on the knowledge of the application domain) are described. Finally, video surveillance is proposed as an application domain. The *protégé* software developed at Stanford University [88] has been used to build the ontology for semantic video description.

In the next sections, we briefly describe basic entities (section 3.2), their extensions (section 3.3) and the proposed domain ontology for visual surveillance (section 3.4).

3.2 Basic concepts/entities

This section enumerates and defines all the basic concepts of the ontology. In a generic video analysis system, we can suppose that the relevant concepts of the a priori knowledge belong to three classes: the *Scene* (or the outside), the *System* (or the inside) and the *User* (or the interface to the user). A more detailed description of these classes is given below:

- ***Scene***: the physical space where a real event occurs and which can be observed by one or several cameras. This concept includes the scene objects, their interactions (events) and the scene context. The scene entity has:
  - ✧ *Object List*: a list of objects in the scene. It is a spatial representation of the current spatial state of the scene. Basically, it covers the objects in the scene and their spatial descriptions. This list can include objects of any kind (mobile/contextual, foreground/background,...)
  - ✧ *Event List*: a list of events that happened in the past or currently being performed . The nature of these events can be very different: illumination changes, appearing/disappearing objects, activity monitoring,... It can be deduced that the objects from the Object List are implicated in these events, either as subjects of the action or as objects of it.
  - ✧ *Scene Context*: it is a priori information of the scene environment and it is acquired before the processing of the scene. It is composed of two kinds of contexts: spatial context and target context (divided into event and object context). The spatial context corresponds to a physical map of the scene. The mission context contains the specific methods to recognize the mission scenarios, the relations between them and other types of a priori information.
- ***System***: a concept for representing the system that uses the ontology. This concept includes the analysis capabilities of the system, the possible responses to the detected events and a system status. The system concept involves:
  - ✧ *System Capabilities*: a description of the system analysis capabilities and the associated parameters: available algorithms, configuration sets,...
  - ✧ *System Status*: a description of the current system status. Depending on the application, this description can have significant changes.
  - ✧ *System Reactions*: a description of the different system reactions to specific detected events or objects (like starting another application, recording the event to disk,...)
- ***User***: this concept represents the interface to the final user that manipulates the semantic information generated by the system. This “user concept” can

be a physical user, another software program,... and it can include a description of the user interaction mode to request information to the system (system information or semantic information extracted)

Although the previous scheme includes the *User* (or consumer information) as a fundamental part of the ontology, its modeling is out of the scope of the work presented in this document. Hereafter, we ignore the modeling of the *User* concept. A hierarchical description of the basic ontology concepts can be found in Figure 3.1 (*Scene* entity) and Figure 3.2 (*System* entity). These basic concepts are extended in the following sections.

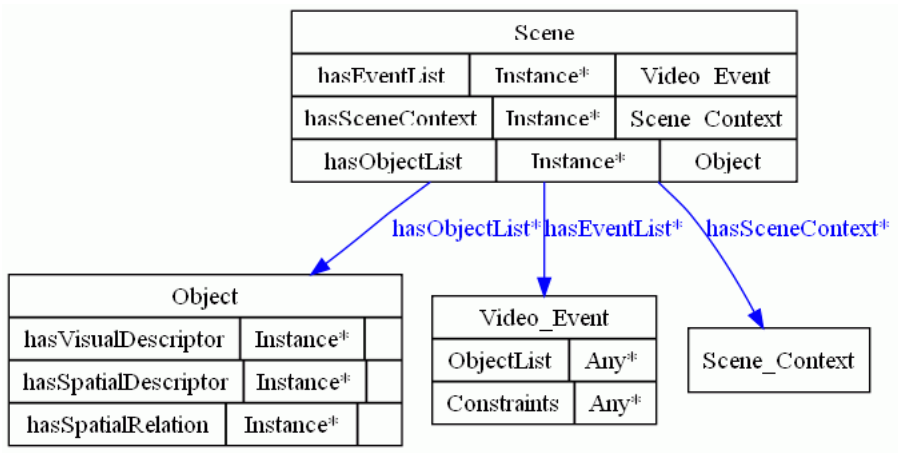


Figure 3.1: Class Hierarchy of the *Scene* entity

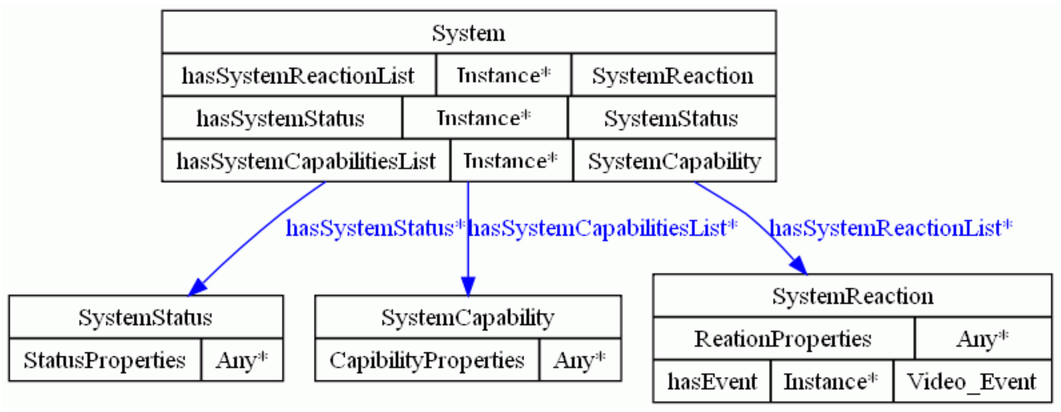


Figure 3.2: Class Hierarchy of *System* entity

3.3 Extensions of basic concepts/entities

3.3.1 Object

The *Object* concept represents any real world object observed by the camera. It is the basic structural unit in the scene and most of the ontology concepts (*SceneContext*, *Event*,...) rely on the *Object* entity. The class of an object corresponds to its nature and can be determined by its properties. The objects in the scene can be classified depending on two aspects:

- The ability to initiate their own motion. This property characterizes the mobility and autonomy of the object in general and divides the *object* entity into mobile and contextual objects:
  - ✧ **Mobile objects**: an object that can initiate its motion. Typical mobile objects are individuals, body parts, groups of people, animals, robots...
  - ✧ **Contextual Objects**: an object that cannot initiate its motion. Depending if the object is movable, we have two subclasses: *Fixed Objects* (if it cannot be displaced) and *Portable Objects* (if it can be displaced).
- The spatial dimensions used for representing the object. Depending on the model dimension we have two subclasses: 2D and 3D.

The proposed scheme for the *Object* entity is shown in Figure 3.3

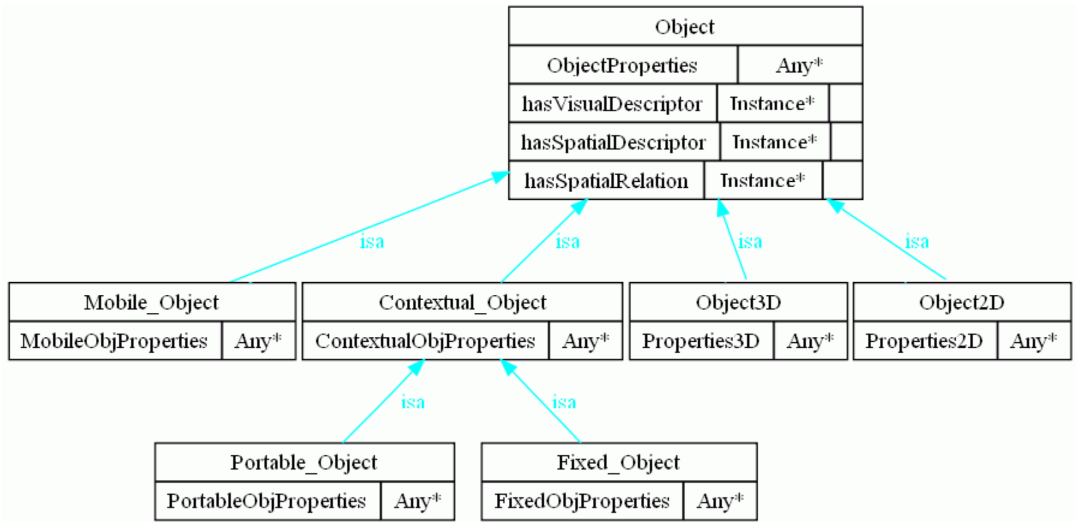


Figure 3.3: Class Hierarchy of *Object* entity

3.3.1.1 Basic Object Attributes

The attributes of an object are all the properties characterizing the *Object* entity. Between the different attributes associated to the *Object* entity, we highlight the following:

- Properties between objects (”inter-object”): they indicate the relationship between different objects and their spatial relation. They are:
    - ✧ “isPartOf” or “hasObject”: indicates that the object is part of another object or have another object as part of it (like fingers are part of a hand)
    - ✧ hasSpatialRelation: indicates the spatial relation between object (like a head is on the top of a human body)
- Properties for describing an object (“intra-object”): they describe the object. Their value belongs to basic types (like integer or string) and VisualDescriptors (using the MPEG-7 standard[89]):
    - ✧ Visual Attributes using basic types
        - ❑ Position-Based: Xpos, Ypos
        - ❑ Global-appearance: height, width, ratio, global_color, size
        - ❑ Local-appearance: silhouette, posture, sub-part_color
    - ✧ hasVisualDescriptor: indicates a MPEG-7 visual description

The different subclasses of the *Object* entity (*MobileObject* and *ContextualObject*) inherit the basic object attributes and they add other properties for characterizing its mobility and autonomy (like liveliness) or their role in the scene (like role). In Figure 3.4 we can see an example of a mobile object (a person) and some of its attributes (indicated in green colour) and its relations with other entities (indicated in blue colour).

3.3.2 Event

The *Event* entity is a generic term to describe any event, action or activity that happens in the scene and that is interesting for a video analysis system (e.g. video surveillance, video indexing). The semantic level of an event is very variable: from low-level events (like motion or illumination change), mid-level events (like appearing, splitting and other object-related events) or high-level events (gesture identification, activity monitoring,...). Video events are characterized by their objects of interest, the

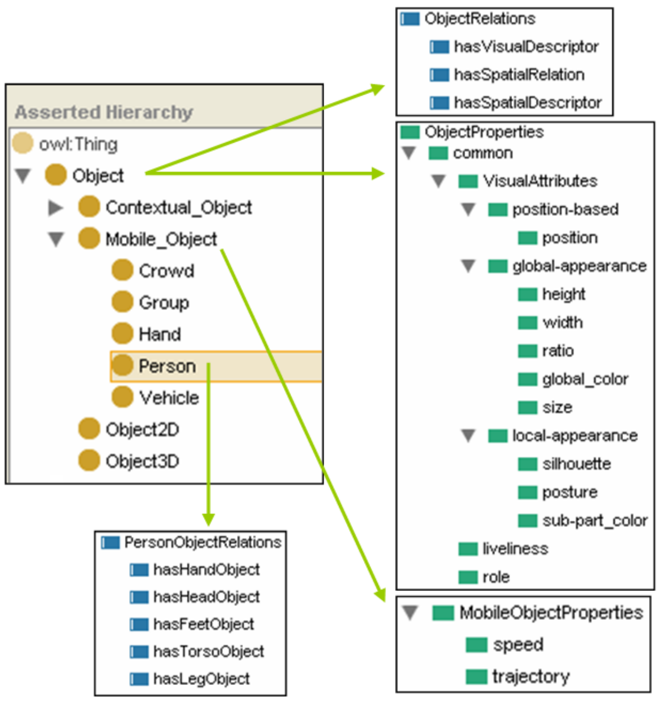


Figure 3.4: Ontology model of a "Person item"

time when the event happens and the capturing conditions (number of cameras used, daytime, weather conditions,...). Examples of typical events are "intrusion inside a restricted area", "detection of suspicious objects",...

There are different ways for distinguishing the kind of video events in the scene. In this work we propose three different aspects for classifying the video events:

- **Number of objects involved**

  ✧ *Multiple Events.* These type of events involve several mobile objects with different motions.

  ✧ *Single Events.* In these type of events only one mobile object performs the action to be recognized.

- **Temporal relation**

  ✧ *Simple Events.* These type of events can be calculated every frame. The properties or the likelihood of the event can be directly acquired by analyzing one single frame (or few frames).

  ✧ *Complex Events.* These type of events present temporal relation between the different parts that compose the event (sub-events). The events occur during a period of time and cannot be calculated by only analyzing few frames.

- **Transitivity**
  - ✧ *Intransitive.* These type of events require a subject that performs the action but it is not required an object for? the action (purpose?)
  - ✧ *Transitive.* These type of events require a subject and a purpose of the action.

In the rest of the chapter, classification of the events based on the transitivity is avoided. This classification only adds the features *action_subject* and *action_purpose* to all the derived classes of the *Event* entity. This fact duplicates all the classes to describe and unnecessarily extends the chapter length (see Appendix A for a complete description of the proposed ontology). Thus combining the two other classification schemes (number of objects and temporal relation) we can classify the video events into four classes:

- *Simple_SingleObject* events (or SSE): these type of events are performed by one mobile object and can be directly inferred from the visual attributes of the mobile object (numerical values). These type of events usually correspond to general physical object properties. For example: “A person stays inside a zone”.
- *Simple_MultipleObject* events (or SME): these type of events are performed by several (at least two) mobile objects and can be calculated every frame. For example: “Two person stay inside a zone” or “Person A is close to object O and person B stay inside zone Z”
- *Complex_SingleObject* events (or CSE): these type of events are a linear combination in time of simple events (directly calculated in each frame) and they are performed by one mobile object. These type of events imply a temporal relation and order between the sub-events that compose the event. For example: “detection of an unattended bag” (composed of the sub-events calculated in different frames: “drop-off object”, “object becomes static” and “owner distance too far”).
- *Complex_MultipleObject* events (or CME): these type of events are the most difficult events to detect. They involve several mobile objects and they are composed of different sub-events with logical and time relations between them.

The proposed scheme for the event entity is shown in the Figure 3.5.

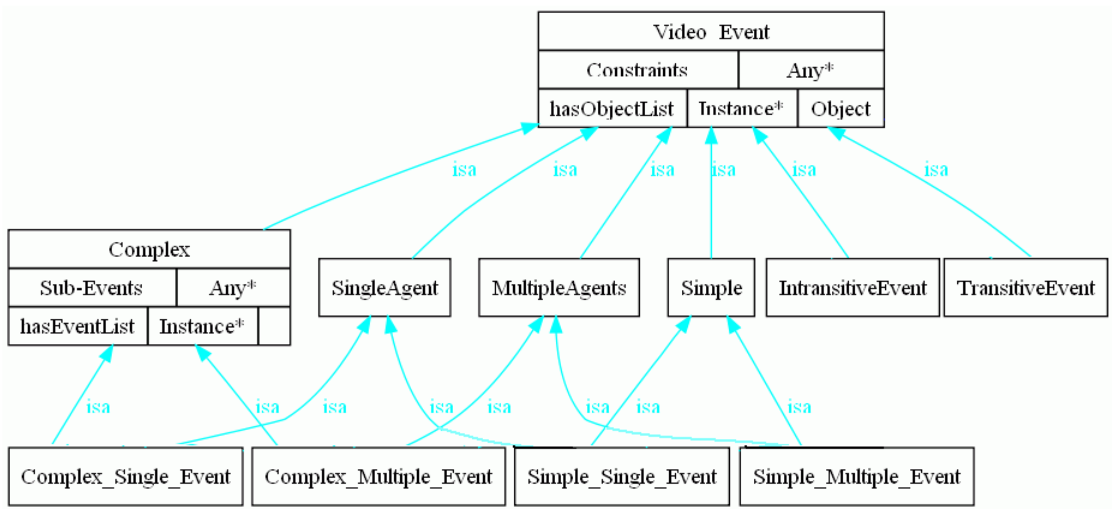


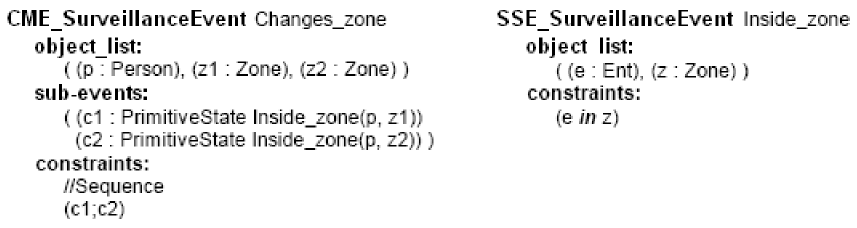
Figure 3.5: Class Hierarchy of *Event* entity

3.3.2.1 Basic Event Attributes

The attributes of an event are all the properties characterizing the event entity. Between the different attributes associated to the event entity, we highlight the following:

- List of objects (mobile and contextual) that perform the action (*ObjectList*)
- Sub-events that compose the event to detect (*Sub-events*)
- Relations or constraints between the objects and the sub-events or events (*Constraints*)

In the following Figure 3.6, we can see an example of the basic event attributes used for a *Complex_SingleObject* event ("Changes_zone") and for a *Simple_SingleObject* event ("Inside_zone").



```
CME_SurveillanceEvent Changes_zone
   object_list:
      ( (p : Person), (z1 : Zone), (z2 : Zone) )
   sub-events:
      ( (c1 : PrimitiveState Inside_zone(p, z1))
        (c2 : PrimitiveState Inside_zone(p, z2)) )
   constraints:
      //Sequence
      (c1;c2)
```

```
SSE_SurveillanceEvent Inside_zone
   object list:
      ( (e : Ent), (z : Zone) )
   constraints:
      (e in z)
```

Figure 3.6: Event Attributes Example

3.3.3 Scene Context

The *SceneContext* entity defines all the information that may influence the way a scene is perceived. This information can be used during the analysis of the scene to help the process to complete the task efficiently. In this context, we distinguish three types of a priori useful information:

- *Spatial Context*: it includes a 2D/3D spatial description of the objects of interest in the scene (mainly fixed and portable objects).
- *Object Context*: it includes the relations between the different objects (mobile and contextual) that can appear in the scene. For example, in video surveillance sequences from airports it is common that people walk with their luggage. In other scenarios, like parkings, it is common that people get out/into the car.
- *Event Context:* it includes the relationship between the events that can occur in the scene. This relationship is composed of two parts: the more likely events and combinations among the events. For example, the event context related to airport surveillance can include the more common events (like "unattended luggage detection") and some typical combinations (like the event "a Boarding gate door is open" is usually followed by "People passing through the door").

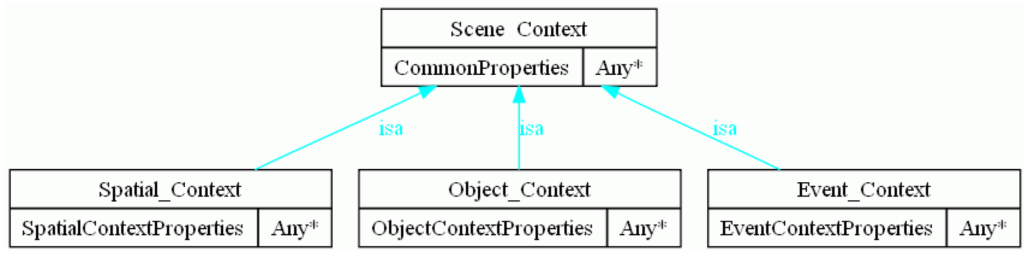


Figure 3.7: Class Hierarchy of *Context* entity

3.4 Domain ontology

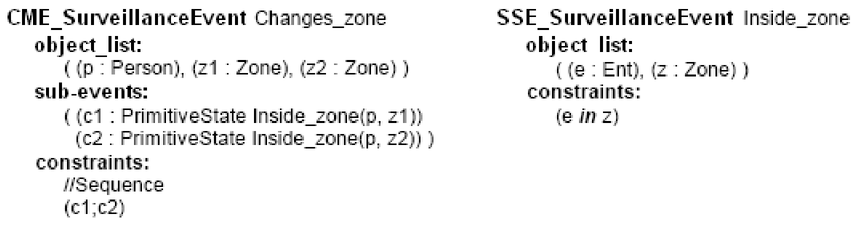
A domain ontology is targeted to model the subset of the world covered by a specific application. This work is focused on the use of the ontology in the analysis of events in video sequences. However, due to the extremely variety of meanings in which the video event detection domain can be conceived, one sub-domain has been selected to model them in detail: video-surveillance of underground stations.

3.4.1 Underground Video surveillance

For the Underground Video-Surveillance domain, the *Object* concept definitions are the following:

- Mobile Objects: Person (p), Group of persons (g), Crowd (c), Metro Train (m), Other (o)
- Contextual Objects
  - ✧ Portable Objects
    - ❑ Luggage (l), Portable furniture (f), generic(g).
  - ✧ Fixed Objects:
    - ❑ Zone (z) with different roles (Entrance_Zone, Exit_Zone,Corridor, Hall,...)
    - ❑ Equipment (eq) with different sub-classes (Wall ,Seat, Poster, Door, Map,...)

For the *Event* concept, Figure 3.8 shows some descriptions of the events and in Figure3.9 there is a list with all the events modeled in the ontology for the Underground Video-surveillance domain. Moreover, a more detailed description of the domain ontology can be seen in section A of the appendix.



```
CME_SurveillanceEvent Changes_zone
   object_list:
      ( (p : Person), (z1 : Zone), (z2 : Zone) )
   sub-events:
      ( (c1 : PrimitiveState Inside_zone(p, z1))
        (c2 : PrimitiveState Inside_zone(p, z2)) )
   constraints:
      //Sequence
      (c1;c2)
```

```
SSE_SurveillanceEvent Inside_zone
   object list:
      ( (e : Ent), (z : Zone) )
   constraints:
      (e in z)
```

Figure 3.8: Underground Video Surveillance Event Examples

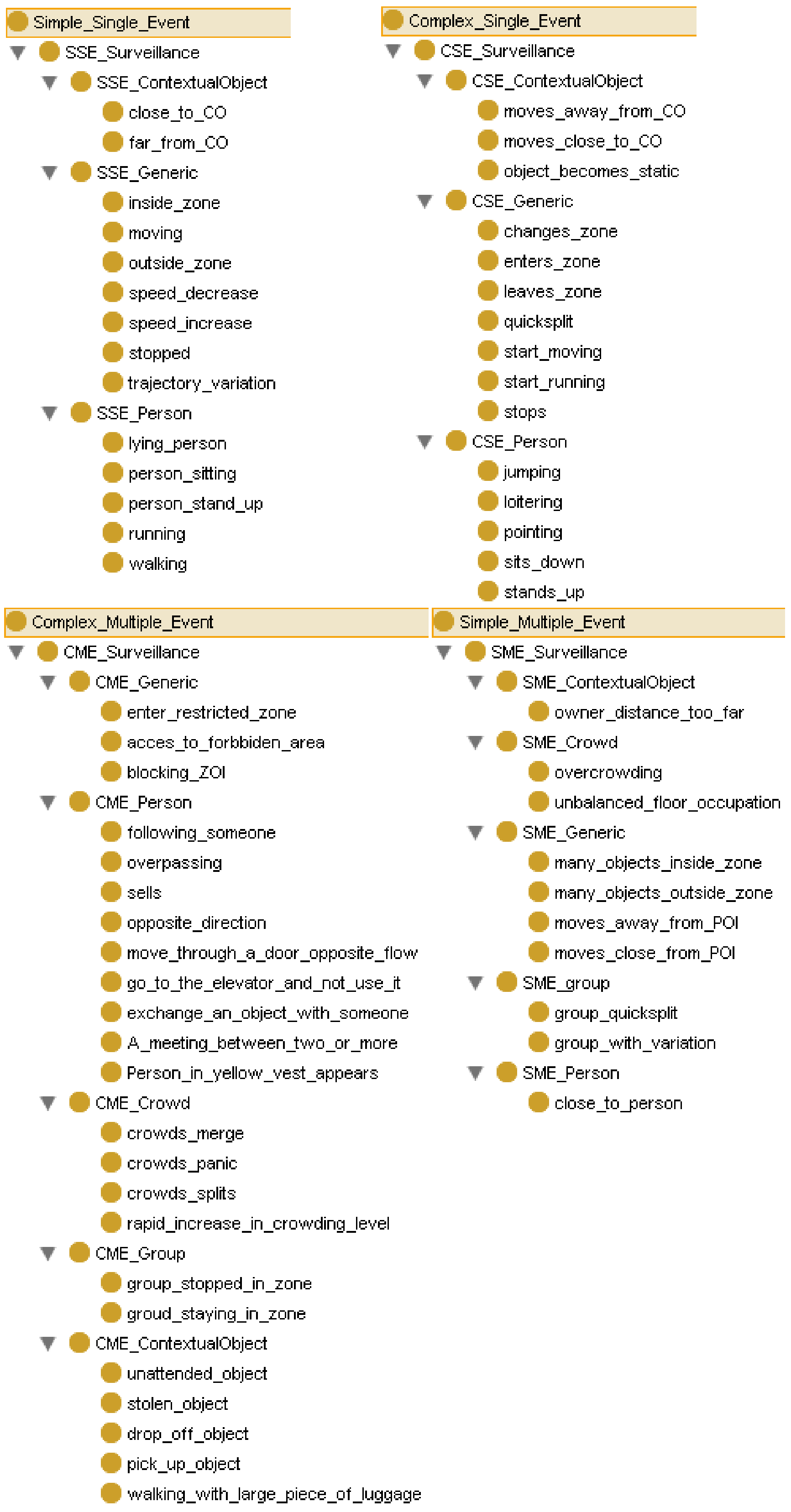


Figure 3.9: Event Ontology for the Underground Video Surveillance domain

Chapter 4

Video Analysis Framework

4.1 Introduction

The proposed video analysis system is depicted in Figure 4.1. After a frame acquisition stage, motion segmentation is performed to detect moving pixels (Foreground Detection Module). Subsequently, the Blob Extraction Module analyzes the connected regions of the binary foreground mask to detect blobs. Then, the Blob Tracking Module generates the trajectories of the blobs between consecutive frames using color and position information. Afterwards a classification stage is applied to the blobs for distinguishing between human and non-human classes. Then, all the generated data and some additional blob features (like type, trajectory, size, speed,...) are used in an event recognition stage (Action or Event Detection Module).

The possible use of this system is two-fold: real-time alarm generation by pre-defining event models like "A human moving in direction d at speed more than s causes alarm a_1" or make use of the produced video object data to index the stored video for offline semantic search. Additionally, the context information (*SceneContext* entity described the previous chapter) is used in the different processing stages to achieve their different objectives (like detecting the objects or zones of interest, identifying the events in the video,...).

This system is assumed to work real time as part of a video-based surveillance system. It implies that the computational complexity of the processing algorithms should not be high. Hence, the decisions on selecting the analysis algorithms are affected by their computational run time performance as well as quality of their outputs.

As the system relies on the background subtraction technique to detect foreground objects, it imposes restrictions in the type of camera and the initialization method.

Regarding to the type of camera, it must be fixed and should not have Automatic Gain Control (AGC) for obtaining a good extraction of video objects. Regarding to the position of the camera, moving objects must be not too close to the camera because the events to be detected need to have the entire object in the scene. Regarding to background initialization, we suppose that there are few starting frames of the video sequence with no-presence of objects to allow the computation of the background model . Another restriction is the need to filter sudden lighting effects, as the proposed system is not tolerant to them and may produce unexpected results (can be corrected with a module that detect these effects).

The remainder of this chapter presents the computational models and methods that have been selected for the different processing stages: frame acquisition (section 4.2), foreground detection (section 4.3), blob extraction (section 4.4), blob tracking (section 4.5), blob classification (section 4.6) and event detection (section 4.7).

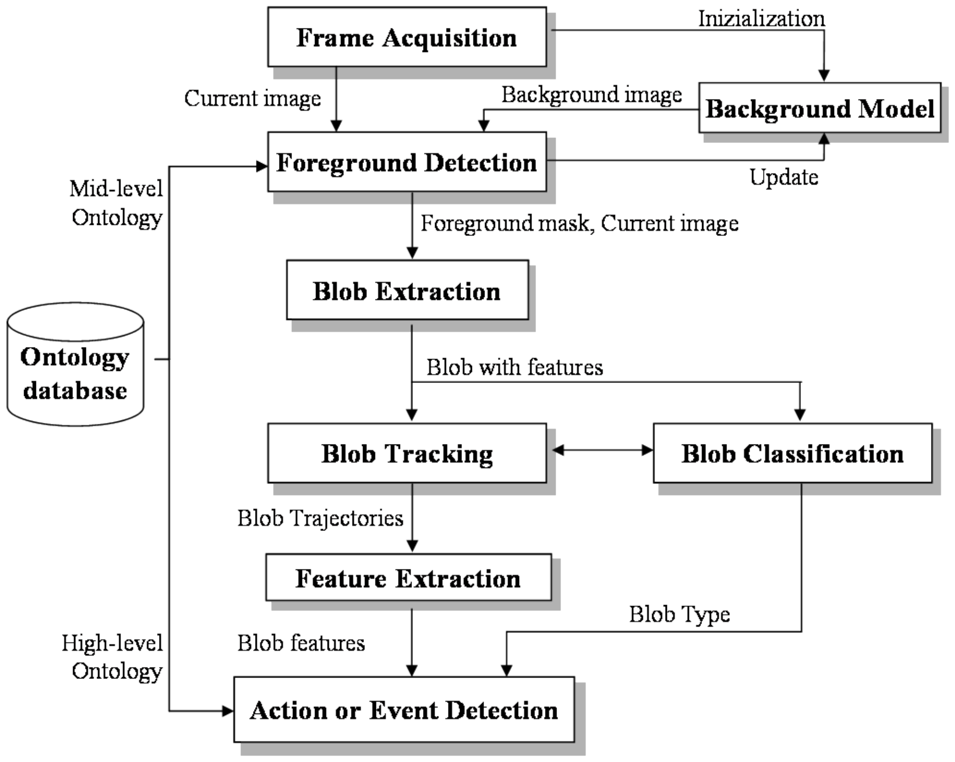


Figure 4.1: Video Analysis Framework

4.2 Frame acquisition

A frame sequence captured by a still camera or a stored video sequence are the inputs to the system. The proposed system works with uncompressed frames with RGB values for each pixel, therefore the input sequence may need some decoding before starting the processing.

The frame acquisition process is the following: the camera captures a frame from the recorded scene and sends it to the system; a specific software for the acquisition of frames is provided in the system and it is responsible for storing the data in a video memory [90]. On the other hand, if the frame source is a stored video sequence, it is sequentially scanned and the frames are stored in memory in the same way as the camera input by the acquisition module. Then the system takes the frames from memory and process them sequentially. The analysis of stored video sequences is done sequentially (without losing frames) and the analysis of frame sequences captured by the camera is done on demand (it may lose frames if the overall processing is too slow).

4.3 Foreground Detection

The system diagram of the foreground detection method is shown in Figure 4.2 and is based on the background subtraction algorithm. This method has been selected due to its low computational cost and the use of static cameras that allows to easily maintain a background model. The module is composed of four stages to extract binary masks indicating which pixels belong to the foreground. The first step is the change detection module. It decides whether each pixel is foreground (corresponding to a blob) or background. Then, noise removal is performed in the binary mask and the detection of shadow pixels is performed. Finally, noise removal is applied again to obtain the final foreground mask.

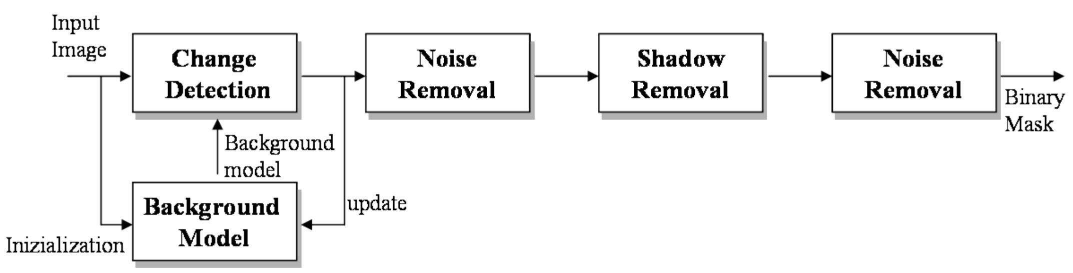


Figure 4.2: Foreground Detection Module

4.3.1 Change Detection

4.3.1.1 Background subtraction

In this module, the background subtraction algorithm is implemented inspired by the study presented in [75]. This algorithm works on gray-scale images with static background (see Appendix C for more information about color conversion). The background/foreground decision is taken by applying to every pixel a modified version of the basic background subtraction thresholding operation (see equation 2.1):

$$F(I[x,y]) \Longleftrightarrow (|I[x,y] - B[x,y]|)^2 > \beta \tag{4.1}$$

As we can see in Figure 4.3, the results of the application of equation 4.1 can produce noisy results mainly due to the noise introduced by the camera.

One of the possible solutions is to subtract a square window around every pixel instead of doing a single subtraction operation.

$$F(I[x,y]) \Longleftrightarrow \sum_{i=-W}^{W} \sum_{j=-W}^{W} (|I[x+i,y+j] - B[x+i,y+j]|)^2 > \beta \tag{4.2}$$

This thresholding operation aims at discarding the effect of the camera noise after frame differencing. In this point, the determination of the probability that the pixel difference at a given position belongs to foreground or background is due to noise and it can be calculated by using the Gamma and the Lower Incomplete Gamma functions (see appendix E for more information). Firstly, it is supposed that there is no moving object in the frame difference and we refer to this hypothesis as the null hypothesis, H_0. Let $g(i,j)$ be the sum of the absolute values of the frame difference in an observation window of q pixels around (i,j)(equation 4.2). Moreover, let us assume that the camera noise σ is additive and follows a Gaussian distribution with variance σ. Given H_0, the conditional probability density function of the frame difference

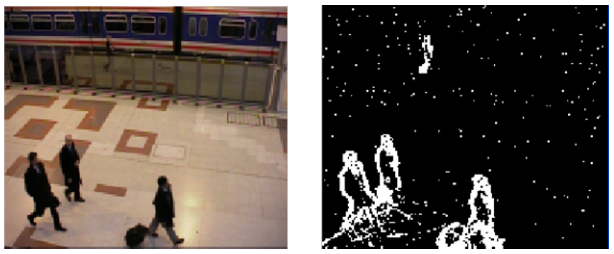


Figure 4.3: Obtained Foreground Mask with the modified basic background subtraction algorithm

follows a distribution $\chi_q^2$with q degrees of freedom defined by the following equation:

$$f(g(i,j)/H_0) = \frac{1}{2^{q/2}\sigma^q\Gamma(\frac{q}{2})} g(i,j)^{(q-2)/2} e^{-g(i,j)^2/2\sigma^2} \tag{4.3}$$

where $\Gamma(\bullet)$ is the Gamma function. Now we can formulate a significance test using the previous assumption for accepting or rejecting the null hypothesis H_0(see Appendix E for more information). This test is:

$$P\{g(i,j) \geq \tau(i,j) \parallel H_0\} = \frac{\Gamma(\frac{q}{2}, \frac{g(i,j)^2}{2\sigma^2})}{\Gamma(\frac{q}{2})} \tag{4.4}$$

This test indicates that if the probability is smaller than a certain value α, we reject the null hypothesis at the current pixel position. Therefore, we label that pixel as belonging to a moving object. The significance level is a stable parameter that does not need manual tuning along a sequence or for different sequences. Experimental results indicate that valid values fall in the range from 10^{-2} to 10^{-6}. Thus, the most critical parameter in this detection process is the variance of the camera noise σ. The main advantage of this technique is that it can compensate a video signal with a time-varying noise level.

4.3.1.2 Background model

In the proposed foreground detection module, the reference background B is initialized with the first video image. To adapt the background model to the dynamic changes of the environment like global illumination changes, moving clouds in a sunny day and long term background updates (objects that become part of the background), the reference background is updated selectively with incoming images. The update process is performed using a running average scheme (see equation 2.3 for more information) only in the pixels detected as background by the change detection module and maintaining the same pixel value in the pixels detected as foreground by the change detection module. The update scheme is:

$$B_{t+1}(x,y) = \begin{cases} \alpha B_t(x,y) + (1-\alpha) I_t(x,y) & if\ F_t(x,y) = 1\,(FG) \\ B_t(x,y) & if\ F_t(x,y) = 0\,(BG) \end{cases} \tag{4.5}$$

4.3.2 Shadow removal

During the segmentation of the objects from the background, moving cast shadows are always misclassified as part of the moving object. Practically any scene, both indoor and outdoor, contains shadows. For general-purpose shadow detection, the pixel based and deterministic non-model based algorithms assures the best results [23]. Other studies, like [91], suggest that one of the best color space for detecting shadows is the HSV space and this color space corresponds closely to the human perception of color [92]. In the proposed system, we have selected an algorithm that combines these two facts. The algorithm used is based on the work presented in [17] and it performs an analysis of the chromaticity and intensity variation of the current and background images in the HSV space.

After the color conversion between the input RGB images to the HSV space, three maps are calculated to detect shadow pixels in the foreground mask previously calculated.

Firstly, the ratio intensity (V) between current and background images is used to filter the possible shadow points. Then, the algorithm exploits that the variation of the chromaticity (Hue (H) and Saturation (S)) between current and background images does not change significantly [23]. The rules that decide if a pixel belongs to a shadow are described in the following equation:

$$SP_t(x,y) = \begin{cases} 1 & if\ \alpha \le \frac{I_t^V}{B_t^V} \ge \beta \\ & \wedge \|I_t^S(x,y) - B_t^S(x,y)\| \le \tau_S \\ & \wedge \|I_t^H(x,y) - B_t^H(x,y)\| \le \tau_H \\ 0 & otherwise \end{cases} \tag{4.6}$$

where $I_t(x,y)$ and $B_t(x,y)$ are the pixel values at coordinate (x,y) in the current input image and in the background model at time t. Finally pixels classified as shadow points are eliminated from the foreground mask.

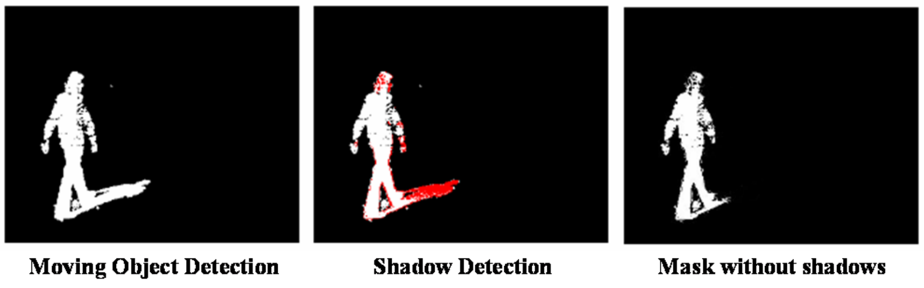


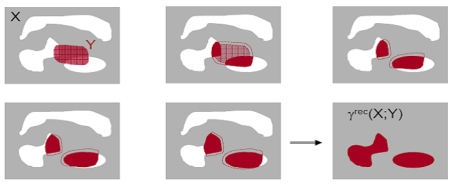
Figure 4.4: Example of shadow removal using the HSV color space

4.3.3 Noise removal

Noise reduction of the binary foreground mask is performed using mathematical morphology. The operation selected is called “Opening by Reconstruction of Erosion” and it is described in [31]. It is an smart combination of the basic erosion and dilation morphological operations. The main advantage of this operation is that it preserves the underlying shape of the object eliminating small artifacts detected in the foreground mask. The size of the small artifacts eliminated (usually noise) depends on the size of the structuring element used in the dilation/erosion operations. The algorithm works as follows:

Algorithm 2 Opening by Reconstruction of Erosion

- The algorithm starts with an image X with some regions to eliminate.
- Then a marker image Y, that indicates some portions of the regions that do no be eliminated, is calculated by an erosion process of X.
- After that, an iterative process is performed:
    - ✧ A dilation operation is calculated in the image Y. Then, the pixels of the dilated image different to image X are eliminated.
    - ✧ The stop condition: when there is no change in the marker image Y (after the dilation and filtering processes) between two consecutive iterations



The implementation of this operation in the proposed video analysis system uses as marker image Y which is an eroded version of the current binary foreground mask (image X). After the dilation process, logical AND operation between X and the eroded Y is calculated to filter the different pixels in the two images. Finally the stop condition has been defined as the minimum percentage change in the final image Y obtained after the dilation and filtering processes. In Figure 4.5 we can see the results of applying the operation in one binary mask and an overall reduction of the noise can be observed.

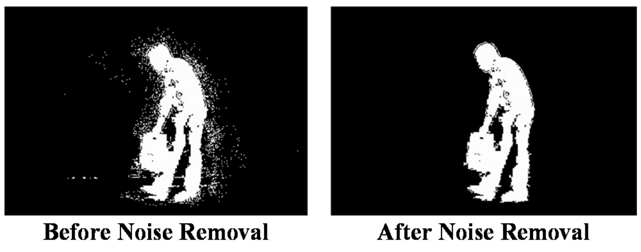


Figure 4.5: Example of noise removal in binary masks using "Opening by Reconstruction of Erosion"

4.4 Blob Extraction

After the extraction and post-processing of binary foreground masks for filtering noise and shadow regions, the Blob Extraction module extracts information about connected components in the binary foreground mask. This processing is performed in two stages: an initial blob detection stage and a post-processing stage.

4.4.1 Blob Detection

The objective of this stage is the extraction of connected components in the binary foreground mask. The technique employed at this stage is the Connected Component Algorithm (CCA) using 8-neighborhood. This algorithm is described in section 2.1.1.3. It assigns a label to each connected region in the mask. Additionally, bounding boxes of these regions are calculated. Figure 4.6 shows an example of this process.

4.4.2 Region Level Post-Processing

After the blob detection performed on the foreground mask, we can notice (see Figure 4.6) that some artificial small regions are detected and labeled. These regions appear even after removing pixel-level noise. This problem can be fixed by eliminating regions with an area lower than a certain threshold (in terms of numbers of pixels). For each frame, this threshold is calculated as a percentage of the average region size and regions that have smaller sizes than the threshold are eliminated from the list of regions detected and the foreground mask.

4.5 Blob Tracking

This module makes a simple tracking of the relevant objects identified in the previous modules. This task is performed by finding the correspondences between the blobs detected in the current frame and the blobs detected in the previous frames (tracks).

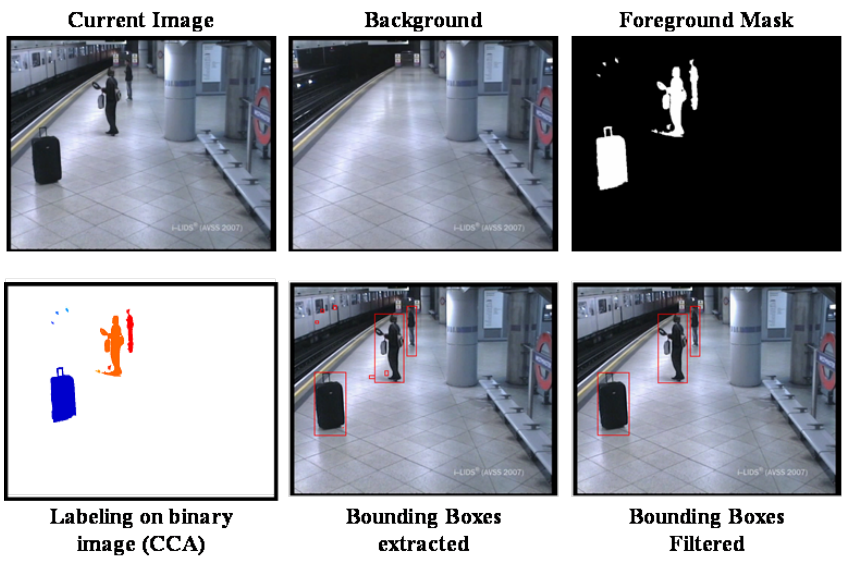


Figure 4.6: Example of the Connected Component Analysis with post-processing

We have decided to use a Match Matrix to find the correspondence between tracks of previous frame (namely T^m_{j-1}) and blobs of current frame (namely B^n_j) simplifying the tracking problem. This Match Matrix is calculated for the tracks and the blobs, and the best matches are calculated from the Match Matrix. Each row in the Match Matrix corresponds to a track from the previous frame and each column corresponds to a blob from the current frame. Each element in the matrix is, thus, the degree of match between a track and a blob. The values entered in the matrix are the Euclidean distance between a track and a blob. In this system, each element in the match matrix is the sum of the Euclidean distance in position values (X and Y coordinate values) and the Euclidean distance in color values (R, G, and B values). The values for X and Y values are normalized, respectively, against the maximum width and height of the images. Similarly, R, G, and B values are normalized against their maximum possible value, which is 255. The definition of the Match Matrix for each position is the following:

$$\begin{aligned} MM_{nm} = &\sqrt{(\Delta Y/Ydim)^2 + (\Delta X/Xdim)^2} + \\ &\sqrt{(\Delta R/255)^2 + (\Delta G/255)^2 + (\Delta B/255)^2} \end{aligned} \tag{4.7}$$

where ΔX and ΔY are the differences in X and Y position values and ΔR,ΔG and ΔB are the differences in mean red/green/blue values between track m of the previous frame and blob n of the current frame. $Xdim$ and $Ydim$ are the frame dimensions.

4.6 Blob Classification

The objective of this module is the identification of the different objects that can appear in the video sequence. As described in section 2.1.3, the discrimination between people and other objects is a key issue in the detection of events. For classifying blobs detected as people and non-people, we have used a combination of two simple algorithms:

- The first algorithm is based on a single feature, the aspect ratio of the blob. Thus, the feature computed is w/h and we assume that it follows a Gaussian distribution with $\mu = 0.3$ and standard deviation $\sigma = 0.2$ (computed from the training set).
- The second algorithm is based on the compactness on the foreground detected regions. If their corresponding area in the binary foreground mask is filled less than some percentage, p, then the item is classified as people. The threshold p is empirically found to be between 70% and 75%.

These two algorithms are combined in a conventional Bayes classifier linear scheme[56]. Figure 4.7 shows an example of the classification results for different foreground objects detected:

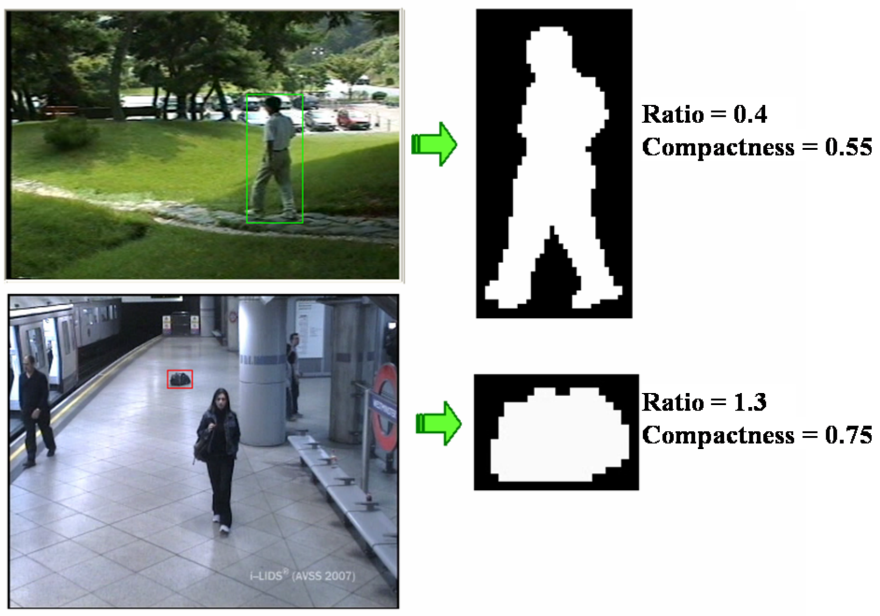


Figure 4.7: People Detection results sample (green/red colour for blobs detected as people/non-people)

4.7 Event Detection

As the event detection process is directly related with the ontology definition of the events, we have to process in a different way each event concept shown in the ontology. In this section the recognition mechanism for detecting simple events (that can be calculated in each frame or few frames) and complex events (that are composed of sub-events and present relation with time) are shown.

Figure 4.8 depicts the scheme of the event detection module. Firstly, this module calculates some properties needed to recognize the different events (the blob categorization is not performed in this stage). Then, the information calculated by the previous modules is analyzed to detect the events. The events to be recognized and some objects/zones of interest are provided by the scene context.

4.7.1 Detection of simple events

These type of events are defined over a short coherent unit in time ($\sim 1 - 25$ frames) and commonly involve few objects. The definition of these events can be directly inferred from logical constraints on other simple sub-events or from mobile object properties. Also a successful event model has to handle the ambiguity in the event definition (using the probabilistic domain instead of the logical one).

Due to this considerations, these type of events can be seen as an instantiation of a Bayesian Network like the authors of [56][93] suggest. The events are considered as hypotheses and the related properties as evidences. Bayesian inference allows to calculate the probability of hypothesis H ($\sim H$ means the opposite of the hypothesis) by analyzing the related evidences E_i using the following equation (assuming that each evidence is conditionally independent given the hypothesis):

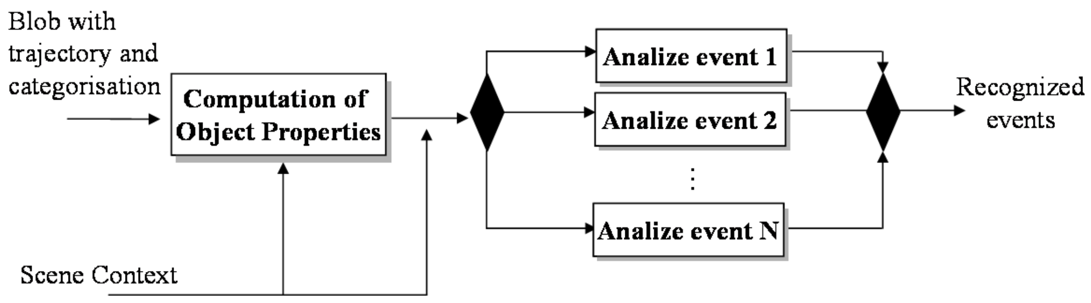


Figure 4.8: Event Detection Module

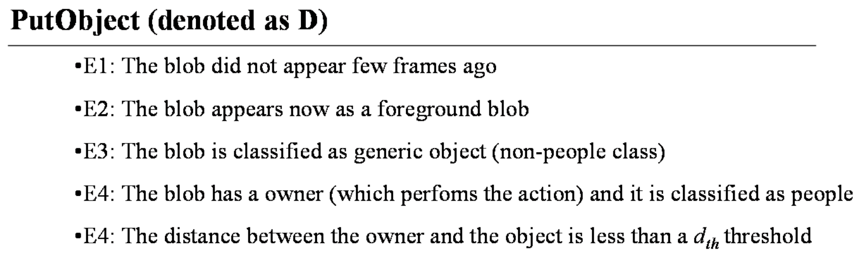
$$P(H/E_1....E_n) = \frac{\prod_{i=1}^{n} P(E_i/H)P(H)}{\prod_{i=1}^{n} P(E_i/H)P(H) + \prod_{i=1}^{n} P(E_i/\sim H)P(\sim H)} \tag{4.8}$$

To model the probability distribution function of the terms $P(E_i/H)$ we suppose four different functions: Threshold, Uniform, Gaussian and Histogram. The parameters of these distributions are learned from training data.

As a example of this model for detecting simple events we provide the definition of the *PutObject* event based on the proposal of [56] with some modifications. In Figure 4.9 we can see the model of this event.

E_1 and E_2 impose together a temporal constraint: the object was carried by the human before the *PutObject* event so the separate track of the object has not started yet. Once it appears, we know the *PutObject* event has just happened and we check if the two blobs involved correspond to an object and the owner classes (evidences E_3 and E_4). $E_5$is a spatial constraint to check the distance between the owner and the object (the owner has to be close to the object when the drop-off just happened). This constraint eliminates irrelevant persons who are just passing by when *PutObject* event takes place. In case that multiple persons are close to the luggage when the drop-off takes place, the closest person is considered as the owner. The prior and the conditional probabilities are listed as follows:

- $P(D) = P(\sim D) = 0.5$
- $P(E_1/D) = 1$,
- $P(E_2/D)$ =Probability of the blob of being foreground blob
- $P(E_3/D)$ =Probability of the object of being non-people class
- $P(E_4/D)$ =Probability of the owner of being people class



PutObject (denoted as D)

•E1: The blob did not appear few frames ago

•E2: The blob appears now as a foreground blob

•E3: The blob is classified as generic object (non-people class)

•E4: The blob has a owner (which perfoms the action) and it is classified as people

•E4: The distance between the owner and the object is less than a d_{th} threshold

Figure 4.9: *PutObject* event modeled using Bayesian Inference

- $P(E_5/D) = \begin{cases} 1 & if\ d < d_{th} \\ 0 & if\ d > d_{th} \end{cases}$
- $P(E_i/ \sim D) = 0.5,\ i = 1, 2, 3{,}4{,}5$ (when $P(E_i/ \sim D)$is unknown or hard to estimate, we use a default value 0.5)

4.7.2 Detection of complex events

These type of events correspond to more complex structures of events. These events may take place over long sequences and present some linear combination in time. As [94] we have decided to use finite-state automata, FSA (or finite-state machine, FSM) to represent this type of events. This FSA structure allows to easily represent the linear temporal relation of sub-events that compose the complex event to detect. In Figure 4.10 we can see an example of the complex event "Detection of abandoned object" composed of the states (or sub-events) *PutObject*, *Object incorporated to the background model* and *Distance to owner too far*. The transitions between consecutive states occur when the state i gets high probability and returns to the initial state in the other case.

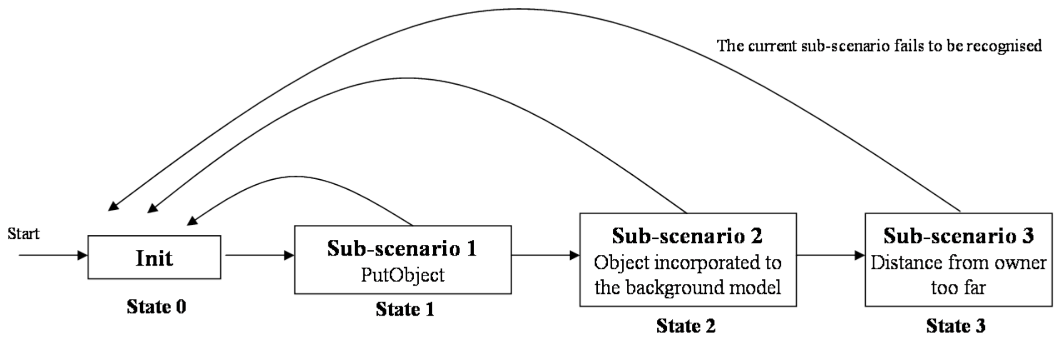

The current sub-scenario fails to be recognised
Start
Init
State 0
Sub-scenario 1
PutObject
State 1
Sub-scenario 2
Object incorporated to the background model
State 2
Sub-scenario 3
Distance from owner too far
State 3

Figure 4.10: Definition of "Abandoned Object" Complex event using a finite-state machine (FSM).

Chapter 5

Introducing feedback

5.1 Introduction

In this chapter we describe the feedback strategies introduced in the proposed video analysis framework. By using feedback strategies we want to:

- improve the performance of the lower processing levels and subsequently to improve the performance of the final results of the system (e.g. events detected)
- evaluate more complex rules or constraints for searching additional information in the object of interest (e.g. finding non-detected parts) in order to reject or accept unknown hypothesis (that is, hypothesis that were in doubt).
- efficiently use the resources without reducing the performance of the final results (in this context, resources can be seen as available processing algorithms)
- modify the parameters of the algorithms for improving performance in future runs in the video sequence analysis (e.g., select the people detection algorithms that produce best results for a video sequence).

We will follow the feedback strategy suggested by several authors [79][80][82], where the generated hypothesis is accepted or rejected by computing the error signal between the hypothesis and the model to be recognized. A new hypothesis is generated if the hypothesis under evaluation is rejected and the new corresponding error signal is computed. This process is repeated until the new hypothesis is accepted. Furthermore, [3] suggests that the feedback strategies can guide the analysis component behavior for future analysis of incoming images. The typical behavior modification performed is parameter adjustment[3][2].

In this work, we use the feedback strategies for evaluating unknown hypothesis and modify the analysis component's behavior for future runs. We interpret the feedback concept as a re-evaluation of some processing stages with a modification on their quality output level. This output quality variation is performed by selecting the desired quality level (a number between 1 and $MAXQlevel$ that is predefined for each module). In each module, the predefined quality levels are automatically mapped into a parameter adjustment or other strategies that change the quality of the output. At this point, we have three issues to solve:

- Which module decides which component has to change its behavior (that is, select the appropriate quality level) and which components need to be re-executed?
- How to implement the output quality levels of the processing stages?
- How to measure the quality of the output data?

The first question can be solved by adding a supervisor module that manages the behavior of the video analysis system and decides if the hypotheses generated are accepted or rejected. If the hypotheses are rejected, this supervisor module is responsible for deciding which analysis components need to be re-executed and the new quality requirements in their output results. These decisions are taken by analyzing the intermediate output results of the analysis components involved in the generated hypotheses. The second and third questions are difficult to answer in a general way due to the output results variety and they will be answered in each particular case.

In the next sections, we briefly describe the feedback approach used in the framework (section 5.2) and their application in the proposed framework (section 5.3). This application is described in terms of the processing stages with output quality variation (sections 5.3.1.1 to 5.3.1.5) and the supervisor module (section 5.3.2). Finally the use of the feedback path (section 5.3.4) and the model modification for future runs on incoming data (section 5.3.5) are discussed.

5.2 Feedback as output quality variation and selective re-execution

As said above, we apply the feedback strategies to change the behavior of the analysis components as an increase/decrease in the quality level of their output results and re-execute them. By using this strategy we want to increase the quality of the final results of the system, that is, to improve the confidence in the semantic description generated (e.g. the events detected) at the same time that we keep the computational cost bounded.

As a first approach to this interpretation of feedback, we distinguish four ways to automatically change the quality of the output results:

- **By using a ROI-based multi-resolution analysis**. This approach allows to process the input data with different granularity on the regions of interest obtained in the lowest resolution level. Furthermore, it reduces the overall analysis effort maintaining a similar output quality performance. Thus, the quality levels of this type of analysis can be associated to the granularity of the analysis.

- **By using an iterative analysis on data**. This approach exploits the iterative data analysis performed by some algorithms. This process usually improves the analysis in each stage and it is stopped by some predefined criteria. Thus, the quality levels of this type of components can be viewed as a modification of this stopping criteria and the parameters involved in the iteration process.

- **By using independent analysis components to perform the same task and fuse their results**. In this approach, we propose to use the agreement between them as the measure of the quality output[95]. Furthermore, it is a key issue to use independent and simple analysis components to exploit the Maximization of Mutual Information concept (MMI)[96]. Thus, the quality level of the output data can be a variation of the agreement between the components involved.

- **By iteratively applying analysis components with more complexity**. This approach is based on the successive application of more complex analysis components to obtain an output with more quality. It assumes that the most complex analysis components produce the best results (that may not always be true) requiring higher computational costs. Thus, we can define the quality level as the application of a set of analysis components and fuse their results (e.g. apply algorithm A for quality level 1, apply algorithm A and B fusing their results for quality level 2,...).

5.3 Application in the proposed video analysis framework

In this section, the application of the previously discussed approaches in the proposed framework is described in detail. We have decided to apply one of each of the proposed

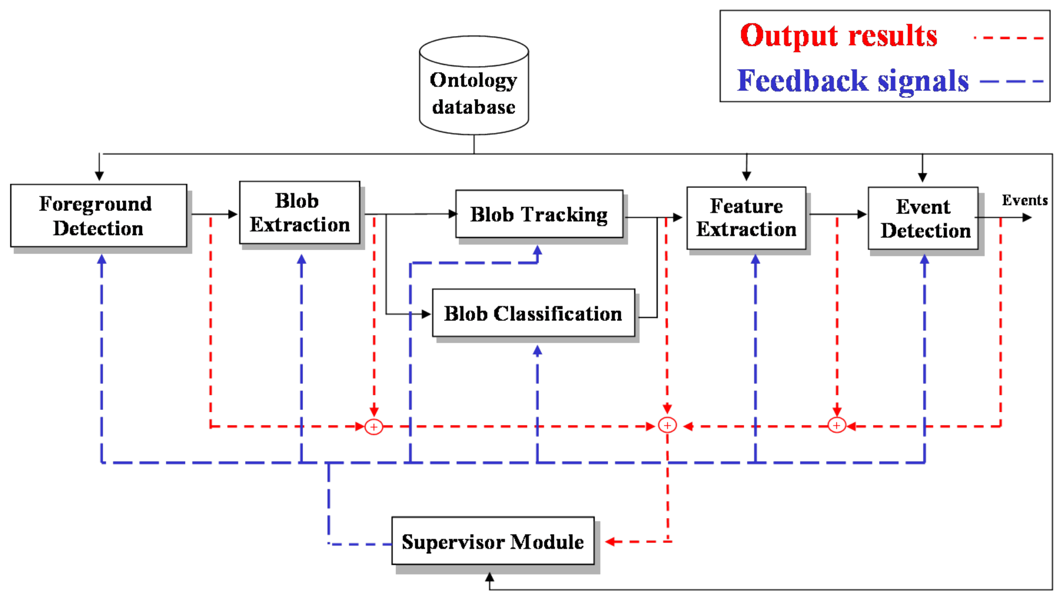


Figure 5.1: Analysis components of the system with Feedback path.

approaches to different analysis components. The selection of which approach is used in each analysis component has been decided taking into account the nature of the processing stage and its implementation complexity. In Figure 5.1 we can see a scheme of the described analysis system with the proposed feedback path. We can see that the supervisor module takes as input the intermediate outputs of the analysis components and produces some feedback signals to change the quality level of the different components.

As a general design pattern, each analysis component that is used in the feedback path has to satisfy two requirements:

- **Performance Evaluation**: the component should provide maps or features to measure the success of its results (that is, to estimate the quality of the output results)

- **Different output quality**: the component should provide different levels of quality for its output data (by applying the approaches proposed in section 5.2).

5.3.1 Implementation of feedback approaches in processing stages

5.3.1.1 Feedback in the foreground segmentation process

In this stage, the ROI-based multi-resolution scheme is implemented due to the nature of this task. As segmentation methods usually rely on analyzing pixels or regions,

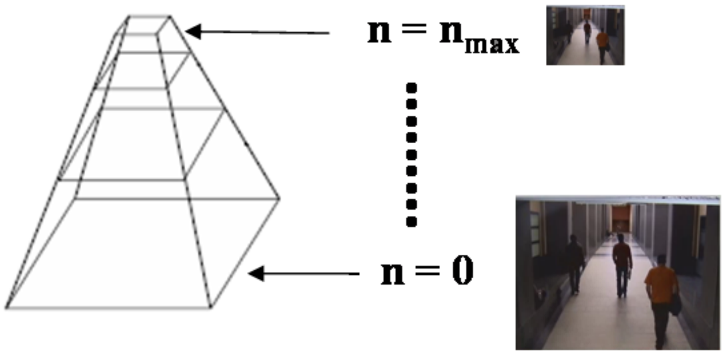


Figure 5.2: Pyramidal representation: levels 0 and $nmax$ represent respectively original resolution and resolution used in background modeling.

the application of a multi-resolution scheme is appropriate for obtaining the different quality levels in the output data by analyzing input images at different levels of detail. The method proposed is based on the pyramidal decomposition [97]. This type of approach allows a quality variation, that is, selecting the different levels of the pyramid.

The first step in the proposed method consists in creating the multi-resolution representation of the image using a regular Gaussian pyramidal representation (see Figure 5.2). In this structure, inter-level relationships are fixed so the structure only reduces the resolution of the input image in the consecutive levels. From an original image ($n = 0$), each pyramid level $n + 1$ is recursively obtained by processing its underlying level n. Specifically, the image at level $n + 1$is convolved with a Gaussian filter (5x5) and then down-sampled by rejecting even rows and columns. Thus, there is a reduction by 2 in width and height. Finally, we iterate to obtain a low resolution image with n equal to n_{max}. In each level, a foreground detection scheme similar to the described in section 4.3.1 is maintained.

In this approach we consider that the foreground seeds (that is, the initial ROIs) to build the foreground/background models are calculated over the lowest resolution image ($n = n_{max}$). This assumption comes from the fact that when we look at an image from a far viewpoint we mainly see the background image. Thus, the segmentation process starts at level $n = n_{max}$ and it is equal to the one described in section 4.3.1. The segmentation results calculated in the lower level of the pyramid ($n = 0$) are propagated towards the upper levels of the pyramid. Let's suppose that we want to grow from level n_2 to level n_1, the segmentation of level n_2 can be propagated to level n_1 by applying three stages. Firstly, we have to calculate the initial foreground mask (mask1) for level n_1, this image is calculated by up-sampling the corresponding image in the previous level n_2(mask2). Secondly, a foreground pixel analysis (in mask1) is performed to calculate the bounding boxes of these pixels. Finally, these detected bounding boxes (ROIs) are refined by applying the same background subtraction process as the one used in level $n = 0$ to obtain the foreground mask of level n_2. In order to avoid that the upper background models of the pyramid become in a non-updated state (remember that the foreground seed (level $n = 0$) is always processed whilst the other quality levels are processed depending on the desired quality), each N frames the whole pyramid model is updated. A description of the iterative process for choosing a quality level and obtaining the desired foreground mask can be seen in the following algorithm description :

Algorithm 3 Algorithm for foreground segmentation using the pyramid structure

```
• Create the Gaussian pyramidal representation of the image
• Calculate the foreground/background model in the lowest resolution ($n = n_{max}$)
• Iterate to the desired output quality (level of the pyramid)
    ✧ Set $n \rightarrow n - 1$
    ✧ While $n > n_{desired}$
        ❏ Up-sample the FG mask of the previous level ($n - 1$)
        ❏ Analyze FG regions to calculate Bounding Boxes (ROIs)
        ❏ Compute the background segmentation process in the detected ROIs
            ▷ Using the foreground/background model of the level
            ▷ Using the corresponding image initially built
        ❏ Set $n \rightarrow n - 1$
    ✧ Get the desired FG mask
```

Variable Quality output The different quality levels can be assumed to correspond with the outputs of the different levels in the pyramid structure. As the levels of the pyramid correspond to an approximation of the analysis of sub-sampled versions of the input image, the different quality levels provide segmentation masks with different resolutions. Figure 5.3 shows an example of segmentation results obtained by selecting different quality levels.

Performance evaluation When the ground-truth segmentation maps are not available, the evaluation of the performance of video object segmentation becomes a difficult task. Some authors like [98], propose metrics based on the color and motion

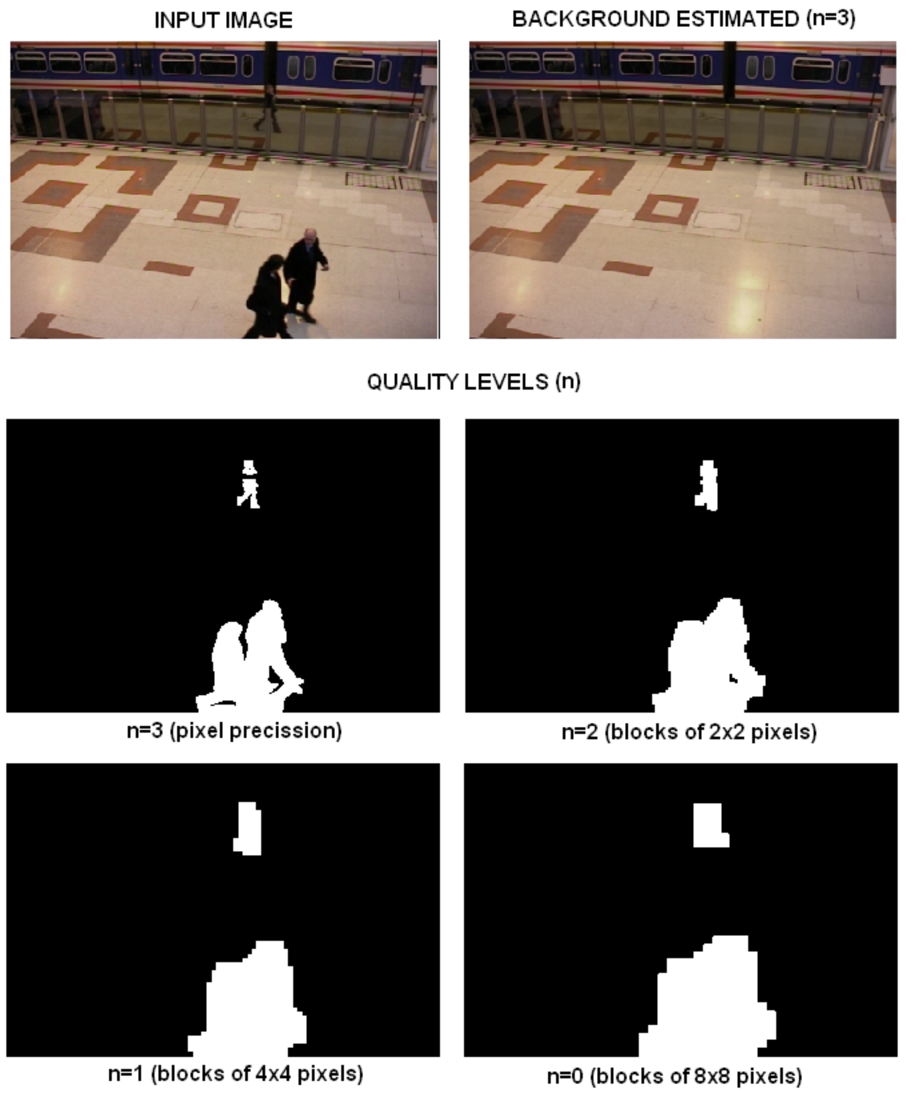

INPUT IMAGE
BACKGROUND ESTIMATED (n=3)
QUALITY LEVELS (n)
n=3 (pixel precission)
n=2 (blocks of 2x2 pixels)
n=1 (blocks of 4x4 pixels)
n=0 (blocks of 8x8 pixels)

Figure 5.3: Example of foreground masks calculated for different quality values (n)

differences along the boundary of the estimated video object plane and the color histogram differences between the current object plane and its temporal neighbors. These experiments show that it is possible to locate the wrong segmented regions which boundary is not correctly detected. The problem of these techniques is that they introduce a computational cost, that is proportional to the number of objects of interest in the scene, limiting the real-time processing capabilities. Due to this problem, we have decided to implement the performance measure as an analysis of the accumulated motion detected in the previous frames and the percentage of the motion detected in the current frame. For example, if we detect that there is not so much motion in the last N frames we can decide to reduce the quality level of the segmentation module. On the other hand, if the sequence becomes crowded, the system must increase the quality level to refine the detection allowing the identification of the individuals in later analysis stages.

5.3.1.2 Feedback in the shadow-removal process

In this stage, the concept of agreement between independent algorithms is used to provide different levels of quality in the shadow detection process. The approach is based on applying independent algorithms to the same data and iteratively try to find the optimum parameters using the agreement as performance measure. To exploit this concept, we have decided to make a simple decomposition of the HSV shadow detection algorithm described in section 4.3.2 and combine it with texture information.

The HSV shadow detection algorithm is decomposed into the intensity and chromaticity parts (described in section 4.3.2) similarly to [95]. Thus, the intensity algorithm is controlled by the parameters α and β. In the chromaticity algorithm, we make the assumption that shadows also cause a decrease in the pixel's colour saturation [23] and we only use *Saturation* as the feature to analyze (discarding *Hue* information). This assumption is true if the background of the scene presents strong colour content. Thus, the chromaticity algorithm is controlled by the parameter τ_S. The texture algorithm is based on the hypothesis that the change of light inside a shadow is quite smooth [99]. In other words, inside a shadow, two adjacent pixels would have the same intensity reduction ratio. If there are multiple shadows of the same object, at the border of the intersection of these shadows, two adjacent pixels may receive two different amounts of light that make the assumption about the consistency of intensity reduction incorrect. Thus, this algorithm is controlled by the value d that indicates the maximum percentage of ratio variation among the shadow regions (or surfaces).

The overall shadow detection process is controlled by the parameters $\{\alpha,\ \beta,\ \tau_S,\ d\}$. If the parameters are set correctly, we expect that there will be a strong agreement between the three independent detectors (that is, the three binary masks show the presence or absence of shadows in each pixel position). As the output of the algorithms are binary images, we can build a 8-value co-occurrence histogram to compute the agreement (combination of two possible outputs for each algorithm). We propose to dynamically adjust these parameters to the incoming data, as in [96],where it was also proposed to measure the agreement between two detectors using the Mutual Information and Kendall's tau (τ)[100]. Kendall's tau can be calculated by using a 8-co-occurrence histogram of the values between the three binary outputs:

$$\tau = \frac{\rho_{XYZ}(1,1,1)\rho_{XYZ}(0,0,0) - \prod\limits_{\substack{i,j,k=0,1 \\ i+j+k\neq 0,3}} \rho_{XYZ}(i,j,k)}{\sqrt{\rho_X(0)\rho_Y(0)\rho_Z(0)\rho_X(1)\rho_Y(1)\rho_Z(1)}} \tag{5.1}$$

where (i, j, k) are binary variables, $\rho_X(i)$,$\rho_Y(j)$,$\rho_Z(k)$ are the partial percentages of the detection of i, j, k values for each algorithm X, Y, Z, $\rho_{XYZ}(i, j, k)$ is the percentage of total points detected as i, j, k and

$$\begin{aligned}\prod_{\substack{i,j,k=0,1 \\ i+j+k\neq 0,3}} \rho_{XYZ}(i,j,k) &= \rho_{XYZ}(1,0,0)\cdot\rho_{XYZ}(0,1,0)\cdot\rho_{XYZ}(0,0,1) \\ &\quad \cdot\rho_{XYZ}(1,1,0)\cdot\rho_{XYZ}(0,1,1)\cdot\rho_{XYZ}(1,0,1)\end{aligned}$$

Then, the gradient ascend optimization technique is iteratively applied to maximize the agreement measured using as a constraint the equation 5.1. In Figure 5.4 we can see the results of the shadow optimization process and the improvement achieved:

Variable Quality output By using this agreement measure, we propose that the quality levels can be the percentage of the unknown points (that is, the points that do not have an agreement between the algorithms). Thus, level 1 can be defined as a 35% of unknown points, level 2 can be defined as a 25% of unknown points, level 3 can be defined as a 20% of unknown points and level 4 can be defined as a 15% of unknown points. The quality level is directly related with the computational complexity of the optimization stage.

Performance evaluation We have selected as the performance measure of the quality levels in this stage the number of unknown points and the percentage of blobs well classified by the people detector (understanding well classified as people

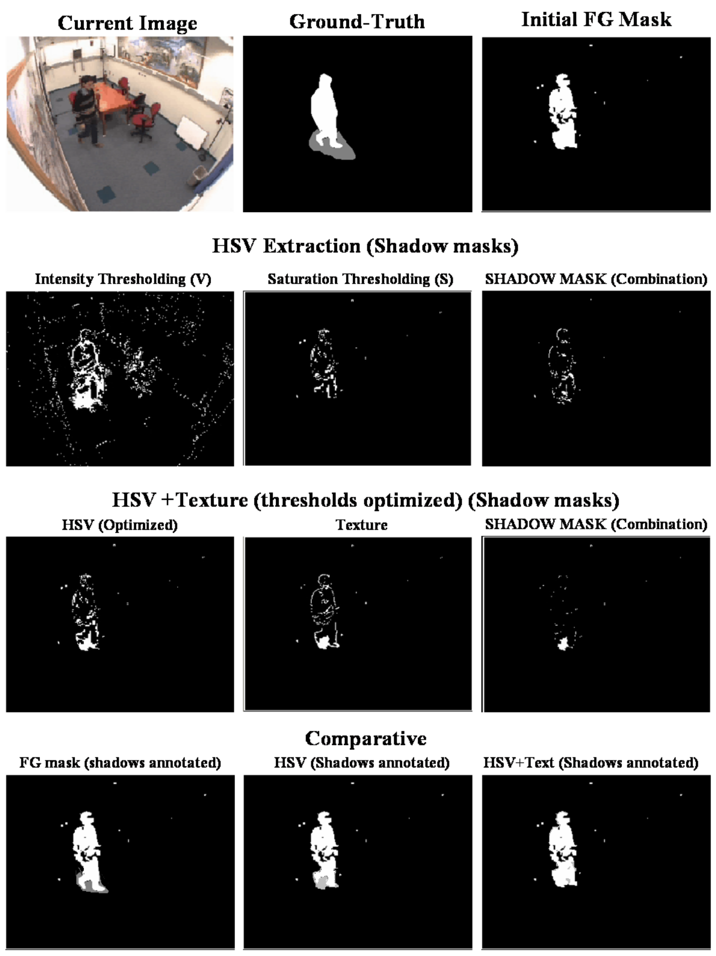

Current Image
Ground-Truth
Initial FG Mask
HSV Extraction (Shadow masks)
Intensity Thresholding (V)
Saturation Thresholding (S)
SHADOW MASK (Combination)
HSV +Texture (thresholds optimized) (Shadow masks)
HSV (Optimized)
Texture
SHADOW MASK (Combination)
Comparative
FG mask (shadows annotated)
HSV (Shadows annotated)
HSV+Text (Shadows annotated)

Figure 5.4: Shadow detection Comparative between the HSV and the proposed algorithms (gray colour indicates shadow detected).

likelihood close to 0 or 1). The supervisor module can decide if the quality level of the shadow detection process needs to be increased or decreased by combining the two measures.

5.3.1.3 Feedback in the noise removal from foreground mask

In this stage, the iterative analysis approach is implemented using the algorithm proposed in section 2.1.1.3. This decision has been taken due to the iterative nature of the algorithm. In this algorithm there are two parameters that describe its behavior: the size of the structuring element (SE) and the stop condition. The size of the SE determines the size and structure (in pixels) of the noise to be eliminated and the stop condition is directly related with the computational cost of the algorithm.

Variable Quality output In this analysis component, we propose that quality variation will be based in the modification of the size of the SE. The stop condition has been established to a fixed value to guarantee the correct recovery of the initially detected blob shape. The modification of the size of the SE is based on the performance evaluation calculated by the supervisor module and affects the size of the noise eliminated. Thus, if we define 4 levels of quality, we propose that level 1 implies a 3x3 size of the SE, level 2 implies a 5x5 size of the SE, level 3 implies a 7x7 of the SE and level 4 implies a 9x9 size of the SE. The shape of the SE has been set to a rectangular shape. In Figure 5.5 we can see an example of the different quality levels and the associated processing time.

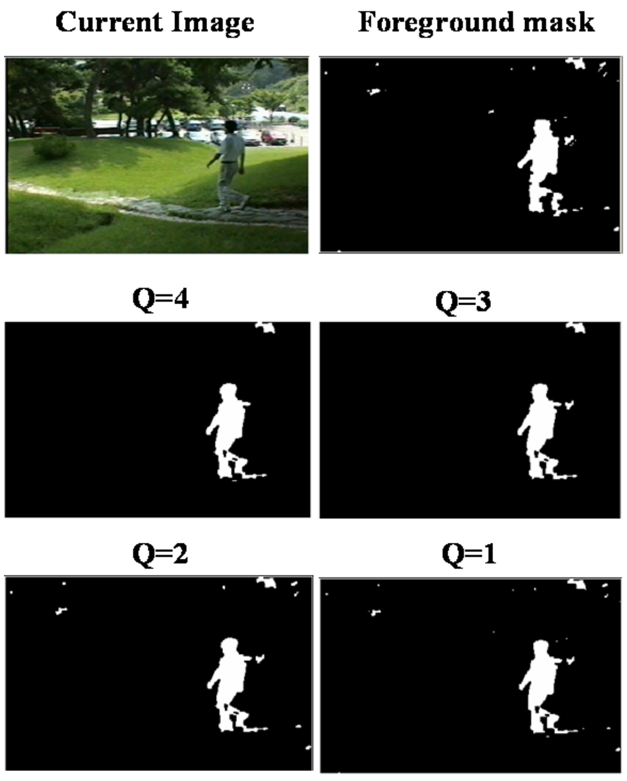


Figure 5.5: Example of the noise removal using different quality.

Performance evaluation In this stage, performance evaluation can be associated to the amount of noise eliminated. This amount is estimated by analyzing the input/output results of this module and the blob analysis module. The performance measure for the size of the SE is the amount of blobs eliminated from the blob extraction stage and their mean size. For example, if there are a lot of small blobs eliminated in the blob extraction stage, probably the size of the SE should be increased. On the other hand, if there is not enough motion in the scene or there are not enough eliminated blobs, this size should be reduced. The size of the SE increases the computational cost of the morphological operations involved in the noise removal process.

5.3.1.4 Feedback in the people detection process

In this stage we want to use the approach of choosing different algorithms (with different complexity) to recognize people in the regions of interest determined in the previous stages. We have decided to add to the described people detection algorithm in section 4.6 two new algorithms: one based on a iterative ellipse fitting process[44] and other based in the approximation of the blob contour with a closed polygon[101](Ghost algorithm)[1].

Variable Quality output As the output variability only depends on the algorithm being executed, the different output quality levels are related with the available algorithms in this analysis component. The algorithms are ordered in increasing computational complexity and executed depending on the desired quality level. Thus, the first algorithm is executed for level 1, the first and second algorithms are executed and fused for level 2 and the three people detection algorithms are executed and fused for level 3. The fusion process is similar to the proposed in [102]. For example in Figure 5.6 we can see the output results of the different quality levels available (levels 1 to 3).

Performance evaluation The proposed performance measure for this stage is very simple. As the output of this module are the probabilities of being people for the blobs detected previously, this measure should indicate if the blobs are correctly detected. We interpret the correct detection when the object is detected as people or non-people with a higher/lower percentage respectively. The proposed measure is described in the equation 5.2 and it is based in a penalty function that penalizes the likelihood values close to 0.5 (worst likelihood result to obtain).

[1]We want to thank Victor Fernandez-Carbajales from the Video Processing and Understanding Lab for providing a C++ implementation of the ellipse and Ghost algorithms.

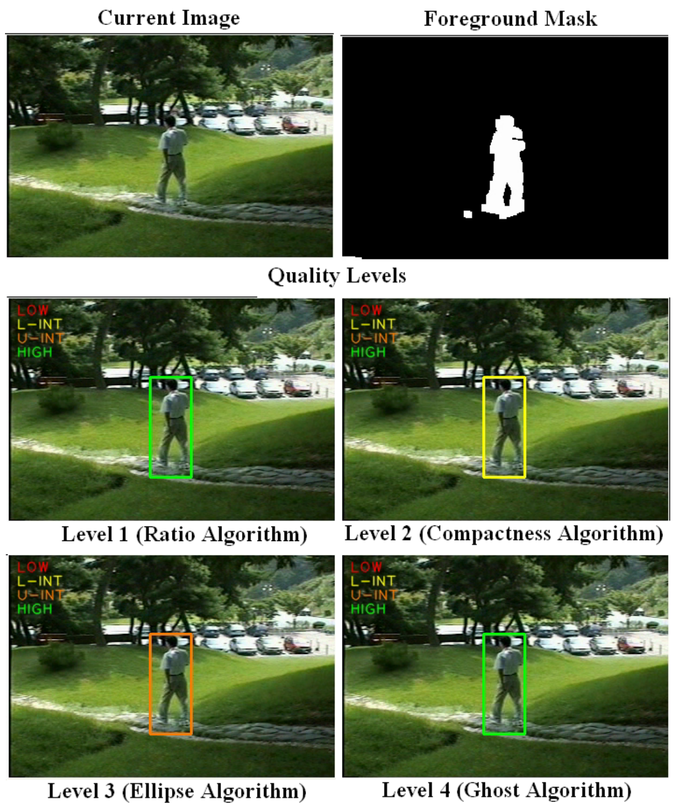

Current Image
Foreground Mask
Quality Levels
LOW
L-INT
U-INT
HIGH
Level 1 (Ratio Algorithm)
Level 2 (Compactness Algorithm)
Level 3 (Ellipse Algorithm)
Level 4 (Ghost Algorithm)

Figure 5.6: Example of the people detection process using different output quality (different algorithms).

$$
\begin{aligned}
PD - performance_i &= \frac{\sum_{j=1}^{N} F\left(PeopleDetectionScore\; Blob_j\right)}{N} \qquad (5.2)\\
where \quad F(x) &= \begin{cases} x & if\; x > 0.5 \\ 1 - x & if\; x < 0.5 \end{cases}\\
N &\quad Number\; of\; blobs\; detected\; in\; frame\; i\\
i &\quad Number\; of\; frame
\end{aligned}
$$

This measure is calculated for each detector used in the selected quality level and for the fusion process. The global measure allows to increase the quality level and the measure of each detector allow to decrease the quality level.

5.3.1.5 Feedback in the event detection process

The detection of the proposed events *PutObject*, *GetObject*, *AbandonedObject* and *StolenObject* requires the use of specific analysis tools (see section 6.2.2). These are "foreground/background object classification" and "owner search".

Foreground/Background object classification In this analysis, a similar approach as the people detection stage has been used. The likelihood of being a foreground/background object is obtained by analyzing the contour and the shape of the blob of interest in the current and background images[103]. Thus, we propose to have three different quality levels corresponding as follows: apply contour analysis for the level 1, colour analysis for the level 2 and a fusion scheme for level 3 (similarly to [103]). The performance measure proposed is based on the likelihood obtained with the analysis performed. Thus, if the analysis applied produces a low probability value, we increase the quality level iteratively to the maximum value (in this case 3). After the evaluation of the feedback path, the quality level returns to the initial state for future detections.

Owner search In this analysis, the owner of the object of interest is searched in the frames close to the event. This search tries to find blobs with high people likelihood close to the object of interest. A more complex algorithm for owner search tries to search the best owner of the object with a more intensive search. It finds the best owner (closest blob with high people likelihood) and re-evaluates its people likelihood with the highest quality level of the people detection stage (that is, applying all the available algorithms and fusing their results). Thus, we propose to have two different quality levels (for the basic search and the intensive search). The performance measure that controls the quality level is the people likelihood of the owner. After the use of the intensive search, the quality level returns to the initial state for future searches.

5.3.2 Supervisor module

This module is responsible for the modification of the analysis components for future runs of the system and the use of the feedback path for re-evaluating the hypothesis. In each frame, it accepts or rejects the hypothesised models (e.g., events) and if the hypothesis is rejected, it activates the feedback path. Firstly, it calculates some performance measures on the partial results of the analysis components. Then it sends feedback signals (see Figure 5.1) with new quality requirements to the modules in which the estimated performance is low to re-execute them in order to accept or reject the unknown hypothesis under evaluation This module has three operation modes: execution of the forward path , model modification for future analysis and feedback path. The initial state of the module is the execution of the forward path. To map the execution state and the possible transition between states, we have decided to use a finite state automata (FSA) (see Figure 5.7).

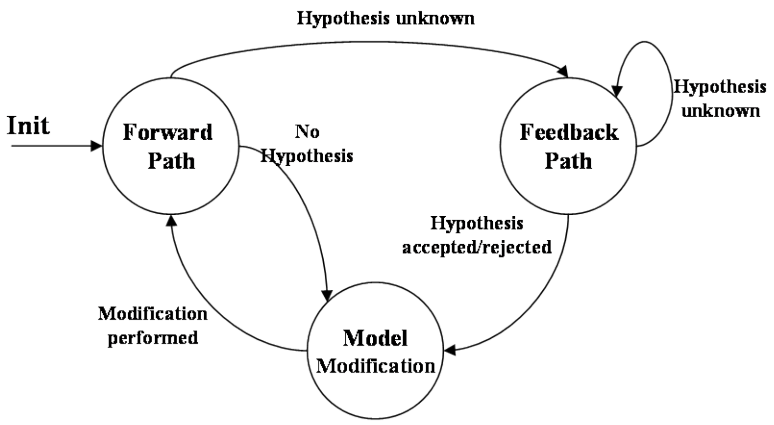


Figure 5.7: Finite State Automata that models the behavior of the supervisor module.

5.3.3 Hypothesis verification

After the execution of the forward path, the supervisor module checks all the hypotheses initially detected. In the proposed analysis framework, the hypotheses that we have to verify are the events detected in the video sequence. The models of these hypotheses are the event models described in section 6.2.2 and the result of the event is a real value between 0 and 1. Thus, the hypothesis acceptation/rejection for each event detected is based on this value (see equation 5.3) activating the “feedback path” if its state is unknown.

$$Hypothesis(H_i) = \begin{cases} Accepted & if\ H_i > 0.7 \\ Rejected & if\ H_i < 0.2 \\ Unknown & otherwise \end{cases} \tag{5.3}$$

5.3.4 Feedback path

After the hypothesis checking phase, each unknown hypothesis is examined in detail to determine if it can be accepted or rejected. The output quality of the selected analysis components is increased to allow a more exhaustive examination and the feedback path re-analyze it. The proposed feedback path is described in Figure5.8.

It is composed of an iterative refinement and re-execution of some analysis modules until the hypotheses is accepted or rejected.

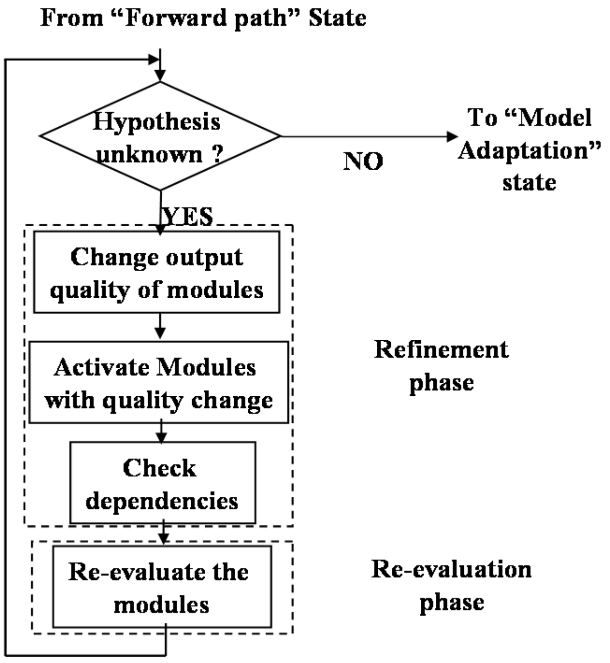

From "Forward path" State
Hypothesis unknown ?
NO
To "Model Adaptation" state
YES
Change output quality of modules
Activate Modules with quality change
Check dependencies
Refinement phase
Re-evaluate the modules
Re-evaluation phase

Figure 5.8: Sequence of operations for the execution of the feedback path.

5.3.4.1 Refinement of processing stages

In this stage, the identification of which module needs to change its output quality depends on the definition of the hypothesis model. For example, the *GetObject* event needs to use the segmentation, tracking and people detection stages. If the corresponding hypothesis is unknown, the identification of which analysis stage has produced the in-determination in the hypothesis is performed. Then, the output quality level of the identified components are increased using the same rules of the Model Modification stage (obviously avoiding the decreasing quality cases) and an iterative demand of higher quality level is started if the hypothesis under examination remains in the unknown state.

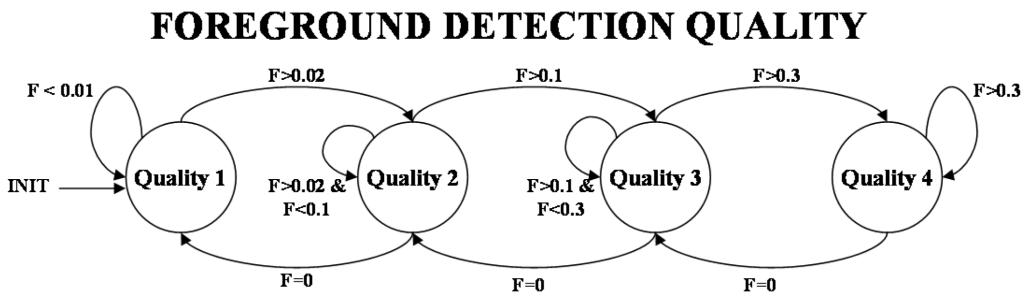
5.3.4.2 Re-evaluation stage

Finally, the components with a change in their output quality level are re-executed. Furthermore, some related components are executed to update the data of the hypothesis under examination. For example, if the segmentation stage needs to be re-executed, we also might re-calculate the blob analysis stage to detect new blobs or eliminate older blobs with the new segmentation masks calculated.

5.3.5 Model modification

In this section, the mechanisms to adapt the general-purpose processing stages to the different video sequences analyzed are described. These modifications should improve the performance of all high-levels that produce the semantic description (e.g., event detection module) and also they should allow to efficiently use the available algorithms and resources.

After the analysis of each frame (forward and feedback paths, if needed), the supervisor module checks the state of all the processing stages changing the output quality requirements of some components by analyzing the intermediate output results (see Figure 5.1). To model the transitions between the different quality levels of each component, we have decided to use finite state automata (FSA) controlled by the corresponding performance measures. Thus, the supervisor module updates the state of each FSA and then changes to the "forward path" state to analyze the next frame. In the following figures we can see the models for modifying the output quality of each stage.



A → Percentage of motion detected in current frame
B → Existence of Blobs Of Interest in the scene (BOI)
Quality → Level i of the pyramid segmentation model

Function $F = AB$

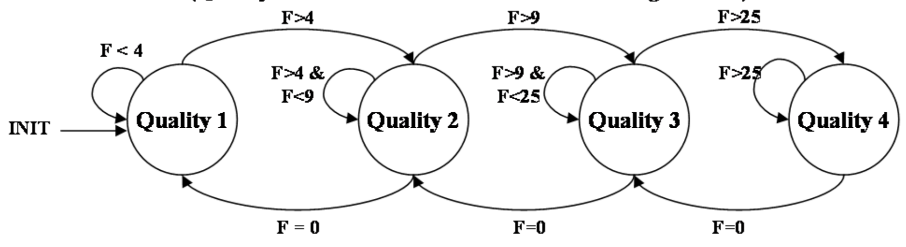
where $A = \# foreground\_pixels / size\_frame$

$$B = \begin{cases} 1 & if \quad \# BOI >= 1 \\ 0 & if \quad \# BOI <= 0 \end{cases}$$

Figure 5.9: FSA that changes the quality levels of the Foreground Segmentation stage.

NOISE REMOVAL QUALITY

(Quality → Size modification of the structuring element)

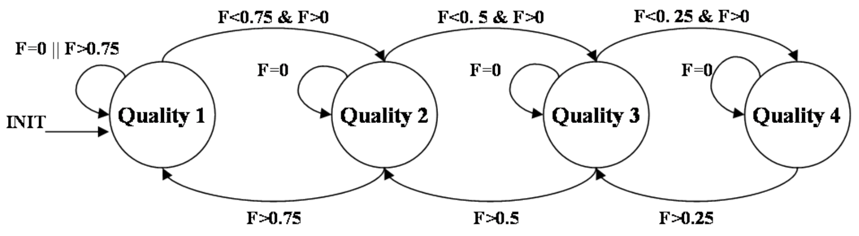


A → Percentage of motion detected in current frame (no execution if A < 0.001)
B → Mean Size of the eliminated blobs after the blob extraction process
C → Number of eliminated blobs after the blob extraction process
Functions $F = BC$

$$A = \# foreground\_pixels / size\_frame$$
where $$B = Mean\_size\_deleted\_blobs$$
$$C = \begin{cases} 1 & if \quad \# BOIDel >= 1 \\ 0 & if \quad \# BOIDel <= 0 \end{cases}$$

Figure 5.10: FSA that changes the quality levels of the Noise Removal stage.

SHADOW REMOVAL QUALITY



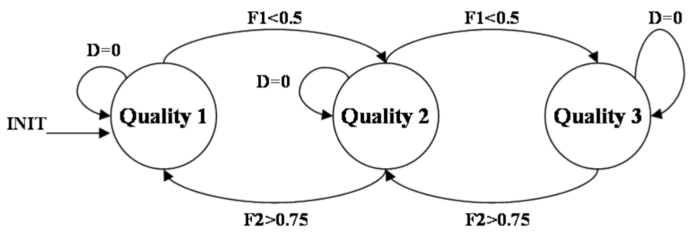
A → Percentage of motion detected in current frame (no execution if A < 0.001)
B → Percentage of blobs well-classified as people (P~1) or non-people (P~0)
C → Blobs detected after the blob extraction process (no state change if C=0)
Quality → Percentage of unknown points in the agreement optimization

Functions $F = BC$

$$A = \# foreground\_pixels / size\_frame$$
where $$B = See\_equation\_5.3$$
$$C = \begin{cases} 1 & if \quad \# BOI >= 1 \\ 0 & if \quad \# BOI <= 0 \end{cases}$$

Figure 5.11: FSA that changes the quality levels of the Shadow detection stage.

PEOPLE DETECTION QUALITY



A → Percentage of motion detected in current frame (no execution if A < 0.001)
B → Percentage of blobs well-classified as people (P~1) or non-people (P~0)
C → Percentage of blobs well-classified as people (P~1) or non-people (P~0) with the previous level
Quality → Percentage of unknown points in the agreement optimization

Functions $F1 = BD$, $F2 = CD$

where $A = \#foreground\_pixels / size\_frame$
$B = see \quad equation \quad 5.3$
$C = see \quad equation \quad 5.3$
$$D = \begin{cases} 1 & if \quad \#BOI >= 1 \\ 0 & if \quad \#BOI <= 0 \end{cases}$$

Figure 5.12: FSA that changes the quality levels of the people detection stage.

Chapter 6

Experimental Work

6.1 Introduction

In this chapter, we describe the experiments carried out for testing the proposed video analysis framework (described in chapter 4) and the feedback and adaptation strategies (described in chapter 5) introduced in that framework. In the next sections, we describe the dataset used and the set of events selected to test the system (section 6.2), the metrics (section 6.3), the evaluation results comparing both systems without/with feedback (section 6.4) and some examples of these results (section 6.5).

6.2 Experimental data

6.2.1 Dataset used

Experiments were carried out on several test sequences from public datasets related with the selected events to detect (see section 6.2.2). They are:

- **PETS2006**: Ninth IEEE International Workshop on Performance Evaluations of Tracking and Surveillance, June 2006.URL http://pets2006.net/
- **PETS2007**: Tenth IEEE International Workshop on Performance Evaluations of Tracking and Surveillance, October 2007. URL http://pets2007.net/.
- **CAVIAR**: Context Aware Vision using Image-based Active Recognition. URL: http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/
- **i-LIDS dataset for AVSS2007**: Fourth IEEE International Conference on Advanced Video and Signal based Surveillance, September 2007. URL: http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007_d.html

- **WCAM**: Video Surveillance video sequences (The test visual material used in this work has been provided with courtesy of Thales Security Systems within the scope of the IST FP6 WCAM project). URL: http://wcam.epfl.ch/
- **VISOR**: Video Surveillance Online Repository.
  URL: http://imagelab.ing.unimore.it/visor
- **CVSG**: A Chroma-based Video Segmentation Ground-truth
  URL: http://www-vpu.ii.uam.es/CVSG/
- **ITEA CANDELA project**: scenarios for abandoned object detection.
  URL: http://www.multitel.be/~va/candela/abandon.htmlCategorization

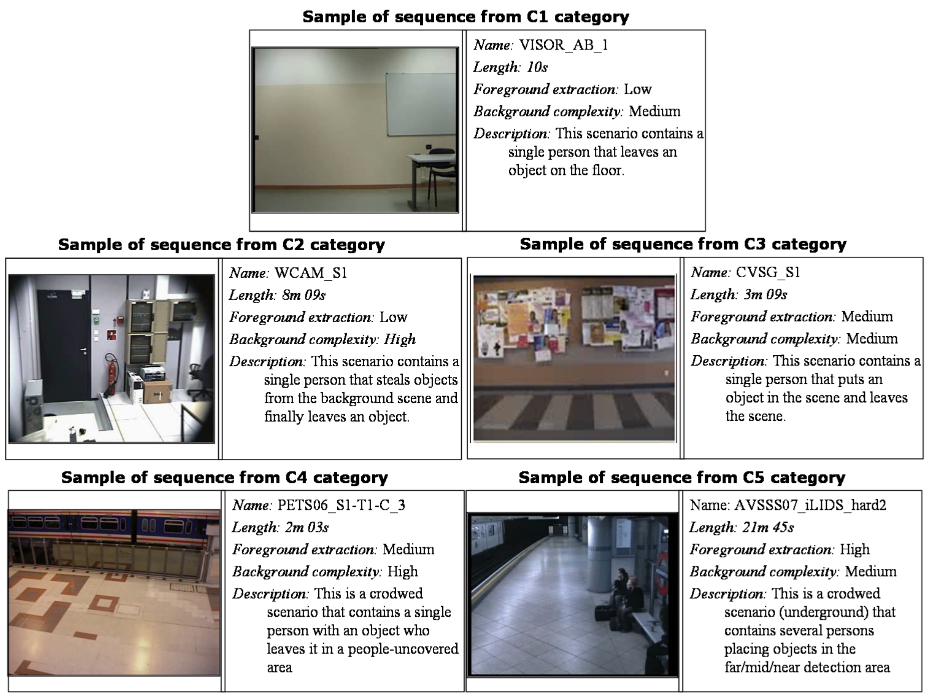
In order to evaluate the performance of the proposed approach, we have grouped all the test sequences into different complexity categories depending on two aspects:

- **Foreground object extraction complexity** defined as the difficulty to extract moving and stationary objects in a scenario. It is related with the number of objects, their velocity, partial occlusions, lighting changes and the people classification difficulty.
- **Background complexity** defined as the presence of edges, multiple textures and objects belonging to the background (like waving trees, water surface,...).

Moreover, all the test sequences have been resized to the 720x576 or 360x288 standard resolutions (depending on which is more close to the initial resolution of the sequence) to homogenize the video data for the experiments. A description of complexity levels and length of the associated content is shown in Table 6.1, whilst Figure 6.1 shows an example of each category with a description of the content:

Category	**Length**	**Complexity**	
		Foreground Extraction	**Background**
C1	15m 20s	Low	Medium
C2	20m 30s	Low	High
C3	1h 32m 18s	Medium	Medium
C4	2h 6m 37s	Medium	High
C5	1h 26m 16s	High	Medium

Table 6.1: Test Sequence Categorization



Sample of sequence from C1 category

Name: VISOR_AB_1
Length: 10s
Foreground extraction: Low
Background complexity: Medium
Description: This scenario contains a single person that leaves an object on the floor.

Sample of sequence from C2 category

Name: WCAM_S1
Length: 8m 09s
Foreground extraction: Low
Background complexity: High
Description: This scenario contains a single person that steals objects from the background scene and finally leaves an object.

Sample of sequence from C3 category

Name: CVSG_S1
Length: 3m 09s
Foreground extraction: Medium
Background complexity: Medium
Description: This scenario contains a single person that puts an object in the scene and leaves the scene.

Sample of sequence from C4 category

Name: PETS06_S1-T1-C_3
Length: 2m 03s
Foreground extraction: Medium
Background complexity: High
Description: This is a crowded scenario that contains a single person with an object who leaves it in a people-uncovered area

Sample of sequence from C5 category

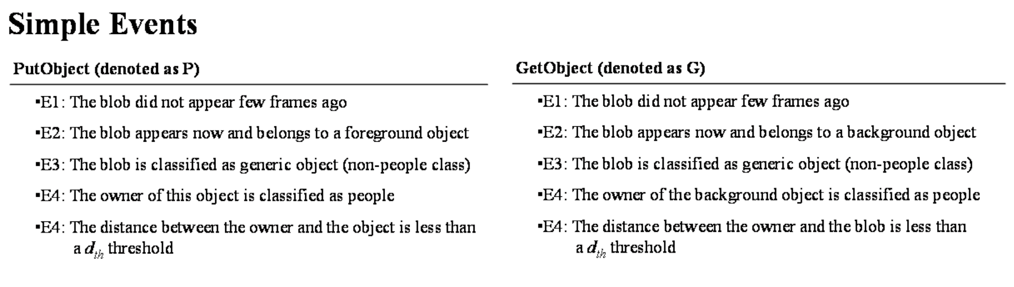
Name: AVSSS07_iLIDS_hard2
Length: 21m 45s
Foreground extraction: High
Background complexity: Medium
Description: This is a crodwed scenario (underground) that contains several persons placing objects in the far/mid/near detection area

Figure 6.1: Examples of the test categories

6.2.2 Selection of events to detect

For evaluating the proposed system, we have designed four event models. For detecting the events, some analysis tools have been implemented [103]. These tools help to identify foreground or background objects by analyzing the contour and color information of the associated blob.

The first two events to be detected correspond to the simple events (*Get/Put Object*) shown in Figure 6.2. These events are based on the action performed by a human (the owner of the object being putted/removed in/from the scene) and the exact instant when the action occurs.



Simple Events

PutObject (denoted as P)

- E1: The blob did not appear few frames ago
- E2: The blob appears now and belongs to a foreground object
- E3: The blob is classified as generic object (non-people class)
- E4: The owner of this object is classified as people
- E4: The distance between the owner and the object is less than a d_{th} threshold

GetObject (denoted as G)

- E1: The blob did not appear few frames ago
- E2: The blob appears now and belongs to a background object
- E3: The blob is classified as generic object (non-people class)
- E4: The owner of the background object is classified as people
- E4: The distance between the owner and the blob is less than a d_{th} threshold

Figure 6.2: Models of the simple events to be detected in the test sequences

The other two events correspond to complex events (*AbandonedObject* and *StolenObject*). These events are composed of different sub-events starting from the *PutObject* and *GetObject* events respectively. For each event, its state returns to the initial state if each sub-scenario fails to be recognized. The final likelihood of each complex event recognized is calculated as the average of the likelihood of the sub-events involved.

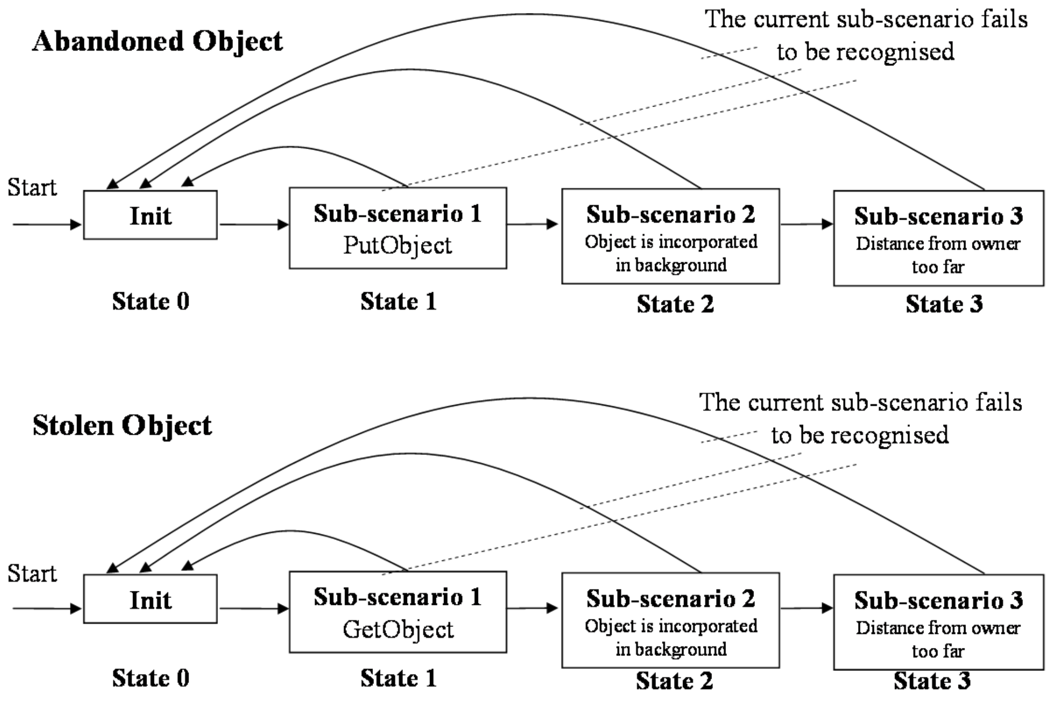
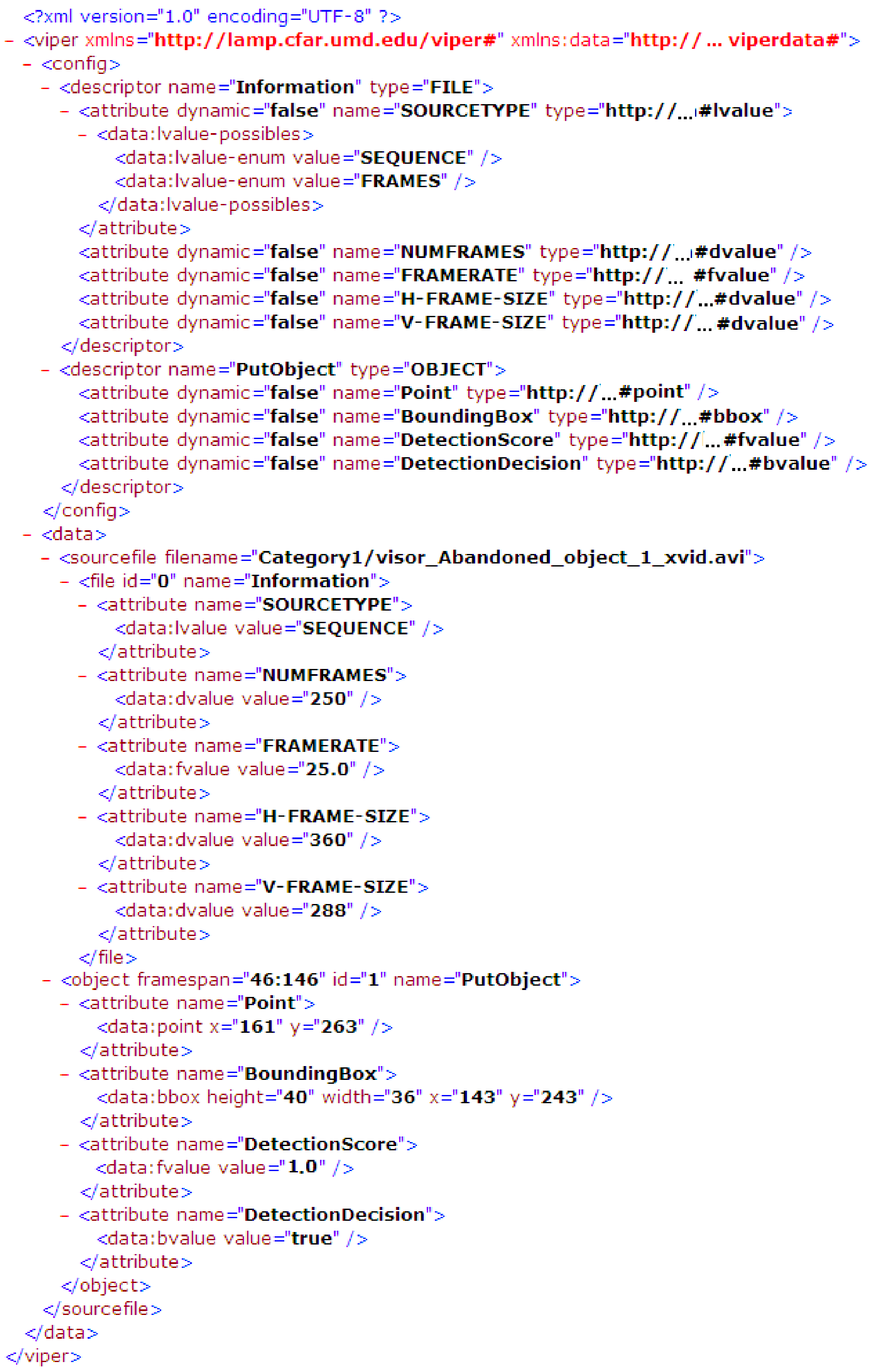

Abandoned Object
The current sub-scenario fails to be recognised
Start
Init
Sub-scenario 1
PutObject
Sub-scenario 2
Object is incorporated in background
Sub-scenario 3
Distance from owner too far
State 0
State 1
State 2
State 3
Stolen Object
The current sub-scenario fails to be recognised
Start
Init
Sub-scenario 1
GetObject
Sub-scenario 2
Object is incorporated in background
Sub-scenario 3
Distance from owner too far
State 0
State 1
State 2
State 3

Figure 6.3: Models of the complex events to be detected in the test sequences

6.2.3 Ground truth

For each video file, we have created a ground truth description of the events of out interest: *PutObject*, *GetObject*, *AbandonedObject* and *StolenObject*. There are some tools available to annotate video files and we have tested the following: Anvil (http://www.anvil-software.de/), IBM Annotation Tool (http://www.research.ibm.com/VideoAnnEx/) and Viper Annotation Toolkit (http://viper-toolkit.sourceforge.net/). Finally we have decided to use the Viper tool because it is the most popular in the research community, it is easy to manage and it has performance evaluation tools. An example of a ground-truth annotation file is shown in Figure 6.4.



```
  <?xml version="1.0" encoding="UTF-8" ?>
- <viper xmlns="http://lamp.cfar.umd.edu/viper#" xmlns:data="http:// ... viperdata#">
  - <config>
    - <descriptor name="Information" type="FILE">
      - <attribute dynamic="false" name="SOURCETYPE" type="http://...#lvalue">
        - <data:lvalue-possibles>
            <data:lvalue-enum value="SEQUENCE" />
            <data:lvalue-enum value="FRAMES" />
          </data:lvalue-possibles>
        </attribute>
        <attribute dynamic="false" name="NUMFRAMES" type="http://...#dvalue" />
        <attribute dynamic="false" name="FRAMERATE" type="http://... #fvalue" />
        <attribute dynamic="false" name="H-FRAME-SIZE" type="http://...#dvalue" />
        <attribute dynamic="false" name="V-FRAME-SIZE" type="http://... #dvalue" />
      </descriptor>
    - <descriptor name="PutObject" type="OBJECT">
        <attribute dynamic="false" name="Point" type="http://...#point" />
        <attribute dynamic="false" name="BoundingBox" type="http://...#bbox" />
        <attribute dynamic="false" name="DetectionScore" type="http://...#fvalue" />
        <attribute dynamic="false" name="DetectionDecision" type="http://...#bvalue" />
      </descriptor>
    </config>
  - <data>
    - <sourcefile filename="Category1/visor_Abandoned_object_1_xvid.avi">
      - <file id="0" name="Information">
        - <attribute name="SOURCETYPE">
            <data:lvalue value="SEQUENCE" />
          </attribute>
        - <attribute name="NUMFRAMES">
            <data:dvalue value="250" />
          </attribute>
        - <attribute name="FRAMERATE">
            <data:fvalue value="25.0" />
          </attribute>
        - <attribute name="H-FRAME-SIZE">
            <data:dvalue value="360" />
          </attribute>
        - <attribute name="V-FRAME-SIZE">
            <data:dvalue value="288" />
          </attribute>
        </file>
      - <object framespan="46:146" id="1" name="PutObject">
        - <attribute name="Point">
            <data:point x="161" y="263" />
          </attribute>
        - <attribute name="BoundingBox">
            <data:bbox height="40" width="36" x="143" y="243" />
          </attribute>
        - <attribute name="DetectionScore">
            <data:fvalue value="1.0" />
          </attribute>
        - <attribute name="DetectionDecision">
            <data:bvalue value="true" />
          </attribute>
        </object>
      </sourcefile>
    </data>
</viper>
```

Figure 6.4: Examples of Ground Truth annotation using the Viper tool

6.2.4 Training phase

Some processing stages of the system need a training phase to acquire the models used in their tasks. These stages are the people detection stage and the object classification stage. The people detection algorithms need a silhouette-based people dataset and the recognition of foreground/background blobs need a training set with the abandoned or removed blobs annotated. In the proposed experiments, we suppose that the necessary training models have been previously acquired (this training phase is out of the scope of this work).

6.3 Metrics

For evaluating the performance of the overall proposed framework and the feedback strategies used, we obtain for each event detected three types of information: its likelihood, location (or area) where the event occurs and range of frames where the event takes place . The final decision is based on the application of constraints over these information as shown in in the following equation:

$$Event\_Detected = \begin{cases} True & if\ score > \rho\ and \\ & \|time\_start\_alarm - time\_start\_gt\| < \tau\ and \\ & \|time\_end\_alarm - time\_end\_gt\| < \tau\ \ and \\ & overlapped\_area > 50\% \\ False & Otherwise \end{cases} \tag{6.1}$$

The performance of the detected events on a video sequence are evaluated in terms of Precision (P) and Recall (R). Precision represents the ratio between the true alarms (that is, they are in the ground truth) and the total number of alarms detected by the module. Recall represents the ratio between alarms that correspond to real alarms in the ground truth and the total number of alarms in the ground truth.

$$R = \frac{TP}{TP + FN} \quad P = \frac{TP}{TP + FP} \tag{6.2}$$

where TP (True Positive or true alarms) are the events detected from the ground truth, FN (False Negatives or missed events) are the event not detected from the ground truth and FP (False Positive or false alarms) are events detected that not exist in the ground truth. The ViPER-PE tool has been used to evaluate the performance of the system (see Appendix B for a complete description of the evaluation process).

6.4 Performance evaluation comparison

In this section, experimental results of the proposed analysis system and the feedback strategies are presented. The system has been implemented in C++, using the OpenCV image processing library (http://www.intel.com/technology/computing/opencv/). The tests have been performed on a Pentium IV with a CPU frequency of 2.8 GHz and 2GB RAM. Due to implementation issues related with the image processing library, the noise removal stage applied after the initial foreground detection has been eliminated maintaining only the last noise removal stage (see section 4.3). Additional results can be found at http://www-vpu.ii.uam.es/~jcs/Master/

6.4.1 Event detection

We have tested the performance of the overall system to check if the use of feedback strategies improves the initial results of the system without using feedback for each category. It should be noted that the same parameters were used for all the sequences, demonstrating the robustness of the proposed system and the feedback strategies introduced for adaptation and hypothesis testing. The results of the experiments carried out are summarized in table 6.2. This table shows that the use of the proposed feedback scheme improves the overall score in each category due to the re-evaluation stage included. As can be seen from the results, our approach achieves better results in sequences with less foreground object extraction complexity and is robust to background complexity. This is because the algorithms that use background information are robust to highly textured and blurred backgrounds (obtained with the multi-resolution approach followed in the pyramidal foreground detection stage).

	No Feedback					**Using Feedback**				
Category	**TP**	**FN**	**FP**	**R**	**P**	**TP**	**FN**	**FP**	**R**	**P**
C1	20	0	6	1	0.77	20	0	2	1	0.91
C2	39	2	15	0.95	0.72	38	3	10	0.92	0.79
C3	480	169	287	0.73	0.62	468	181	198	0.72	0.70
C4	258	201	536	0.56	0.32	215	244	303	0.46	0.41
C5	60	109	168	0.35	0.26	55	114	125	0.32	0.30
Total	857	481	1012	0.64	0.45	796	542	638	0.59	0.55

Table 6.2: Overall performance of the proposed framework and the improvement using feedback strategies

6.4.2 Computational Cost

In this section, the computational performance of the proposed system without/with the use of feedback strategies is discussed. As said above, there are two resolutions in the dataset test sequences (360x288 and 720x576). The results obtained are summarized in terms of these two resolutions (in table 6.3) and plotted in Figure 6.5. These results show that the application of feedback strategies reduce the overall computational cost without affecting the overall event detection performance (table 6.2). For example, if the sequence presents too much activity the supervisor module will probably use a higher quality level on foreground segmentation but real video-surveillance sequences are characterized for presenting long periods without interesting events to detect and the supervisor module will reduce the quality requirements of some analysis components until the scene activity increases

	Average processing time (ms) 720x576		Average processing time (ms) 360x288	
Processing Stage	No Feedback	Feedback	No Feedback	Feedback
Foreground detection	61.93 (34%)	22.43 (20%)	15.45 (41%)	5.21 (20%)
Shadow Detection	65.50 (35%)	29.67 (25%)	16.42 (44%)	12.44 (48%)
Noise Removal	12.74 (7%)	10.64 (9%)	2.81 (7.5%)	2.67 (10%)
Blob Extraction	1.31 (1%)	1.35 (1%)	0.36 (1%)	0.20 (1%)
Tracking	2.90 (2%)	3.21 (3%)	0.41 (1%)	0.43 (2%)
People Detection	13.15 (7%)	14.85 (13%)	1.37 (3.5%)	1.82 (7%)
Event Detection	27.13 (15%)	29.96 (26%)	0.85 (2%)	2.14 (8%)
Total	184.69 ms	117.22 ms	37.69 ms	25.68 ms
Average fps	5.41	8.53	26.52	38.94

Table 6.3: Computational Cost comparison between the two proposed systems

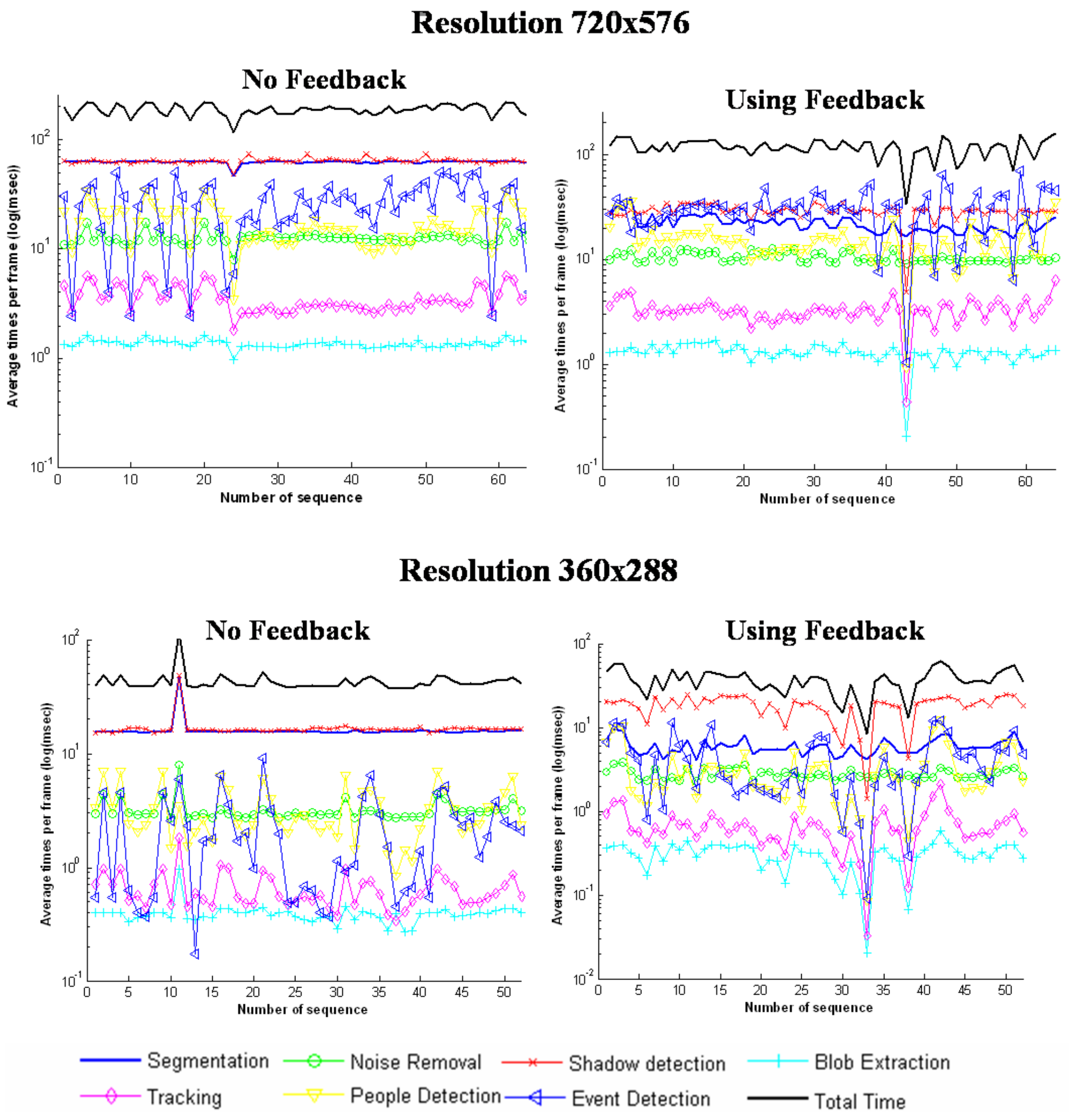


Figure 6.5: Computational Cost Comparative between the system not using or using feedback strategies for adapt the models and re-evaluate the hypothesis (results are presented in logarithmic scale to distinguish the time values)

6.5 Examples

6.5.1 Computational cost reduction

The introduced feedback strategies allow to change the quality level of the analysis components during run-time for improving future runs. This change or adaptation increases/reduces the computational complexity of some algorithms depending on the activity in the video sequence. In Figure 6.6 we can see how the supervisor module reduces the computational complexity when there are not objects of interest in the scene and how it increases the quality requirements when there are objects of interest that may produce events to detect.

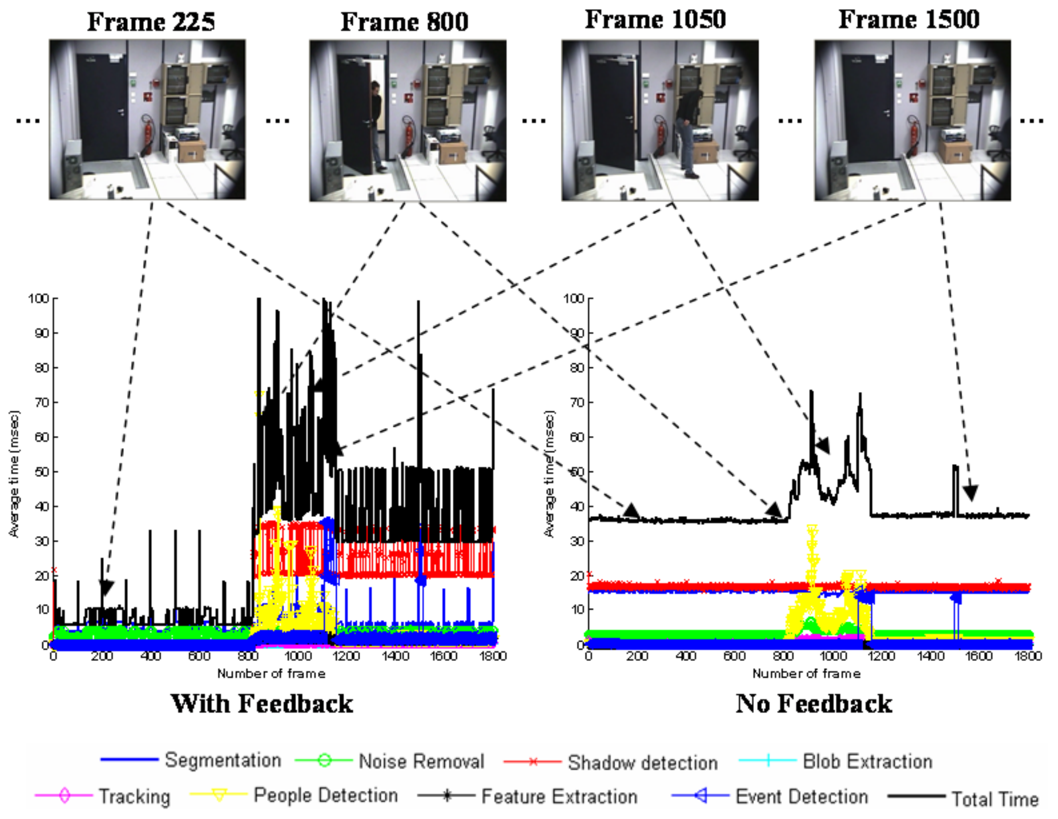


Figure 6.6: Computational Cost Comparative Sample between the system not using or using feedback strategies for adapting the models and re-evaluating the hypothesis

6.5.2 Improved event detection

Figure 6.7 shows an example of how the proposed feedback strategy improves the detection likelihood of the unknown hypothesis (accepting it). Firstly, an initial likelihood of the *AbandonedObject* hypothesis is computed and the supervisor module executes the feedback path to re-evaluate the unknown hypothesis. People detection quality is increased in the refinement stage. As it remains in the unknown state, the supervisor module decides to increase the quality of the foreground detection and the people detection stages (re-evaluating the shadow and noise removal stages).

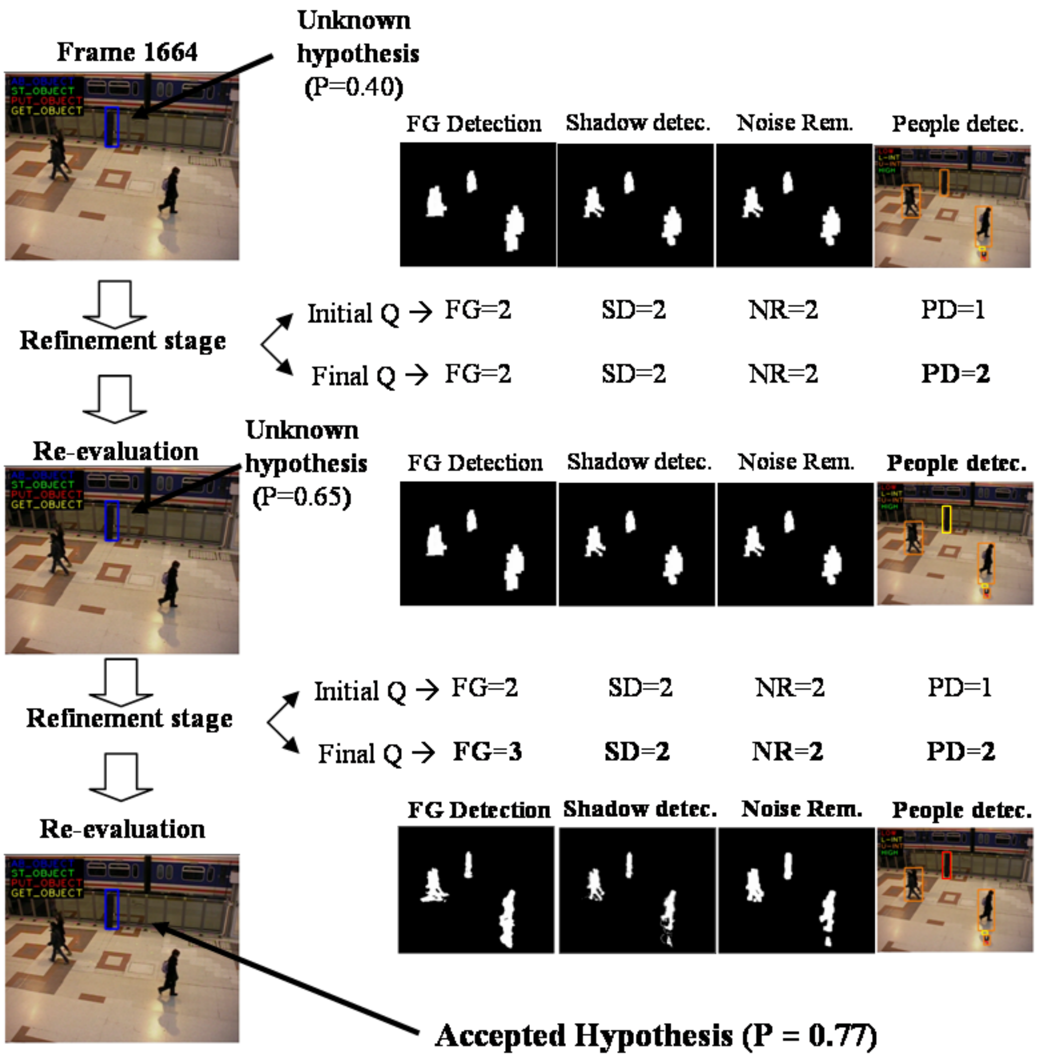


Figure 6.7: Improved hypothesis (event) detection by using the proposed feedback scheme

6.5.3 Improved event rejection

In Figure 6.8 we can see an example of how the proposed feedback path rejects an unknown hypothesis by computing an iterative evaluation and refinement of it. Firstly, the supervisor module decides to increase the quality of the people detection stage to re-evaluate the unknown hypothesis. As it remains in the unknown state, the supervisor module decides to increase quality of the shadow detection and people detection stages (not increasing foreground quality due to this stage is in the maximum quality level). Finally, the re-evaluation of this hypothesis with the new quality requirements allows to reject the hypothesis under evaluation.

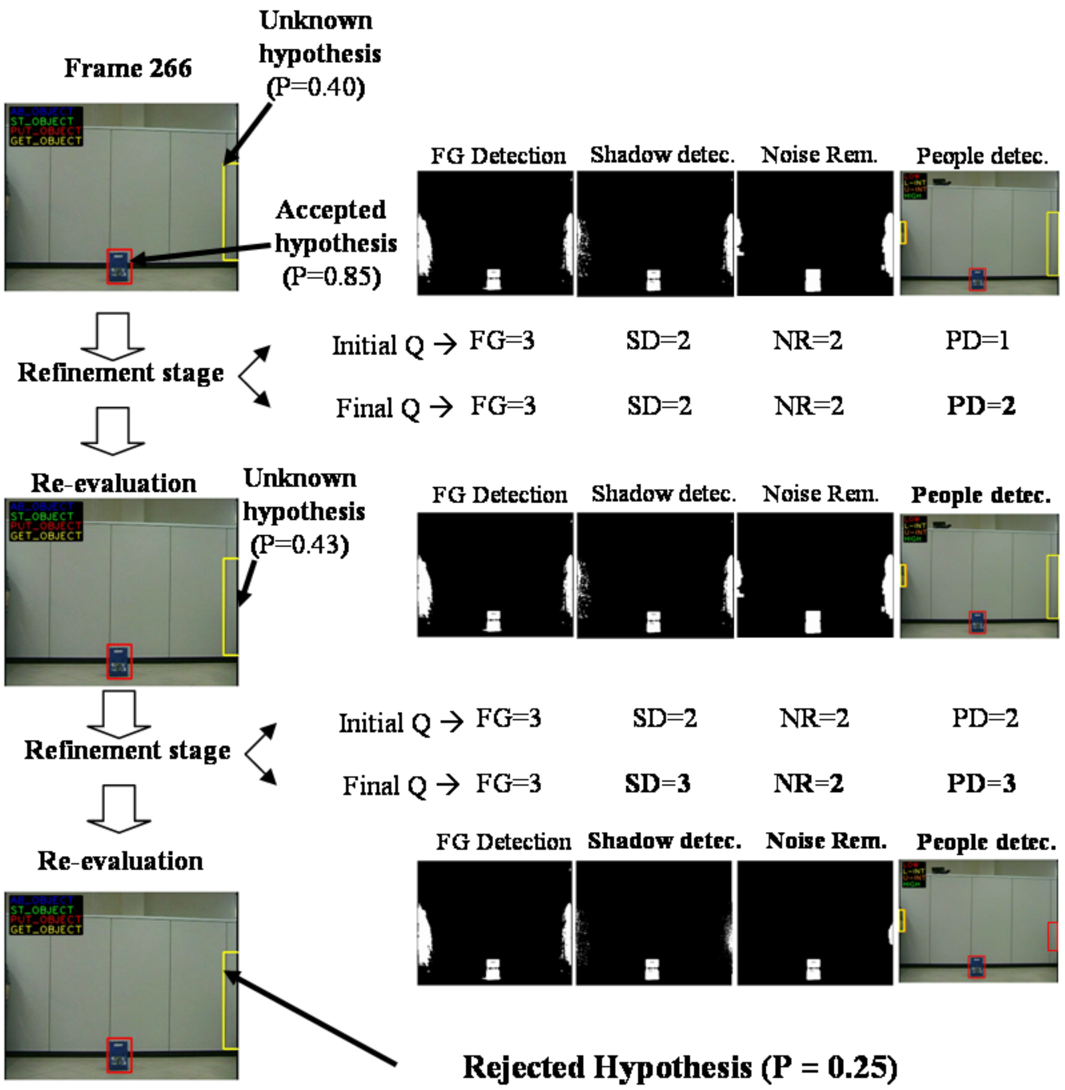

Frame 266
Unknown hypothesis (P=0.40)
Accepted hypothesis (P=0.85)
FG Detection
Shadow detec.
Noise Rem.
People detec.
Refinement stage
Initial Q → FG=3 SD=2 NR=2 PD=1
Final Q → FG=3 SD=2 NR=2 PD=2
Re-evaluation
Unknown hypothesis (P=0.43)
FG Detection
Shadow detec.
Noise Rem.
People detec.
Refinement stage
Initial Q → FG=3 SD=2 NR=2 PD=2
Final Q → FG=3 SD=3 NR=2 PD=3
Re-evaluation
FG Detection
Shadow detec.
Noise Rem.
People detec.
Rejected Hypothesis (P = 0.25)

Figure 6.8: Improved hypothesis (event) rejection by using the proposed feedback scheme

Chapter 7

Conclusions and future work

7.1 Summary of work

In this document, four main contributions have been presented:

- **An ontology for detecting video events in video-surveillance sequences**. The proposed ontology is structured in three main parts: the basic concepts/entities, their extensions and the domain extensions. The basic concepts of the ontology are: *scene*, *system* and *user*. This ontology is centered only in the scene and system concepts. The extensions of these concepts are based on the knowledge of the surveillance domain. Thus the *scene* entity is described in terms of *object*, *event* and *sceneContext* entities, and the *system* entity is described in terms of *status*, *capabilities* and *reactions* entities. Finally, a domain extension is proposed for the underground surveillance domain.

- **A generic video analysis framework for detecting events in video sequences**. The design of this framework has been guided by the proposed ontology. It includes a background subtraction algorithm to detect foreground objects and some post-processing algorithms (noise removal and shadow detection). Finally, a tracking stage and a people detector stage are combined in an event detection phase to recognize the events of interest. Furthermore, due to its modular design the framework components can easily be changed with better components.

- **The design of feedback strategies** to improve the event recognition process and to modify the parameters of the algorithms for improving performance in future runs. Feedback strategies are based on selective re-evaluation of some analysis stages which can have changing quality output. This output quality variation is performed by selecting the desired quality level and it is automatically mapped into a parameter adjustment. Then, we have identified four different approaches for quality variation that are implemented in different processing stages. Furthermore, these feedback strategies allow to automatically change the quality of the processing stages for improving performance in future runs over the video sequence (e.g., reducing the computational cost of the analysis). A supervisor module has been added to coordinate the feedback strategies employed in the proposed framework.

- **The design and annotation of an event-based dataset**. In order to provide a good performance evaluation of the proposed framework, we have designed a dataset composed of several sequences from public datasets. We have grouped the test sequences into five different complexity categories depending on the of the foreground object extraction difficulty and the background complexity. Then we have annotated these sequences using the Viper-GT tool [104] describing each event in terms of its location in space and time. Finally, the dataset and its ground-truth is used for evaluation with the Viper-PE tool [104].

7.2 Conclusions

The work presented in this document is focused on the improvement of the processes for semantic description extraction from video sequences. The application of contextual information in the video analysis process and the use of this information to provide feedback between the analysis stages are proposed to enhance the performance of the semantic description extraction process operating with real-time constraints. Firstly, the analysis of video-surveillance sequences is proposed as a case of study. The contextual information is represented by an event detection ontology and the feedback strategies applied are based on the computation of performance measures in the partial results of the analysis.

As a first step, an ontology has been designed and implemented to detect events in video-surveillance sequences. This ontology defines the concepts that compose the contextual information of the video surveillance analysis domain. Then a framework has been designed guided by the ontology concepts. As the results in chapter 6 show, this framework fails to recognize the events of interest in some cases. This is mainly due to the inaccurate detection of the object of interest. The main problem is that objects are detected as divided in multiple parts (e.g, people divided into several blobs) and therefore it reduces the performance of the high-level processing stages (e.g., people recognition). In this failure cases, the use of contextual information (type of objects to be recognized, mean size of them,...) can increase the success detection rate. The overall event recognition rate is about 64% which is an acceptable level of performance for the evaluation of different scenarios. The main disadvantage is the high detection rate of false alarms (55% of output alarms are false alarms). These failures can not be detected by using only contextual information and we propose to employ feedback strategies for solving them.

Feedback strategies are employed to increase the performance of the event recognition stage. Experimental results show that these strategies maintain the initial results obtained detecting 59% of the labeled events and reducing the percentage of false alarms (45% against 55% in the initial proposed framework). Also, they reduce the overall computation cost modifying the quality levels of the analysis components (38.9 fps against 26.5 fps in the initial proposed framework for 360x288 sequences). As we can see in chapter 6, the improvement of each stage depends on its complexity and computational cost.

In conclusion, this work enhances the performance of a generic visual analysis framework via the use of context information and feedback strategies. The context information is useful for adapting generic processing stages to improve the semantic description obtained. Feedback strategies can be used to increase the hypothesis detection rate and modify the processing stages for reducing computational cost in future runs.

7.3 Future Work

The work described in this document is a step in the direction of the use of context information and feedback strategies to improve the results obtained in the visual analysis process. As it is said above, the proposed problem (event detection) is certainly not fully solved. We identify four main areas for future work:

- **Extension of the proposed ontology.** The event detection ontology described is based on the interaction between blobs (identified as people or generic objects). We propose to extend the ontology with a lower representation of the entities involved in the ontology. For example, the *object* entity can be extended with the hierarchical representation of the parts that compose the object. This extension allows to study a new type of events to be detected in video sequences (like activity recognition or people-object interaction) and incorporate them to the ontology definition. Furthermore, we propose the integration of MPEG-7 descriptions for object visual description and the integration of the feedback schemes (expressing the correlation between concepts and stages) in the ontology.

- **Improvement of analysis components**. As said above, in the initial proposed framework the object extraction process fails in some cases. Thus, we propose to study new algorithms and post-processing techniques to increase the performance of the object extraction process. The object extraction process is one of the critical points in any visual analysis system working at object level and as there are no overall solutions it is still a highly active research area. Therefore we propose to start studying some recently proposals for object extraction (like the combination and restoration operators that enhance the detection of people by using a simple people model, described in [19]). Furthermore, in event detection system the identification of people (or crowds) is another critical task: here we propose to investigate people recognition methods more robust to the different poses that can be found in video sequences. Finally, the extension of the proposed ontology needs the development of new analysis components to identify the different parts that compose a specific object described in the ontology.

- **Study of complex feedback strategies.** This future work area has three main parts. Firstly, we propose to study in depth the relations between the different processing stages and the dependency of the final results with them. This study may serve to know the implications of changing the output quality in some processing stages in the final results. Thus, more complex and specific feedback strategies can be used. Secondly, as said above, we have identified four different approaches for quality variation of the analysis components. We propose to continue studying new approaches for this concept and to provide to the framework a bigger set of algorithms to increase the possibilities of feedback strategies. Finally, we propose to study other application domains with more uncertainty (e.g., action recognition in smart rooms[105]). In these more challenging domains the possibilities for recognizing events are difficult and their definition is more complex, allowing to fully explore the possibilities of feedback strategies.

- **Study of how to use context information described in the ontology**. This work is focused on the study of how scene context can be used to guide the analysis performed on video sequences (camera distance to the recorded scene, average size of objects, relationship between events,...). .

Bibliography

[1] E. Izquierdo, A. Katsaggelos, and M. Strintzis, “Special issue on audio and video analysis for multimedia interactive services,” in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14(5), pp. 569– 571, May 2004.

[2] M. Rincón, E. Carmona, M. Bachiller, and E. Folgado, “Segmentation of moving objects with information feedback between description levels,” in *In Proceding of IWINAC 2007*, 2007.

[3] M. Harville, “A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models,” in *ECCV*, vol. 3, pp. 543–560, 2002.

[4] V. Mezaris, I. Kompatsiaris, and M. G. Strintzis, “Region-based image retrieval using an object ontology and relevance feedback,” in *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 886–901, Hindawi Publishing Corp., 2004.

[5] N. Maillot, M. Thonnat, and A. Boucher, “Towards ontology-based cognitive vision,” in *Machine Vision and Applications (MVA)*, vol. 16 (1), pp. 33–40, Springer-Verlag New York, Inc., 2004.

[6] M. Valera and S. Velastin, “Intelligent distributed surveillance systems: a review,” in *IEE Proceedings on Visual Image Signal Processing*, vol. 152, pp. 192–204, 8 April 2005.

[7] T. Meier, “Segmentation for video object plane extraction and reduction of coding artifacts,” in *Ph.D Thesis, Dept. Of Electrical and Electronic Egineering, The University of Western Australia*, 1998.

[8] D. Zhang and G. Lu, “Segmentation of moving objects in image sequence: A review,” in *IEEE Trans. on Circuits, Systems and Signal Processing (Special Issue on Multimedia Communication Services)*, vol. 2, pp. 143–183, 2001.

[9] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa, “A system for video surveillance and monitoring,” in *VSAM Final Report, Carnegie Mellon University, 2000.*, 2000.

[10] G. Havely and D. Weinshall, “Motion of disturbances: Detection and tracking of multi-body non-rigid motion,” in *Computer Vision and Pattern Recognition, IEEE Computer Society, Puerto Rico*, pp. 897–902, 1997.

[11] C. Stauffer and W. Grimson, "Learning patterns of activity using real-time tracking," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 747–757, 2000.

[12] A. M. McIvor, "Background subtraction techniques," in *In Proc. of Image and Vision Computing, Auckland, New Zealand*, 2000.

[13] M. Cristiani, M. Bicego, and V. Murino, "Multi-level background initialization using hidden markov models," in *In First ACM SIGMM Int. Workshop on Video Surveillance*, vol. 1, 2003.

[14] S. Cheung and C. Kamath, "Robust techniques for background subtraction in urban traffic video," in *Proc. SPIE Video Communications and Image Processing*, 2004.

[15] M. Piccardi, "Background subtraction techniques: a review," in *Proc. IEEE Intl. Conf. on Systems, Man and Cybernetics,*, 2004.

[16] S. Y. Elhabian, K. M. El-Sayed, and S. H. Ahmed, "Moving object detection in spatial domain using background removal techniques-state-of-art," in *Recent Paterns on Computer Science*, vol. 1(1), 2008.

[17] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts and shadows in video streams," in *IEEE Trans. on Patt. Anal. and Machine Intell.*, vol. 25, pp. 1337–1342, 2003.

[18] C. Wren, A. Azarhayejani, T. Darrell, and A. Pentland, "Pfinder: real-time tracking of the human body," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19(7), pp. 780–785, 1997.

[19] E. Carmona, J. Martinez-Cantos, and J.Mira, "A new video segmentation method of moving objects based on blob-level knowledge,," in *Pattern Recognition Letters*, vol. 3, pp. 272–285, 2008.

[20] A. Elgammal, D. Hanvood, and L. Davis, "Nonparametric model for background subtraction," in *Proc. ECCV*, pp. 751–767, 2000.

[21] D. Scott, "Multivariate density estimation. theory, practice and visualization.," in *John Wiley & Sons, Inc*, 1992.

[22] N. M. Oliver, B. Rosario, and A. P. Pentland, "A bayesian computer vision system for modeling human interactions," in *EEE Trans. on Patt. Anal. and Machine Intell*, vol. 22, pp. 831–843, 2000.

[23] A. Prati, I. Mikic, Mohan, M. Trivedi, and R. Cucchiara, "Comparative analysis of moving shadow detection algorithms," in *Image and Vision Computing Journal (special issue on Visual Surveillance)*, 2003.

[24] T. H. D. Harwood and L. Davis, "A statistical approach for realtime robust background subtraction and shadow detection," in *In Proc. of IEEE Frame Rate Workshop (Kerkyra, Greece)*, pp. 1–19, 1999.

[25] D. Xu, J. Liu, X. Li, Z. Liu, and X.Tang, "Insignificant shadow detection for video segmentation," in *IEEE Trans. Circuits Syst. Video Technol*, vol. 15, pp. 1058–1064, 2005.

[26] I. Haritaoglu, "A real time system for detection and tracking of people and recognizing their activities," in *PhD thesis, University of Maryland at College Park*, 1998.

[27] J. Stauder, R. Mech, and J. Ostermann, "Detection of moving cast shadows for object segmentation," in *IEEE Transactions onMultimedia*, vol. 1, pp. 65–76, 1999.

[28] D. Duque, H. Santos, and P. Cortez, "Moving object detection unaffected by cast shadows, highlights and ghosts,," in *IEEE Int. Conf. On Image Processing (ICIP)*, vol. 1, pp. 413–416, 2005.

[29] M.-T. Yang, K.-H. Lo, C.-C. Chiang, and W.-K. Tai, "Moving cast shadow detection by exploiting multiple cues," in *IET Image Processing*, vol. 2 of *95-104*, 2008.

[30] R. Jain, R. Kasturi, and B. Schunk, "Machine vision," in *McGraw-Hill Inc.*, pp. pp. 63– 69.

[31] P. Salembier and J. Ruiz, "On filters by reconstruction for size and motion simplification," in *Proc. of Int. Symposium in Mathematical Morphology*, pp. 425–434, 2002.

[32] E. Sánchez-Nielsen and M. Hernández-Tejera., "Tracking moving objects using the hausdorff distance. a method and experiments," in *Frontiers in Artificial Intelligence and Applications: Pattern Recognition and Applications*, pp. 164–172, 2000.

[33] R. J. Schalkoff, "Digital image processing and computer vision," in *Editorial John Wiley & Sons*, pp. 213–218, 1989.

[34] B. Horn, "Robot vision," in *MIT Press, Cambridge, MA, & McGrawn-Hill Book Company, New York.*, 1986.

[35] M. Wessler and L. Stein, "Robust active vision from simple symbiotic subsystems," in *echnical Report, MIT,1997*, 1997.

[36] P. Anandan., "A computacional cuadrowork and an algorithm for the measurement of visual motion," in *International Journal of Computer Vision*, vol. 2(3), pp. 283–310, 1989.

[37] S. A. Brock-Gunn, G. Dowling, and T. J. Ellis, "Tracking using colour information," in *Technnical Report TCU.CS.1994.7*, 1994.

[38] M. Kass, A. Witkin, and Terzopoulos, "Snakes: Active contour models," in *International Journal of Computer Vision*, vol. 1(4), pp. 133–144, 1987.

[39] S. Gill, R. Milanese, and T. Pun, "Feature selection for object tracking in traffic scenes," in *In SPIE International Symposium on Smart Highways*, 1994.

[40] R. Cutler and L. S. Davis, "Robust real-time periodic motion detection, analysis, and applications," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(8), pp. 781–796, 2000.

[41] M. K. Leung and Y.-H. Yang., “First sight: A human body outline labeling system.,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17(4), pp. 359–377, 1995.

[42] M.Hessein, “Real-time human detection, tracking, and verification in uncontrolled camera motion environments,” in *IEEE Proceedings of International Conference on Vision Systems*, pp. 41–47, 2006.

[43] B. Wu and R. Nevatia, “Detection of multiple, partially occluded humans in single image by bayesian combination of edgeless part detector,” in *Proc. of 10th IEEE International Conference on Computer Vision*, pp. pp. 90–97., 2005.

[44] F. Xu and D. Fujimura, “Human detection using depth and gray images,” in *Proceedings of IEEE Advanced Video and Signal Based Surveillance*, pp. 115–121, 2003.

[45] R. I. Haritaoglu, D. Harwood, and L. S. Davis, “W4: Real-time surveillance of people and their activities,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(8), pp. 809–830, 2000.

[46] B. F. Cupillard, F. and and M. Thonnat, “Behaviour recognition for individuals, groups of people and crowd,” in *Proceedings of the 1st Workshop on Intelligent Distributed Surveillance Systems*, 2003.

[47] C. Castel, L. Chaudron, and C. Tessier, “What is going on? a high level interpretation of sequences of images,” in *Proceedings of the 4th European Conference on Computer Vision*, pp. 17–23, 1996.

[48] V. D. Shet, D. Harwood, and L. S. Davis, “Top-down, bottom-up multivalued default reasoning for identity maintenance,” in *International Multimedia Conference Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, pp. 79 – 86, 2006.

[49] Y. Ivanov and A. Bobick, “Recognition of visual activities and interactions by stochastic parsing,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(8), pp. 852–872, 2000.

[50] T. Vu, F. Brémond, and M. Thonnat, “Temporal constraints for video interpretation.,” in *In Proceedings of the 16th European Conference on Artificial Intelligence*, 2002.

[51] T. Vu, F. Brémond, and M. Thonnat, “Automatic video interpretation: a novel algorithm for temporal scenario recognition,” in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, vol. 1, pp. 1295–1302, 2003.

[52] T. Vu, F. Brémond, and M. Thonnat, “Automatic video interpretation: a recognition algorithm for temporal scenarios based on pre-compiled scenario models,” in *Proceedings of the 3rd International Conference on Computer Vision Systems*, vol. 1, pp. 523–533.

[53] M. Brand, N. Oliver, and A. Pentland, “Coupled hidden markov models for complex action recognition,” in *In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR’97)*, vol. 1, pp. 994–999, 1997.

[54] M. Rosenblum, Y. Yacoob, and L. Davis, "Human emotion recognition from motion using a radial basis function network architecture.," in *Proceedings of the IEEE Workshop on Motion of Non-Rigid Articulated Objects*, 1994.

[55] C. Lin, H. Nein, and W. Lin, "A space-time delay neural network for motion recognition and its application to lipreading," in *nternational Journal of Neural Systems*, vol. 9(4), pp. 311–334, 1999.

[56] F. Lv, X. Song, B. Wu, V. K. Singh, and R. Nevatia, "Left-luggage detection using bayesian inference," in *9th Intl. Workshop on Performance Evaluation of Tracking and Surveillance (PETS-CVPR'06)*, pp. 83–90, 2006.

[57] Z. D., G.-P. D., B. S., M. I., and L. G., "Modeling individual and group actions in meetings: a two-layer hmm framework," in *IEEE Workshop on Event Mining at the Conference on Computer Vision and Pattern Recognition*, 2004.

[58] A. A. Efros, A. C. Berg, G. Mori, and J. Malik., "Recognizing action at a distance," in *IEEE International Conference on Computer Vision*, pp. 726–733, 2003.

[59] A. Yilmaz and M.Shah. in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 984–989, 2005.

[60] I. Laptev and T. Lindeberg, "Space-time interest points," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 432–439, 2003.

[61] C. Rao, A. Yilmaz, and M. Shah, "View-invariant representation and recognition of actions," in *International Journal on Computer Vision*, pp. 203–226, 2002.

[62] F. Lv and R. Nevatias, "Recognition and segmentation of 3-d human action using hmm and multi-class adaboost," in *European Conference on Computer Vision*, pp. 359–372, 2006.

[63] W. is an Ontology, "http://ksl-web.stanford.edu/kst/what-is-an-ontology.html," 2000.

[64] F. Bremond, N. Maillot, M. Thonnat, and V. Vu, "Ontologies for video events," in *Technical report, INRIA, no 5189 (http://www.inria.fr/rrrt/rr-5189.html)*, 2004.

[65] M. Srikanth, J. Varner, M. Bowden, and D. Moldovan, "Exploiting ontologies for automatic image annotation," in *Proc. of the Twenty-Eighth Annual International ACM Conference on Research and Development in Information Retrieval ACM SIGIR*, pp. 552–558, 2005.

[66] N. Voisine, S. Dasiopoulou, F. Precioso, V. Mezaris, I. Kompatsiaris, and M. Strintzis, "A genetic algorithm-based approach to knowledge-assisted video análisis," in *Proc. ICIP*, vol. 3, pp. 441–444, 2005.

[67] C. Town, "Ontological inference for image and video analysis," in *Machine Vision and Applications*, vol. 17(2), pp. 94–115, 2006.

[68] W. M. S. I. G. Wiki:, "http://www.w3.org/2005/incubator/mmsem/wiki/frontpage,"

[69] S. Dasiopoulou, S. Dasiopoulou, V. Mezaris, I. Kompatsiaris, V.-K. Papastathis, and M. Strintzis, "Knowledge-assisted semantic video object detection," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, pp. 1210–1224, 2005.

[70] B. Bolles and R. Nevatia, "A hierarchical video event ontology in owl," in *ARDA Challenge Workshop Report (https://rrc.mitre.org//nwrrc/OWLevents-final-report.pdf)*, 2004.

[71] A. François, R. Nevatia, J. Hobbs, and R. Bolles, "Verl: An ontology framework for representing and annotating video events," in *IEEE Multimedia*, vol. 12, pp. 76–86, 2005.

[72] A. O. M. Arens and H.-H. Nagel, "Using behavioral knowledge for situated prediction of movements," in *In Proc. 27th German Conference on Artificial Intelligence (KI-2004)* (Springer, ed.), vol. LNAI 3238, pp. 141–155, 2004.

[73] G. Medioni, I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia, "Event detection and analysis from video streams," vol. 23, pp. 873–889, Aug. 2001.

[74] M. Beynon, D. Van Hook, M. Seibert, A. Peacock, and D. Dudgeon, "Detecting abandoned packages in a multi-camera video surveillance system," in *Proc. IEEE Conference on Advanced Video and Signal Based Surveillance*, pp. 221–228, 21–22 July 2003.

[75] A. Cavallaro, O.Steiger, and T. Ebrahimi, "Semantic video analysis for adaptive content delivery and automatic description," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15(10) of *1200-1209*, 2005.

[76] J. Orwell, P. Remagnino, and G. Jones, "From connected components to object sequences," in *In Wkshp. on Perf. Evaluation of Tracking and Surveillance.*, 2000.

[77] A. Mittal and D. Huttenlocher, "Scene modeling for wide area surveillance and image synthesis," in *In CVPR*, 2000.

[78] M. Mirmehdi, P. L. Palmer, J. Kittler, and H. Dabis, "Multi-pass feedback control for object recognition," in *Procs. of Vision Interface*, pp. 49–56, 1996.

[79] J. K. H. D. M. Mirmehdi, P.L. Palmer, "Complex feedback strategies for hypothesis generation and verification," in *Proc. of the 7th British Machine Vision Conference*, vol. 1, pp. –, 1996.

[80] M. Mirmehdi, P. Palmer, J. Kittler, and H. Dabis, "Feedback control strategies for object recognition," in *IEEE Transactions On Image Processing*, vol. 8, pp. 1084–1101, Aug. 1999.

[81] B. N. L. Hotz, K. Terzic, and J. Dochman, "Feedback between low-level and high-level image processing," in *TR FBI-B-278/07, Department of Informatics, University of Hamburg*, 2007.

[82] J. Martínez-Cantos, E. Carmona, A. Fernández-Caballero, and M. T. López, "Parametric improvement of lateral interaction in accumulative computation in motion-based segmentation," in *4th International Symposium on Neural Networks: Algorithms and Applications*, vol. 71, pp. 776–786, 2006.

[83] A. Garcia and J. Bescos, "Video object segmentation based on feedback schemes guided by a low-level scene ontology," in *AICVS*, 2008.

[84] Q. Zhou, L. Ma, and D. Chelberg, "Adaptive object detection and recognition based on a feedback strategy," in *IVC(24)*, vol. 1, pp. 80–93, 2006.

[85] E. Erdem, A. Tekalp, and B. Sankur, "Video object tracking with feedback of performance measures,," in *CirSysVideo(13)*, vol. 4, pp. 310–324, 2003.

[86] L. Taycher, J. W. Fisher, III, and T. J. Darrell, "Incorporating object tracking feedback into background maintenance framework,," in *Motion05*, vol. 2, pp. 120–125, 2005.

[87] L.-Q. Xu, "An efficient object segmentation algorithm with dynamic and selective background updating and shadow removal.," in *IEEE International Conference on Image Processing*, pp. 2393–2396, 2006.

[88] "The protégé project, university of stanford," in *http://protege.stanford.edu/*, 2000.

[89] J. Martínez, "Mpeg-7 overview (version 10), iso/iec jtc1/sc29/wg11n6828.," in *70th MPEG meeting Palma de Mallorca*, 2004.

[90] J. SanMiguel, J. Bescós, J. M. Martínez, and . García., "Diva: a distributed video analysis framework applied to video-surveillance systems," in *9th International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2008.*, vol. 1, pp. 207–211, 2008.

[91] Y. Shan, F. Yang, and R. Wang, "Color space selection for moving shadow elimination," in *ICIG '07: Proceedings of the Fourth International Conference on Image and Graphics*, pp. 496–501, 2007.

[92] N. Herodotou, K. Plataniotis, and A. Venetsanopoulos, "A color segmentation scheme for object-based video coding," in *Proceedings of the IEEE Symposium on Advances in Digital Filtering and Signal Processing*, vol. 1, pp. 25–29, 1998.

[93] S. Hongeng, R. Nevatia, and F. Bremond, "Video-based event recognition: activity representation and probabilistic recognition methods," 2004.

[94] S. Hongeng, F. Bremond, and R. Nevatia, "Bayesian framework for video surveillance application," in *Proc. 15th International Conference on Pattern Recognition*, vol. 1, pp. 164–170, 3–7 Sept. 2000.

[95] C. O'Conaire, N. O'Connor, and A. Smeaton, "Detector adaptation by maximising agreement between independent data sources," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–6, 2007.

[96] C. O'Conaire, N. O'Connor, E. Cooke, and A. Smeaton, "Detection thresholding using mutual information," in *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2006.

[97] M. Comer and E. Delp, "Multiresolution image segmentation," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2415–2418, 1995.

[98] C. Erdem, A. Murat Tekalp, and B. Sankur, "Metrics for performance evaluation of video object segmentation and tracking without ground-truth," in *IEEE Proceedings of International Conference on Image Processing*, vol. 2, pp. 69–72, 2001.

[99] A. Leone, C. Distante, and F. Buccolieri, "A texture-based approach for shadow detection," in *IEEE Conference on Advanced Video and Signal Based Surveillance*, vol. 1(1), pp. 371–376, 2005.

[100] M. Kendall, "A new measure of rank correlation.," in *Biometrika*, vol. 30(1-2), pp. 81–93, 1938.

[101] D. H. I. Haritaoglu and L. Davis, "Ghost: A human body part labelling system using silhouettes," in *Proceedings of International Conference on Pattern Recognition"*, pp. 77–82, 1998.

[102] V. Fernández-Carbajales, M. A. García, and J. M. Martínez, "Robust people detection by fusion of evidence from multiple methods," in *9th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 2008.

[103] J. SanMiguel and J. M. Martínez, "Robust unattended and stolen object detection by fusing simple algorithms," in *IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance*, pp. 18–25, 2008.

[104] D. Doermann and D. Mihalcik, "Tools and techniques for video performances evaluation," in *IEEE Int. Conference on Pattern Recognition*, vol. 1(1), pp. 167–170, 2000.

[105] Z. Yu, M. Ozeki, Y. Fujii, and Y. Nakamura, "Towards smart meeting: enabling technologies and a real-world application," in *ICMI '07: Proceedings of the 9th international conference on Multimodal interfaces*, pp. 86–93, 2007.

Publications

Part of this work has produced the following publications:

- Juan Carlos San Miguel, Jesús Bescós, José M. Martínez y Álvaro García, "*DiVA: a Distributed Video Analysis framework applied to video-surveillance systems*", in 9th International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS'2008, Klagenfurt, Austria, May 2008. pp. 207-211 (ISBN 978-0-7695-3130-4)

- Juan Carlos San Miguel, José M. Martínez, "*Robust unattended and stolen object detection by fusing simple algorithms*", in Proceedings of the 2008 IEEE International Conference on Advanced Video and Signal based Surveillance, AVSS'2008, Santa Fé, United States, September 2008. pp. 18-25 (ISBN 978-0-7695-3422)

List of Abbreviations

BG	BackGround
Blob	Binary Large Object
CCA	Connected Component Analysis
CPU	Central Processing Unit
DBN	Dynamic Bayesian Network
DL	Description Logic
FG	ForeGround
FSA	Finite State Automata
GM	Gaussian Model
GMM	Gaussian Mixture Model
GRA	Gaussian Running Average
HMM	Hidden Markov Model
HTPP	HyperText Transport Protocol
KDE	Kernel Density Stimator
MoG	Mixture of Gaussians
MPEG-7	Motion Picture Experts Group 7
MRF	Markov Random Field
MVO	Moving Visual Object
NN	Neural Network
OWL	Ontological Web Language
PCA	Principal Component Analysis
PDF	Probability Density Function
RDFS	Resource Description Framework Schema
ROI	Region Of Interest
SVM	Support Vector Machine
URL	Universal Resource Locator
XML	eXtensible Markup Language

Appendix A

Ontology for video event detection

A.1 Basic ontology

In this section the entities that compose the ontology proposed in this work are completely described.

A.1.1 Scene

The *Scene* entity is the physical space where a real world event occurs. In Figure A.1we can see the whole description of the *Scene* entity

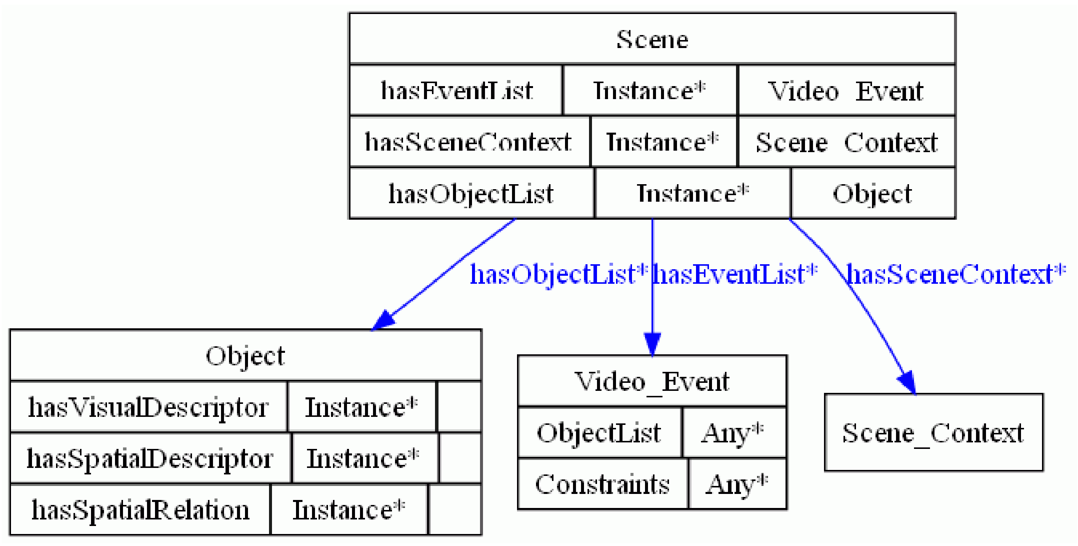


Figure A.1: Class Hierarchy of the *Scene* entity of the Ontology

A.1.2 System

The *System* entity represents the system that uses the ontology. In Figure A.2 we can see the description of it:

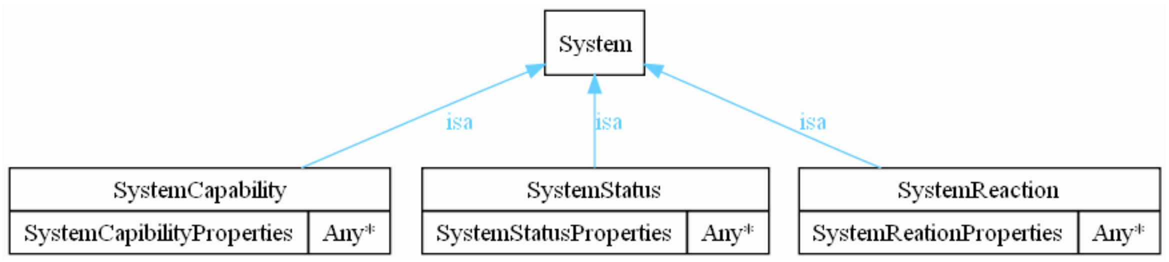


Figure A.2: Class Hierarchy of the *System* entity of the Ontology

A.1.3 Object

The *Object* concept represents any real world object observed by the camera. In Figure A.3 we can see the description of it:

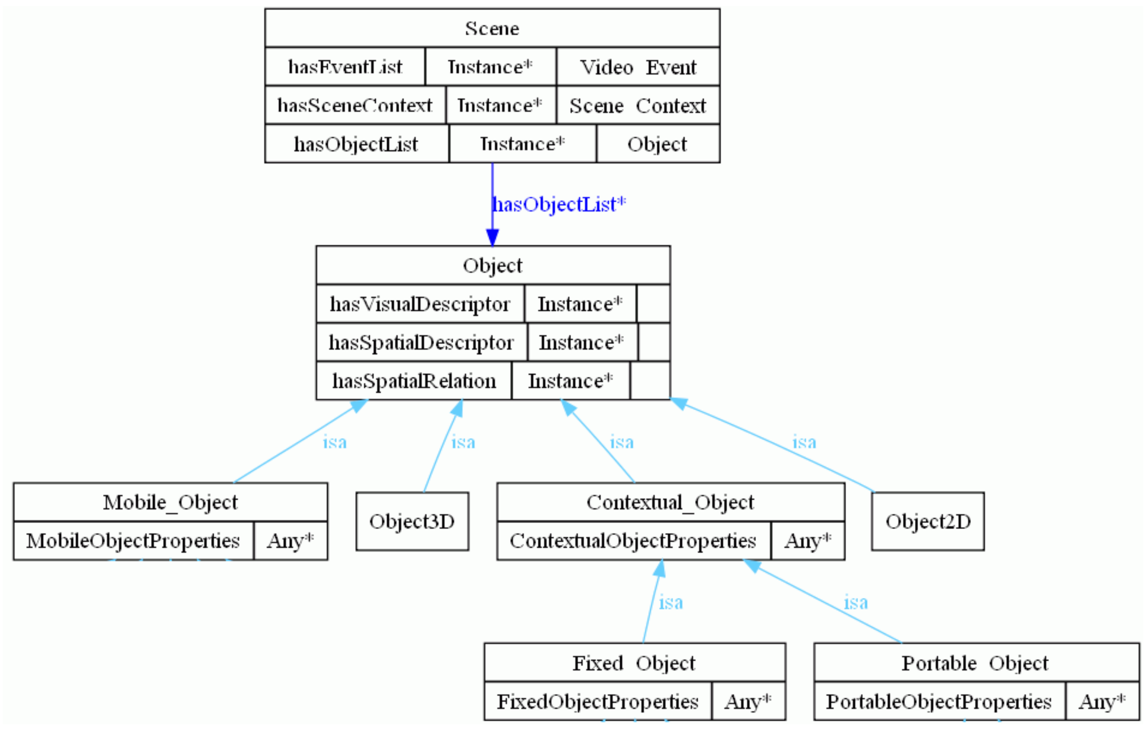


Figure A.3: Class Hierarchy of the *Object* entity of the Ontology

A.1.4 Event

The *Event* entity is a generic term to describe any event, action or activity that happens in the scene. In Figure A.4 we can see the description of this entity. The acronyms used for the event classes are:

- *Simple_SingleObject* events (SSE_T and SSE_I for Transitive/Intransitive events)
- *Simple_MultipleObject* events (SME_T and SME_I for Transitive/Intransitive events)
- *Complex_SingleObject* events (CSE_T and CSE_I for Transitive/Intransitive events)
- *Complex_MultipleObject* events (CME_T and CME_I for Transitive/Intransitive events)

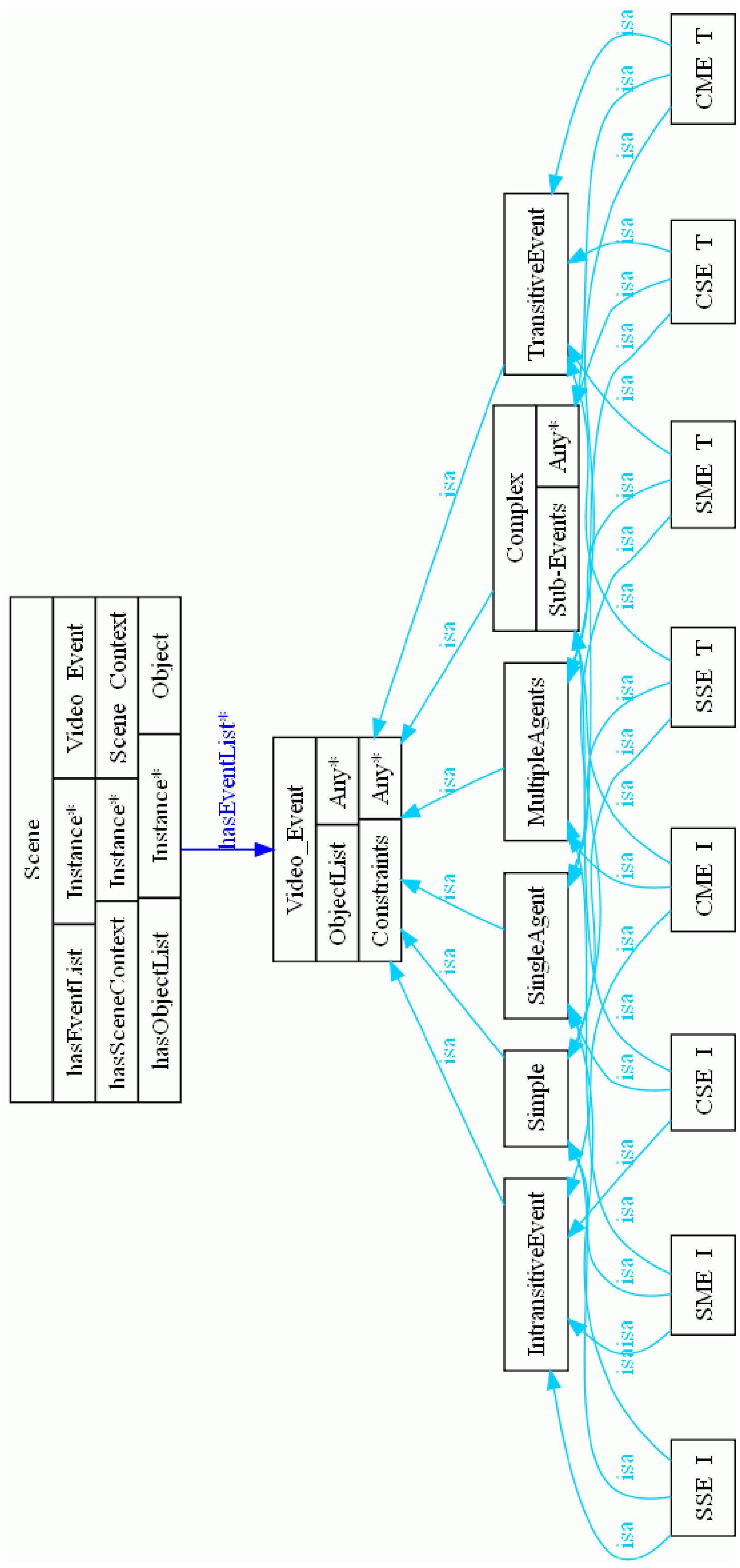


Figure A.4: Class Hierarchy of the *Event* entity of the Ontology

A.2 Underground Video surveillance domain

In this section the ontology described in section 3 is particularized for the underground video surveillance domain. In the following subsections the *Object* and *Event* entities are shown in Figures A.5 and A.6. Moreover, some definitions of the proposed events are described in last subsection.

A.2.1 Class hierarchy (Object and Event)

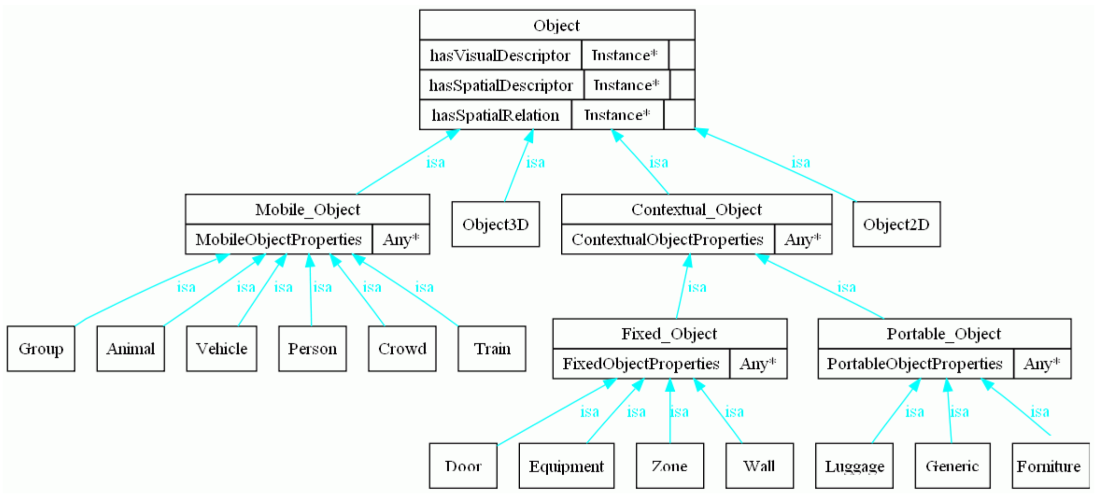


Figure A.5: Object Ontology for the Underground Video Surveillance domain

A.2.2 Event definitions

A.2.2.1 Simple and single events (SSE)

SSE_SuveillanceEvent Inside_zone
  Object_list: ((o: Object), (z1: Zone))
  Constrains: (o in z)

SSE_SuveillanceEvent Speed_increase
  Object_list: (p: Person)
  Constrains: (IncreaseSpeed(p) is true)

SSE_SuveillanceEvent Stopped
  Object_list: (o: Object)
  Constrains: (speed of o < minspeed)

SSE_SuveillanceEvent Stopped
  Object_list: (o: Object)
  Constrains: (speed of o < minspeed)

SSE_SuveillanceEvent Trajectory_variation
  Object_list: (mo: Mobile Object)
  Constrains: (Trajectory (mo) > significanttrajectoryvariation)

A.2.2.2 Simple and multiple events (SME)

SME_SuveillanceEvent Group_quicksplit
  Object_list: ((g: Group))
  Constrains: (Split (o) > quicksplit)

SME_SuveillanceEvent Owner_distance_too_far
  Object_list: ((p: Person), (co: ContextualObject))
  Constrains: (distance(p,co) > distanceTh)

SME_SuveillanceEvent many_object_inside_a_zone
  Object_list: ((o1:object1), (o2:object2),...,(on:objectN))
  Constrains: (inside_zone(o1) and inside_zone(o2) ... and inside_zone(on))

A.2.2.3 Complex and single events (CSE)

CSE_SuveillanceEvent Put_Object

Object_list: ((p: Person), (co: Mobile Object))

Sub-events:

e1: quicksplit(p, co)

e2: DetectedAsForegroundObject(co)

e3: p is owner of co

e4: Owner_distance_too_far (p, co)

Constrains:

e1 and e2 and e3 and e4

CSE_SuveillanceEvent Changes_zone

Object_list: ((mo: Mobile Object), (z1: zone), (z2. zone))

Sub-events:

e1: SSE Inside_zone(mo, z1)

e2: SSE Inside_zone(mo, z2)

Constrains:

//Sequence (e1; e2)

CSE_SuveillanceEvent jumping

Object_list: (p: Person)

Sub-events:

e1: SSE Speed_increase (p)

e2: SSE Legs_up(p)

Constrains:

//Sequence (e1; e2)

A.2.2.4 Complex and multiple events (CME)

CME_SuveillanceEvent Abandoned_object

Object_list: ((p: Person), (co: ContextualObject)

Sub-events:

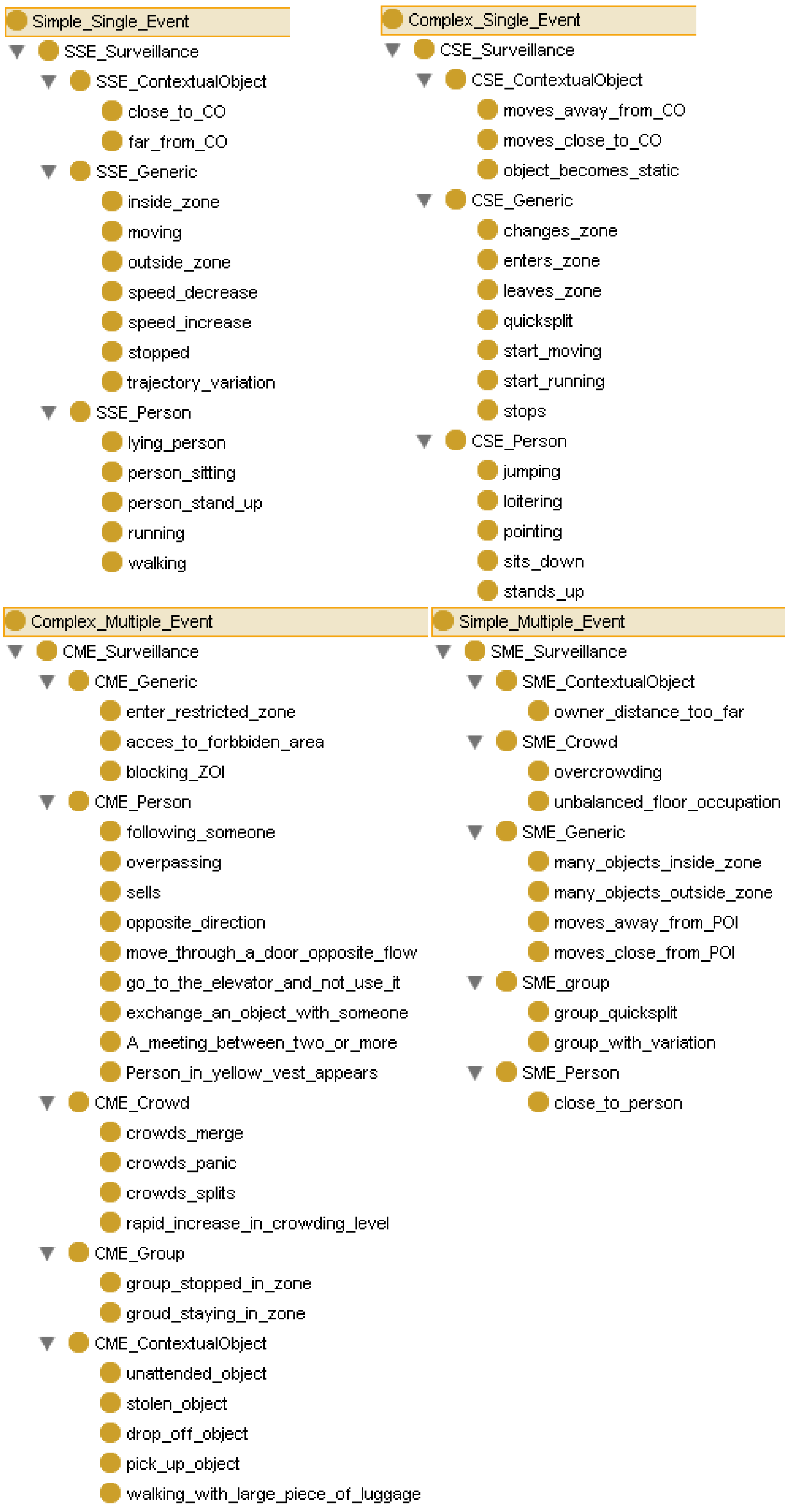
e1: Put_Object (p, co)

e2: BecomeStationary(co)

e3: Owner_distance_too_far (p, co)

Constrains:

e1 before e2 before e3



- Simple_Single_Event
  - SSE_Surveillance
    - SSE_ContextualObject
      - close_to_CO
      - far_from_CO
    - SSE_Generic
      - inside_zone
      - moving
      - outside_zone
      - speed_decrease
      - speed_increase
      - stopped
      - trajectory_variation
    - SSE_Person
      - lying_person
      - person_sitting
      - person_stand_up
      - running
      - walking

- Complex_Single_Event
  - CSE_Surveillance
    - CSE_ContextualObject
      - moves_away_from_CO
      - moves_close_to_CO
      - object_becomes_static
    - CSE_Generic
      - changes_zone
      - enters_zone
      - leaves_zone
      - quicksplit
      - start_moving
      - start_running
      - stops
    - CSE_Person
      - jumping
      - loitering
      - pointing
      - sits_down
      - stands_up

- Complex_Multiple_Event
  - CME_Surveillance
    - CME_Generic
      - enter_restricted_zone
      - acces_to_forbbiden_area
      - blocking_ZOI
    - CME_Person
      - following_someone
      - overpassing
      - sells
      - opposite_direction
      - move_through_a_door_opposite_flow
      - go_to_the_elevator_and_not_use_it
      - exchange_an_object_with_someone
      - A_meeting_between_two_or_more
      - Person_in_yellow_vest_appears
    - CME_Crowd
      - crowds_merge
      - crowds_panic
      - crowds_splits
      - rapid_increase_in_crowding_level
    - CME_Group
      - group_stopped_in_zone
      - groud_staying_in_zone
    - CME_ContextualObject
      - unattended_object
      - stolen_object
      - drop_off_object
      - pick_up_object
      - walking_with_large_piece_of_luggage

- Simple_Multiple_Event
  - SME_Surveillance
    - SME_ContextualObject
      - owner_distance_too_far
    - SME_Crowd
      - overcrowding
      - unbalanced_floor_occupation
    - SME_Generic
      - many_objects_inside_zone
      - many_objects_outside_zone
      - moves_away_from_POI
      - moves_close_from_POI
    - SME_group
      - group_quicksplit
      - group_with_variation
    - SME_Person
      - close_to_person

Figure A.6: Event Ontology for the Metro/Underground Surveillance domain

Appendix B

ViPER toolkit

In order to evaluate a video analysis algorithm, or a set of algorithms, it is necessary to define a methodology. The ViPER Toolkit (http://viper-toolkit.sourceforge.net) provides a general framework for evaluation. In this appendix the ground truth annotation and performance evaluation stages using the ViPER toolkit are discussed.

B.1 ViPER-GT

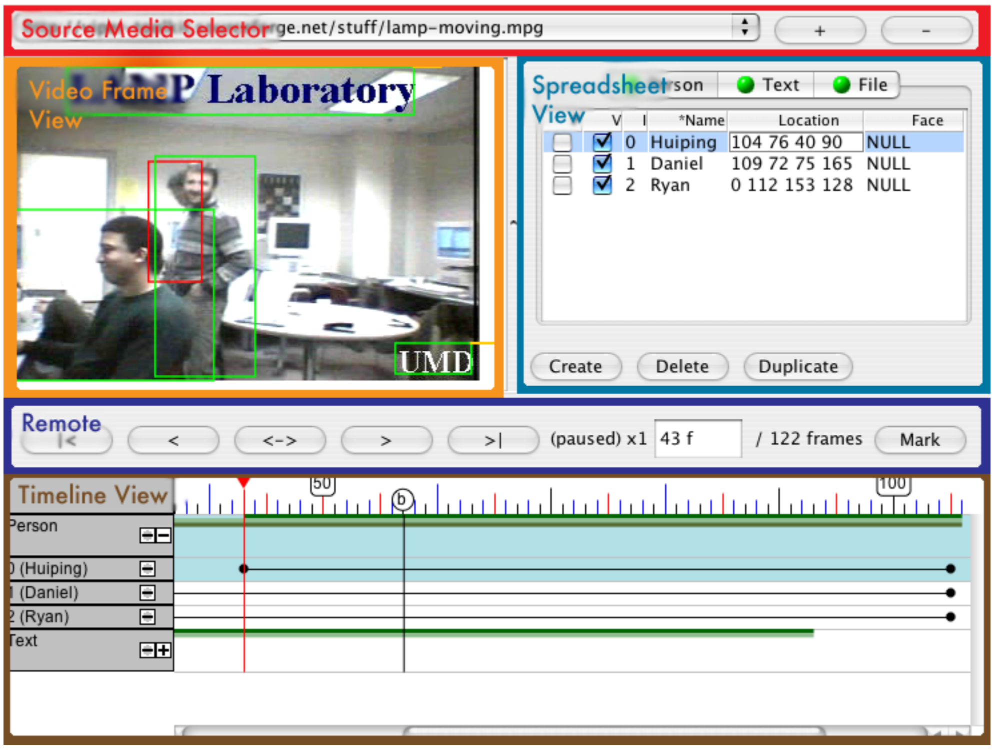
B.1.1 Ground Truth Annotation user interface

ViPER-GT is the tool for creation and editing metadata using a Java graphical user interface. It is designed to allow frame-by-frame markup of video metadata stored in the Viper format. For more information, see the ViPER-GT product page (http://viper-toolkit.sourceforge.net/products/gt/).

In Figure B.1 we can see an example of the ViPER-GT user interface. At the top of the frame is a pull-down menu that shows the name of the currently loaded video file; this panel, the source media selector, also allows the user to edit the list of described media files. The video frame view is in the upper-left quadrant of the screen; this displays the video with spatial annotations. To the right of the video frame is the spreadsheet view, which displays the annotations as a table. Beneath these two views of the data is the timeline view, which displays summary of the video annotation, indicating when descriptors are marked as valid.

B.1.2 Creating Schema Descriptions

ViPER-GT also provides a graphical tool to create descriptions. A descriptor can be: a record describing some element of the video, an object that conforms to a user



ge.net/stuff/lamp-moving.mpg

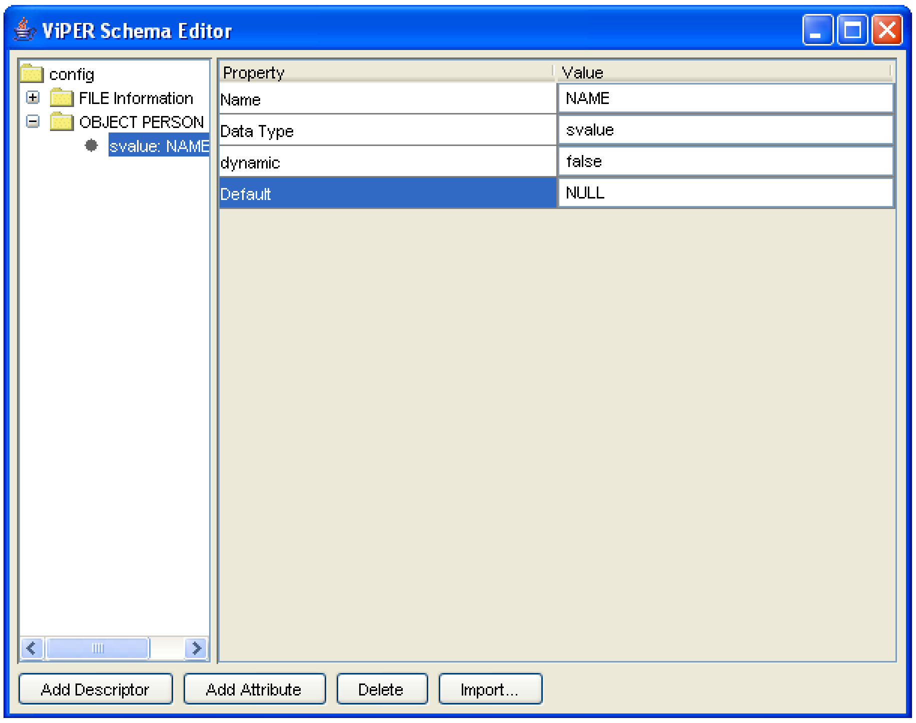
V	I	*Name	Location	Face
✓	0	Huiping	104 76 40 90	NULL
✓	1	Daniel	109 72 75 165	NULL
✓	2	Ryan	0 112 153 128	NULL

(paused) x1 43 f / 122 frames

Figure B.1: ViPER-GT user interface example

defined schema and it is composed of attributes. We have different types of descriptors (*File, Content,* and *Object). File* descriptor reflects the video as a whole, or other metadata about the video, such as file format and frame rate. *Content* descriptions describe metadata that may only occur one at a time. Each instance of this type has a time span and a set of attributes. *Object* description refers to an object that may have many instances at any given time, and whose instances may change over time. (e.g., the events to detect). In Figure we can see an example of the interface.

In the following table we can see a description of the basic attribute data types:



config
FILE Information
OBJECT PERSON
svalue: NAME

Property	Value
Name	NAME
Data Type	svalue
dynamic	false
Default	NULL

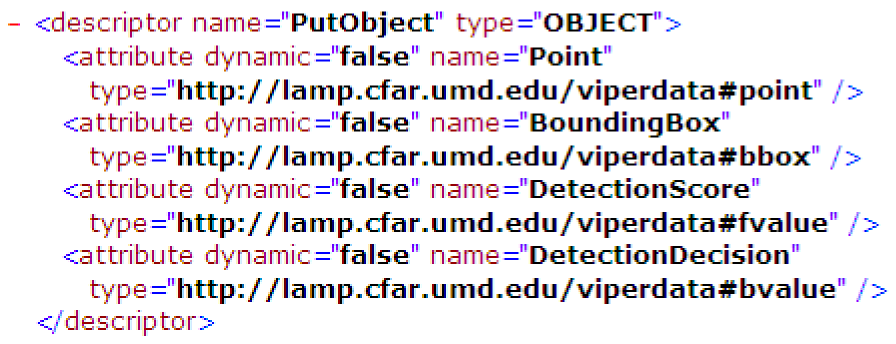
Figure B.2: Edition of the schema description

Attribute	**Description**
bbox	A bounding box (rectangle on the image)
bvalue	A boolean value
circle	A circle, in terms of center point and radius
ellipse	An ellipse, in terms of its bounding box
fvalue	Enumeration type
obox	An oriented bounding box
point	Some specific pixel in the image
polygon	A polygon, given as a list of points
relation	A set of object identification numbers to a certain type of descriptors
svalue	A string value

Table B.2: Basic Attribute Data types in Viper

B.1.3 Descriptor designed for this work

In the Figure B.3 we can see the descriptor for representing the events annotated in the ground truth and detected by the proposed framework. This descriptor is composed of four parts: a point (obtained as the centroid of the bounding box), a bounding box (the area where the event takes place), a detection score (a real value indicating the likelihood of the event) and final detection decision (indicating if the event is detected or not).



```
- <descriptor name="PutObject" type="OBJECT">
    <attribute dynamic="false" name="Point"
      type="http://lamp.cfar.umd.edu/viperdata#point" />
    <attribute dynamic="false" name="BoundingBox"
      type="http://lamp.cfar.umd.edu/viperdata#bbox" />
    <attribute dynamic="false" name="DetectionScore"
      type="http://lamp.cfar.umd.edu/viperdata#fvalue" />
    <attribute dynamic="false" name="DetectionDecision"
      type="http://lamp.cfar.umd.edu/viperdata#bvalue" />
  </descriptor>
```

Figure B.3: Event Descriptor designed

B.2 ViPER-PE

B.2.1 Overview of the ViPER-PE tool

ViPER-PE is a tool for running video evaluations, comparing video metadata against ground truth (namely Object analysis). It is designed to allow for the maximum flexibility in designing experiments, and does not impose many standards as to how such evaluations should be run. Its manual provides some greater depth as to how an evaluation works. It is designed to work in concert with other tools in the ViPER toolbox (ViPER-GT).

Object analysis attempts to match candidate objects to target objects. Object Analysis is divided into multiple phases:

- **Detection**. In the Detection phase, ViPER-PE looks at descriptor types and the frames in which the objects occur. If a candidate overlaps the appropriate number of frames as any target descriptor, then both candidate and target are counted as detected.

- **Localization**. It works much like detection. Instead of looking at all frames of the objects, it compares only those frames where the attributes are similar. Attribute similarity, like frame range similarity, is user-defined. For a given candidate-target pair, dissimilar frames are counted as missed or false in the range distance calculation.

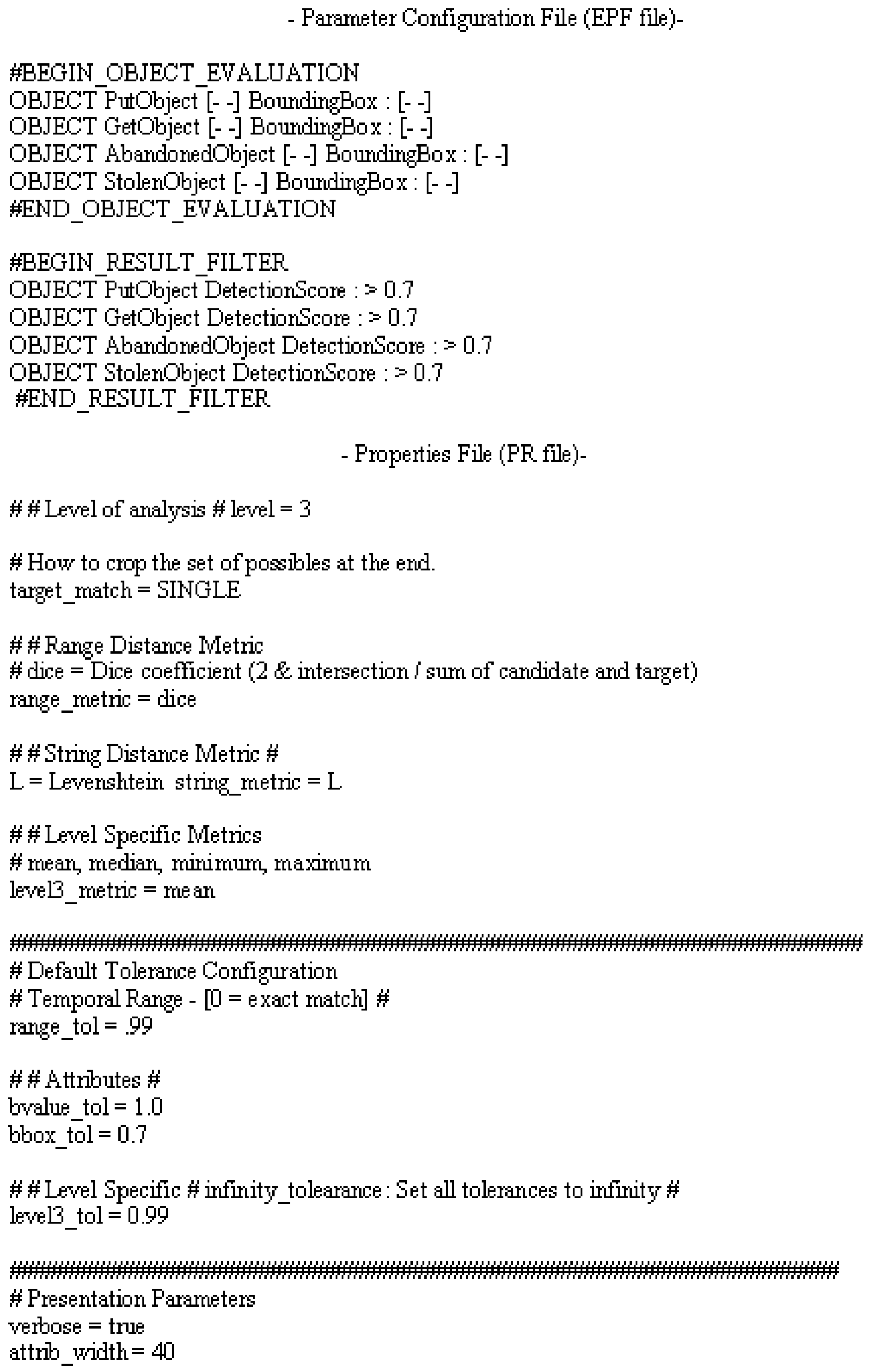
- **Statistical comparison**. From the localized matches, it computes the average, median, minimum, or maximum distance, and then thresholds that value against another tolerance value.
- **Target Matching**. After these different measures, ViPER-PE has a list of candidates that match targets. Many candidates may match a single target, and vice versa. While this may be perfectly acceptable, it is often not enough to leave it at this. There are many situations where having two objects match one do not make sense. Allowing only one candidate for each target is a possible solution. For many cases, such as text detection, a scheme of splitting or merging descriptors so that only appropriate multiple groupings are made may be preferable. Since both have their use, ViPER-PE provides both methods.

Each attribute distance is calculated separately in the properties (PF) and the evaluation parameters file (EPF). These files include all the needed configuration for the performance evaluation process. Finally, the command line application *Viper-pe.jar* is invoked like the following:

```
Java -jar Viper-pe.jar –pr properties.pr -epf configuration.epf -g ground_truth_file.xml –r test.xml -o results.out -raw results.raw
```

B.2.2 Performance evaluation process for this work

In this work, the following configuration and properties files has been designed and used (plus additional MATLAB scripts not shown) to manage the whole evaluation process:



- Parameter Configuration File (EPF file)-

```
#BEGIN_OBJECT_EVALUATION
OBJECT PutObject [- -] BoundingBox : [- -]
OBJECT GetObject [- -] BoundingBox : [- -]
OBJECT AbandonedObject [- -] BoundingBox : [- -]
OBJECT StolenObject [- -] BoundingBox : [- -]
#END_OBJECT_EVALUATION

#BEGIN_RESULT_FILTER
OBJECT PutObject DetectionScore : > 0.7
OBJECT GetObject DetectionScore : > 0.7
OBJECT AbandonedObject DetectionScore : > 0.7
OBJECT StolenObject DetectionScore : > 0.7
 #END_RESULT_FILTER
```

- Properties File (PR file)-

```
# # Level of analysis # level = 3

# How to crop the set of possibles at the end.
target_match = SINGLE

# # Range Distance Metric
# dice = Dice coefficient (2 & intersection / sum of candidate and target)
range_metric = dice

# # String Distance Metric #
L = Levenshtein  string_metric = L

# # Level Specific Metrics
# mean, median, minimum, maximum
level3_metric = mean

###########################################################################################
# Default Tolerance Configuration
# Temporal Range - [0 = exact match] #
range_tol = .99

# # Attributes #
bvalue_tol = 1.0
bbox_tol = 0.7

# # Level Specific # infinity_tolearance : Set all tolerances to infinity #
level3_tol = 0.99

########################################################################################
# Presentation Parameters
verbose = true
attrib_width = 40
```

Figure B.4: Performance evaluation configuration files

Appendix C

Color Conversion

In this appendix we describe some linear transformations used in this document. As said in chapter 4, input images are in RGB color space and some analysis perform their tasks in other color spaces. The RGB data was scaled so that values were between 0 and 1. They are the following:

C.1 RGB to YUV

YUV space is the basic format for the composite color television standard for NTSC, PAL and SECAM. In this work, this transformation is used to compute the intensity value. Pixel intensity values are used for segmenting moving foreground objects from background. . These are the equations of the transformation:

$$\begin{aligned} Y &= 0.30R + 0.59G + 0.11B \\ U &= 0.60R - 0.28G + 0.31B \\ V &= 0.21R - 0.52G + 0.31B \end{aligned} \tag{C.1}$$

C.2 RGB to HSV

This transformation is used to detect shadows in the images by thresholding the intensity ratio and chromaticity differences between the background and current images. These are the equations of the transformation:

$$\begin{gathered} V = max(R,G,B) \\ S = \begin{cases} \frac{max(R,G,B)-min(R,G,B)}{max(R,G,B)}, & max(R,G,B) \neq 0 \\ 0 & max(R,G,B) = 0 \end{cases} \\ H = acos\frac{0.5((R-G)+(R-B))}{\sqrt{(R-G)(R-G)+(R-B)(G-B))}} \end{gathered} \tag{C.2}$$

Appendix D

Analysis techniques

D.1 Meanshift

The mean shift algorithm is a powerful technique for clustering. It is used in several areas like segmentation, moving object detection, tracking, feature selection,... This technique is based on its recursive application to find the stationary point more close to density function to be estimated.

The mean shift algorithm presented here was used by Comaniciu et al. in the paper titled "Mean shift a robust approach toward feature space analysis" (2002). The multivariate kernel density estimate is:

$$f(x) = \frac{1}{nh^d}\sum_{i=1}^{n} K(\frac{x - x_i}{h})$$

where x is a point, n is the number of points, $K=$ selected kernel, d = dimension of the space, and h = window radius or bandwidth. The kernel is a Epanechnikov kernel, and its formula is:

$$K_E(x) = \begin{cases} \frac{1}{2}C_d^{-1}(d+2)(1-\|x\|^2), & if\ \|x\| < 1 \\ 0 & otherwise \end{cases}$$

where C_d is the volume of the unit d-dimensional sphere. The colour distribution of the object is converted to 1D distributions. The multivariate density estimate is used to weight colours based on their appearance inside the kernel. A Bhattacharyya coefficient is used to evaluate the difference δ between two distributions:

$$\delta(y) = \sqrt{1 - \sum_{u=1}^{m} p(y)q}$$

where q is the target model, p is the calculated distribution, and y is location. The tracking starts initialization from which the target histogram is set. In the next frame, the surroundings of former localization is sought to find a position in which the difference δ is smaller than a certain threshold.

D.2 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is an orthogonal linear transformation often used to reduce multidimensional data sets to lower dimensions for analysis (based on the mean square error). This technique has been widely used in compression and data analysis (like Pattern Recognition). In high-dimensionality problems, PCA is used to reduce dimensionality as follows:

Given $x \in \mathbb{R}^d$as a column vector, we look for a linear transformation $y = Wx$, with $y \in \mathbb{R}^k$and $k < d$, that produces the minimum error when we approximate x by using y. Supposing that we have n samples of the vector $x(x_j,\ j = 1, 2, 3, ...n)$, the previous minimization condition is equivalent to minimize the following equation:

$$J = \sum_{j=1}^{n} \|x_j - y_j\|$$

Then we apply the PCA algorithm :

Algorithm 4 PCA Algorithm

1. Firstly, the covariance matrix (or dispersion matrix) for x is calculated (S)
2. Diagonalize $S : S = VDV_t$with D a diagonal matrix of eigenvalues and V an orthogonal matrix ($VV_t = I$) with the eigenvectors as columns. Matrix D and V are sorted in descending order of the eigenvalues.
3. Select the first $k$eigenvalues and eigenvectors. If we call V_k the matrix formed with first k columns, the transformation matrix is $W = V_k^t$.

To select an appropriate number of k components, this value should represent a significance percentage of the total variance.

Appendix E

Gamma-based Foreground segmentation

E.1 Background subtraction

The foreground segmentation stage is based on a modification of the basic background subtraction operation (subtracting a square window around every pixel to avoid noisy results):

$$F(I[x,y]) \iff \sum_{i=-W}^{W} \sum_{j=-W}^{W} (|I[x+i,y+j] - B[x+i,y+j]|)^2 > \beta \tag{E.1}$$

Let $X_i, \forall i \in [1,k]$, be k independent, normally distributed random variables with zero mean and variance equal to 1. The random variable Q follows a Chi-square distribution of k degrees of freedom:

$$Q = \sum_{i=1}^{k} X_i \approx \chi_k^2 \tag{E.2}$$

and the probability distribution function of $Q$is:

$$P(Q \leq p) = \frac{\gamma(k/2, q/2)}{\Gamma(k/2)} \tag{E.3}$$

where γ is the "Lower Incomplete Gamma Function" and Γ is the "Gamma Function". Thus, if region around $I[x,y]$ in current frame belongs to background:

$$X_{i,j}(x,y) = \frac{I[x+i,y+j] - B[x+i,y+j]}{\sigma} \approx N(0,1) \tag{E.4}$$

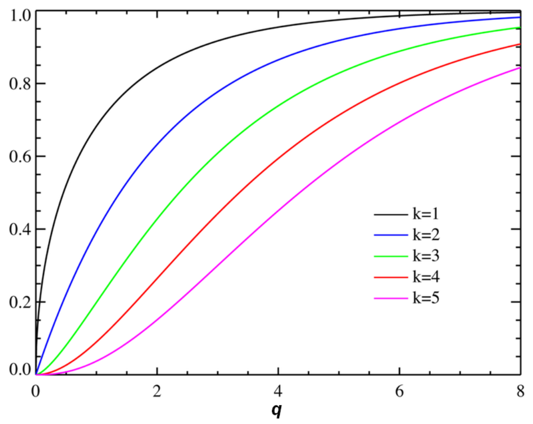


Figure E.1: Chi-square probability distribution function

$$\Rightarrow Q([x,y) = \sum_{i=-W}^{W} \sum_{j=-W}^{W} X_{i,j}^2(x,y) \approx \chi^2_{2W+1} \tag{E.5}$$

$$\Rightarrow P(Q(x,y) \leq p) = \frac{\gamma((2W+1)/2, q/2)}{\Gamma((2W+1)/2)} \tag{E.6}$$

and finally the foreground test is as follows (equals to the significance test described in section):

$$Foreground(I[x,y]) \Longleftrightarrow \frac{\gamma((2W+1)/2, Q(x,y)/2)}{\Gamma((2W+1)/2)} > \beta \tag{E.7}$$

E.2 Gamma-related functions

E.2.1 Gamma function

Gamma function extends the factorial analysis to the complex domain ($\mathbb{C}$). In mathematics, the Gamma function (represented by Γ) is an extension of the factorial function to real and complex numbers. For a complex number z with positive real part, this function is defined by the integral:

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt \tag{E.8}$$

and satisfies the recurrence relation:

$$\Gamma(z+1) = z\Gamma(z) \tag{E.9}$$

When the argument $z$is an integer, the gamma function is just the familiar factorial function, but offset by one:

$$n! = \Gamma(n+1) \tag{E.10}$$

When the argument $z$is a complex number (and $Re(z) > 1$), the gamma function is:

$$\Gamma(z) = \frac{e^{-\gamma z}}{z} \prod_{n=1}^{\infty} (1 + \frac{z}{n})^{-1} e^{z/n}$$

E.2.2 Gamma Incomplete functions

The incomplete gamma function are defined as an integral function of the same Gamma integrand. There are two varieties of the incomplete gamma function: the upper incomplete gamma function is for the case that the lower limit of integration is variable (i.e. where the "upper" limit is fixed), and the lower incomplete gamma function can vary the upper limit of integration.

The upper incomplete gamma function is defined as:

$$\Gamma(a,x) = \int_x^{\infty} t^{a-1} e^{-t} dt \tag{E.11}$$

The lower incomplete gamma function is defined as:

$$\gamma(a,x) = \int_0^{x} t^{a-1} e^{-t} dt \tag{E.12}$$