Towards an Evaluation and Repair Framework for Video Interpretation

Benoît Georis

September 2, 2003

Table of Contents

Table of Contents 1

Acknowledgements 3

1 Introduction 4

2 Related Work 7
2.1 Program Supervision Techniques . . . 7
2.2 Evaluation Techniques . . . 10

3 Video Interpretation and Tracking 12
3.1 Video Interpretation . . . 12
3.2 Mobile Object Detection . . . 14
3.3 Tracking . . . 15

4 Evaluation Framework 17
4.1 Video Sequence Characterization . . . 17
4.1.1 Sensor Type . . . 18
4.1.2 Camera and Scene Motion . . . 18
4.1.3 Illumination . . . 19
4.1.4 Cluttered Scene . . . 19
4.1.5 Video Sequence Selection . . . 20
4.2 Ground Truth Generation . . . 21
4.2.1 Mobile Object Detection . . . 21
4.2.2 Tracking . . . 24
4.3 Result Criteria . . . 24
4.4 Evaluation Algorithms . . . 26

5 Global Evaluation **28**
5.1 Mobile Object Detection Evaluation 29
5.1.1 Algorithm 29
5.1.2 Results and Analysis using global criteria 30
5.1.3 Results and Analysis using technical criteria 31
5.1.4 Results and Analysis using detection criteria 32
5.2 Tracking Evaluation 34
5.2.1 Algorithm 34
5.2.2 Results and Analysis using global criteria 35
5.2.3 Results and Analysis using technical criteria 36
5.2.4 Results and Analysis using detection criteria 37
5.2.5 Results and Analysis using tracking criteria 38

6 Fine Evaluation **40**
6.1 F2F Tracker Algorithm 41
6.1.1 Computation of Clusters of Moving Regions 41
6.1.2 Discrimination Between Concurrent Hypotheses 42
6.2 Fine Evaluation Algorithm 45
6.3 Repair 46

7 Conclusion **48**

8 An Evaluation Framework for Video Interpretation **50**
8.1 Automatization 50
8.2 Generalization 51

Bibliography **53**

Acknowledgements

I would like to thank Monique Thonnat, my French supervisor, for her time and her ideas during this year. I am also thankful to Benoît Macq, my Belgian supervisor, for his guidance.

Many thanks to François Brémond who shared with me his knowledge of tracking and with whom I had a lot of enriching discussions. He gave me constant and friendly encouragement.

I would like to thank also Sabine Moisan who opened my eyes on the beautiful world of program supervision and knowledge representation. She took time to answer to my numerous questions.

Of course, I am grateful to my parents for their patience and *love*. Without them this work would never have been finished.

I wish to thank all my friends for their support and all the good time we had together.

Finally, thanks to the Walloon Region for funding my studentship under the FIRST EUROPE program and to Jean-François Delaigle who initiated this program.

Chapter 1

Introduction

At each second, our brain receives thousands of information it has to process from various sensors like eyes or ears. With our eyes, we are able to understand and to represent ourselves the world around us, for example to distinguish different people or to know if it is a windy day by looking at the bushes, and so on. The mechanisms involved in such tasks are very complex. Some attempts to model the visual system, which are called psychovisual studies, can be found in [1]. Nevertheless, the way human brain analyzes and interprets perception data are not yet well understood.

Our motivation here is not to model human brain but to find and to model which knowledge human beings use to interpret visual scenes. It originates from the idea that a system composed of cameras, processing resources and the appropriate knowledge about the task to be performed could be developed to reach the ambitious goal of automatic video interpretation. This automatic interpretation is dedicated to human behaviour recognition from video sequences.

In the past few years, many interpretation systems have been developed but none of them have been successfully applied to real world applications. One major weakness of these systems is the tracking process. Tracking is still a central issue in scene interpretation, as the loss of a tracked object prevents the analysis of its behaviour. Tracking has been extensively studied for many years. Various techniques have been explored, both model-based [2], [3] and model-free [4]. Nevertheless, the tracking problem remains unsolved since there are many sources of ambiguities like shadows, illumination changes, over-segmentation and mis-detection. These difficulties need to be handled in order to make the correct matching decision.

In addition, the increasing number of installed surveillance systems need reasoning capabilities requiring highly efficient tracking algorithms. These systems are running 24 hours a day in varying conditions. Our goal is to conceive a generic human tracking algorithm which can adapt itself automatically to a scene change.

To face this problem, we propose in this work a general framework for algorithm evaluation, applied to the tracking problem. As a first step, we investigate supervised evaluation since we compare algorithm outputs with ground truth. Our final objective is to have a fully automatic evaluation to adapt dynamically the interpretation platform to any new situations. Algorithm assessment is mandatory - but not sufficient - to design robust systems. We first propose a global evaluation method for ranking tracking algorithms. Evaluation criteria have been defined accordingly. Second, we propose a fine evaluation method which classifies and diagnoses tracking errors. This step is algorithm dependant and try to locate exactly the code or the parameter which is responsible for an error class. This second evaluation level and the associate repair methodology is the main contribution of this work.

This report is laid out as follows. Chapter 2 browses existing evaluation processes and briefly describes program supervision techniques which guide our work. Our video interpretation platform is explained and illustrated in chapter 3. Chapter 4 describes the evaluation process, preceded by a video input characterization, a description of the ground truth generation and a definition of the results criteria. Chapter 5 details the global evaluation algorithm and discusses the results obtained, both for mobile object detection and tracking. Chapter 6 explains how we refined this global evaluation algorithm in the tracking case. We conclude this chapter by describing the repair methodology using this fine evaluation. Chapter 7 draws the conclusion of this work. Finally, chapter 8 indicates further work which is composed of two main research axes, the first one being the automatization of evaluation and repair and the second one being the generalization of the evaluation.

Chapter 2

Related Work

In this chapter, we introduce useful program supervision concepts in section 2.1 and we give a brief overview on existing evaluation techniques in section 2.2.

2.1 Program Supervision Techniques

Today, many disciplines like scientific computing or image processing require complex software libraries to solve real life problems. Most of the time, these libraries are written by domain specialists but they are used by non specialists. Thus, many difficulties arise when users have to choose a program to solve a specific task, to combine several programs or to tune them. In addition, code documentation is not sufficient and does not contain all the necessary knowledge about the use of a program.

In recent years, program supervision techniques have appeared to be an adequate answer to these difficulties. For examples, program supervision techniques have been successfully applied to galaxy classification [10] or SAR image analysis [11].

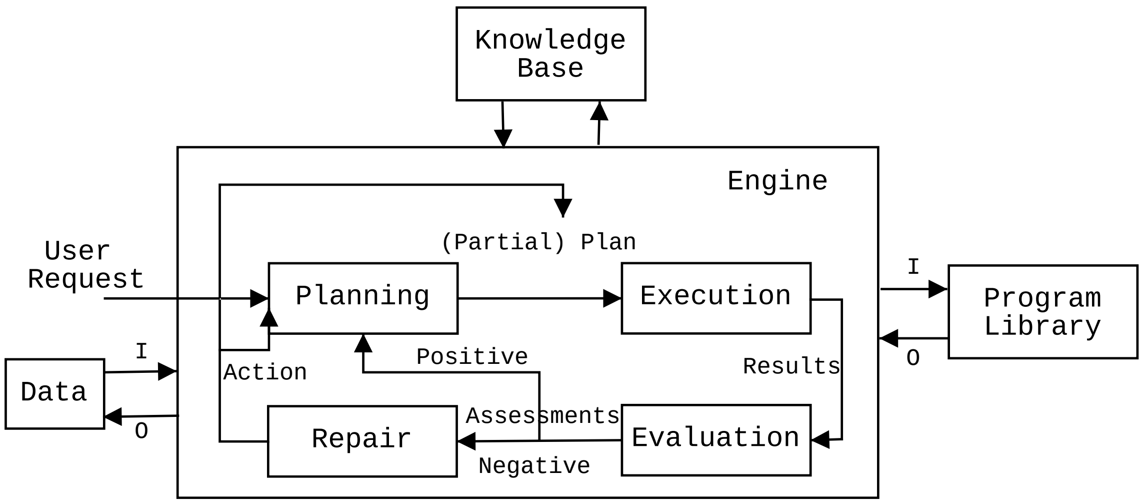


Figure 2.1: General program supervision system architecture

Program supervision consists in the automation of planning and control of execution of programs from an existing library [12]. A general program supervision system architecture is presented in figure 2.1. The system is composed of a program library, an engine and a knowledge base. It uses artificial intelligence techniques, and more precisely knowledge based techniques, to represent explicitly the knowledge about programs and their use. Advantages of doing so are numerous, compared to a knowledge integration into the code (i.e., a smart algorithm). For example, a formalism (e.g., production rules) is provided, various execution modes (ranging from interactive to fully automatic) can be investigated, reuse of code becomes possible, and so on.

The reasoning of a program supervision system, the engine, consists of 4 different phases, that can be completely or only partly automated [13]:

1. planning: the selection of a set of programs that can solve the user's request, and the actual scheduling of the programs.

2. execution: the initial setting of program parameters and the execution of programs.
3. evaluation: the evaluation of the results of program execution to ensure that the programs and/or their tuning were appropriate.
4. repair: the correction of the scheduled programs, or the adjustment of program parameters, if the results are deemed unacceptable.

According to [14] and [15], the knowledge base contains operators which are representations of programs (with descriptions of their data and parameters) and of typical combinations of programs, as well as criteria to guide the reasoning process. There are two types of operators: primitive and complex ones. A primitive operator represents a particular program (of the library) and a complex operator represents a particular combination of programs.

There are four types of criterion:

- initialization criterion: it contains information to assign value to parameters.
- choice criterion: it is used to choose, among all available operators, the most pertinent operators.
- evaluation criterion: it assesses the results obtained after operator execution.
- repair criterion: it says how to modify a parameter value in case of bad evaluation.

Once a user emits a request, the engine selects and schedules operators according to the information (criteria) contained in the knowledge base, then executes the plan, evaluates the results and finally proposes a repair step, if necessary.

Most of the time, the main difficulty arises from the knowledge base modelling by a domain expert. But, as the tracking problem is still an open problem, we must first focus our researches on knowledge acquisition. This work is a first step towards this objective. We provide a classification of tracking errors, then we show how to diagnose these errors and finally we show potential repair processes. Once this step will be done, knowledge base modelling will be easier. We will be able to list operators and their associate criteria.

Moreover, current work done by the ORION team [16] will help either the design of a new knowledge based program supervision system or the reuse and adaptation of an existing one. They propose a mathematical model and a formal language to describe the knowledge about behaviour. With such tools, the behavioural correctness of a modified existing system can be assessed.

2.2 Evaluation Techniques

For three years now, the scientific community has created a workshop dedicated to evaluation of tracking, named PETS (Performance Evaluation of Tracking and Surveillance) [5]. This enforces the idea that we need evaluation techniques to assess the reliability of existing tracking algorithms. But this workshop is mostly intended to test various algorithms on the same video inputs. Algorithm comparison is only qualitative and quantitative comparison based on precise criteria is not available.

Nevertheless, we can mention an interesting theoretical work on performance evaluation which can be found in PETS 2002 [6]. It first discusses the importance to test algorithms on real video sequences, for instance with various weather conditions for outdoor sequences. Second, it presents pros and cons of ground truth techniques and their alternatives.

There are also some low-level algorithms comparison, for example [7] and [8], but less on tracking algorithms. The ODVIS platform [9] has been proposed as an open framework for researchers who want to test their tracking algorithm on various video sequences. Graphical facilities and error metrics computation ease results visualization. Nevertheless, none of these techniques propose to use evaluation results to improve tracking processes. They only make a global evaluation and their use is limited to the ranking of algorithms.

In conclusion, the lack of useful evaluation techniques to improve tracking algorithms has motivated our work.

Chapter 3

Video Interpretation and Tracking

Our final objective is to design a video interpretation platform dynamically adaptable to any new situations. For this reason, we give in this chapter an overview of video interpretation (section 3.1) followed by a more careful description of two sensitive components, mobile object detection (section 3.2) and tracking (section 3.3). These two components are the central issue of the evaluation.

3.1 Video Interpretation

The goal of video interpretation is to give a semantic description of a scene depicted by video streams. As in most of the cases, video streams come from fixed monocular cameras. We limit our study to this configuration. However, some systems work with mobile cameras or other type of sensors (e.g., stereo cameras). A video interpretation process consists of five main tasks:

1. Mobile Object Detection

2. Frame to Frame Tracking (F2F Tracking from now)

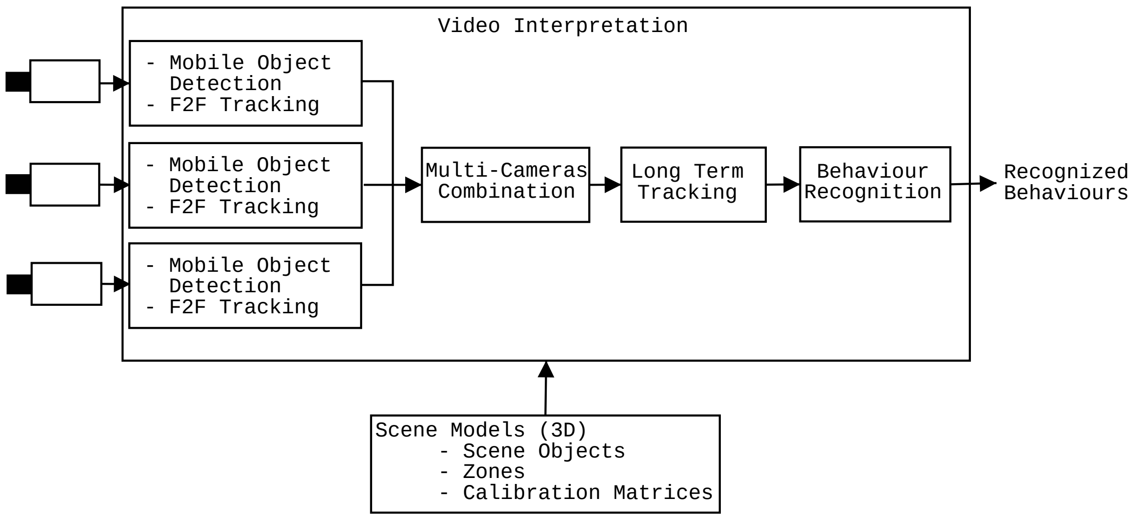


Figure 3.1: Video interpretation process overview

3. Multi-Cameras Combination

4. Long Term Tracking

5. Behaviour Recognition

This is illustrated in figure 3.1. First a mobile object detection and a F2F tracking processes generate a graph of mobile objects for each camera. Second, a combination mechanism is performed to combine the graphs computed for each camera into a global one. Then, this global graph is used for long term tracking of actors evolving in the scene. Finally, behaviour recognition is performed for each tracked actor. So, the output is a semantic description of the recognized behaviours. In addition, 3D scene models can be used as a priori contextual knowledge of the observed scene. For instance, a scene model can contain 3D positions and dimensions of static scene objects (e.g., a bench, a ticket vending machine) and zones of interest (e.g., an entrance zone). Semantic attributes (e.g. fragile) can be associated to objects or zones of interest to be used in the behaviour recognition process.

For us, the main idea is to help the whole processing chain by adding as much knowledge as we can at each level of reasoning. For instance, mobile object detection is driven with a human model represented by the mean width and height of a person. Contextual knowledge of the scene environment provides to classification the necessary information to label a person as occluded by a static object. A detailed description of our video interpretation platform can be found in [17].

Of course, video interpretation can be done with a camera alone. In such a case, no fusion operation is performed. This will be our testing conditions. The next two sections describe in more details the processing done for each camera before fusion.

3.2 Mobile Object Detection

The goal of this step is to detect for each frame moving regions corresponding to mobile objects in the scene and to classify them with a label corresponding to their type, such as PERSON. This task can be divided into five sub-tasks: moving region detection, feature extraction, moving region classification, merge and split of moving regions and reference image adaptation.

Moving region detection is usually computed as a difference between a current image and a reference image. Because the camera is fixed, the reference image is a still image representing the scene without mobile objects (also called background image). Then, a thresholding of this difference image is realized and a filtering and connected component analysis create all the moving regions (represented by their bounding boxes) that correspond to mobile objects. Other detection techniques can be applied such as frame differencing or optical flow.

Feature extraction consists in parameter computation for each moving region: centre of gravity, position, height and width all four defined in 2D (in the image) and/or in 3D (in the scene), color and texture signature, posture,...

Classification labels moving regions according to a semantic class: PERSON, GROUP_OF_PEOPLE, SCENE_OBJECT, NOISE, VEHICLE, UNKNOWN,... A scene object is for instance a door or a chair.

Then, moving regions can be merged or splitted to improve the matching with a class. For instance, two moving regions corresponding to the head and the body of a person can be merged to form a new moving region belonging to the class PERSON.

Finally, the reference image is updated to take into account illumination changes. This task can be simply a temporal filtering on several images or can be smarter and be based on information coming from higher level tasks (e.g., tracking or behaviour recognition).

3.3 Tracking

In general, tracking is aimed at linking moving regions in consecutive frames. In most of the cases, the linking process is done on successive frames and is called F2F tracking. A second and important tracking step can further be added to disambiguate situations which require a frame history. This is called a long term tracking process. We focus here on the F2F tracking only.

F2F tracking takes as input at each time t a list of moving regions from the mobile object detector, and its goal is to produce a temporal graph $\Gamma(t) = (X, U)$, where X is a set of nodes containing the detected moving regions and U is a set of links between two moving regions at successive times. More precisely, given a set of moving regions $N = \{n_1 ... n_p\}$ in the new frame and a set of moving regions $O = \{o_1 ... o_q\}$ in the old frame, the objective is to create m quantified links between old and new moving regions. Sequences of links (paths) in this graph represent the various possible trajectories a moving region may have. Moreover, a value can be associated to each link to quantify the confidence about the correspondence between the moving regions connected by the link. The goal of the F2F tracker is to not miss any correspondence and to reduce the ambiguities.

The F2F tracker algorithm is usually guided by two criteria: the first one is the correspondence between detected moving regions and a model, and the second one is the temporal continuity indicating that moving regions detected at time t are expected to be detected at time $t + 1$.

Chapter 4

Evaluation Framework

This chapter presents the general framework for algorithm evaluation. It is presented here for tracking algorithms but extension to other types of algorithms is discussed in chapter 4. Section 4.1 emphasizes on the importance of choosing the algorithm inputs according to several difficulties in order to make a significant evaluation. A discussion on the ground truth definition and the need to be the most objective and algorithm independent is done in section 4.2. Next, section 4.3 enumerates the criteria which will be used to evaluate the algorithm results. Finally, section 4.4 describes the overall evaluation method composed of two evaluation levels. Each level will be described in more details in the following chapters.

4.1 Video Sequence Characterization

Tracking algorithms are developed following precise hypotheses which describe the type of video sequences algorithms can process. Thus, tracking evaluation strongly depends on the input sequence type. Moreover, all tracking algorithms succeed when video sequences are simple and fail when they are difficult. We describe hereunder the main difficulties we can encounter and we conclude this section with a description of the video sequence selection and the associate difficulties we have chosen to focus on.

4.1.1 Sensor Type

The sensor type (high resolution, PTZ, infrared, aerial, omnidirectional, stereo, ...) and acquisition/storage materials are of prime importance because it dictates the processing which can be applied on the corresponding video sequences. For instance, a background subtraction technique to detect motion can not be directly applied with a moving camera. We can also have a large field of view camera which introduces distortion or we can have an omnidirectional sensor which requires dedicated techniques.

We can distinguish a large view (e.g., a human is 2 pixels high) from a close-up view (e.g., the human height is 200 pixels high), which are the two extreme range of the sensor. We can also underline the effect of the camera position. With a top view, the perspective effect is less important but it is difficult to compute the height of a person. With a side view, the perspective makes difficult the estimation of the 3D parameters.

4.1.2 Camera and Scene Motion

Although the system uses fixed cameras, motion perturbations can be induced by two kinds of motion. First, camera vibrations which cause mainly translation between consecutive frames. For example, a camera on a pillar for highway surveillance can be oscillating due to wind. Second, scene motion which can be generated by any object in the scene like an escalator or relative motion which occurs when the camera is fixed inside a moving object like a car or a train.

4.1.3 Illumination

We can list four main problems caused by illumination changes:

- Slow illumination changes: in outdoor scenes, slow illumination changes can occur due to clouds or solar rotation. For indoor scenes, it can be caused by intensity variations from the light source (bulb,...). These changes add noise in the motion detection procedure.
- Fast illumination changes: some image parts are sometimes submitted to fast illumination changes. In an indoor scene for instance, closing blinds can introduce a dark area in the image.
- Automatic gain control of cameras: when the camera is exposed to a strong illuminance change (for instance, an object passing in front of the camera), his gain is automatically corrected to insure the best image quality. This leads to detect motion in the whole image and prevents to find the objects.
- Visual artifacts: these are perturbation phenomenon like reflection against walls, low target contrast, shadows,...

4.1.4 Cluttered Scene

Depending on whether the camera is intended to survey a big parking lot or a small office, different problems will arise. In a small office for instance, people tend to get close to each other or will be more often occluded by a scene object like a desk or a chair. In such a case, people will probably not be correctly detected and tracked. Motion detection and tracking are thus considerably affected by clutter.

4.1.5 Video Sequence Selection

Ideally, we should choose video sequences which enable to study each of these difficulties. An exhaustive work would make vary, inside a video sequence set, one difficulty from the most simple case to the most difficult case, all other difficulties being kept constant and average. But this is a complex task. Actually, we have chosen our test sequences according to three difficulties:

- The average number of persons in the scene. This value can range from 1 or 2 for the simplest sequences to more than 10 in very difficult sequences. We want the tracker to be robust until the scene is overcrowded.
- The detection quality. A good tracker must be able to handle the diverse problems encountered by the detection algorithm like shadows, reflections, low target contrast, etc.
- The number of crossings between persons. It is mandatory to have frequent and long crossings in the video in order to assess the reliability of the tracker since single person tracking does not represent a great challenge. The duration of the crossings depends also on the camera orientation.

Currently, we have selected 15 indoor scene video sequences from three different applications: a bank agency, a metro platform and an office. These sequences are taken from a fixed calibrated monocular camera. Results will be presented for 3 video sequences taken from the metro platform:

- V1, which is 200 frames long. The detection quality is good, there are few noise and good target contrast. The scene is composed of 2-3 persons and few crossings.
- V2, which is 580 frames long. The detection quality is average. The scene is composed of 4-6 persons and contains several crossings.
- V3, which is 500 frames long. The detection quality is bad, there are many lighting variations and some shadows. The average number of persons in the scene ranges from 7 to 11 and they are frequently crossing each other.

4.2 Ground Truth Generation

Tracking algorithms are designed in function of specific results to achieve related to the target application. Some algorithms want to correct detection results and process a whole person (recover his/her feet) even if this person is partially observable due to an occlusion (his/her feet are behind a desk). Other algorithms do not need to recompute accurately the detection of the person. Therefore, the issue of defining ground truth consists in being impartial for all tracking algorithms. In our case, we have acquired ground truth using a software interface called ViPER [18]. We have defined two different types of ground truth: one for mobile object detection and one for tracking. We discuss these two types of ground truth in the next two sections.

4.2.1 Mobile Object Detection

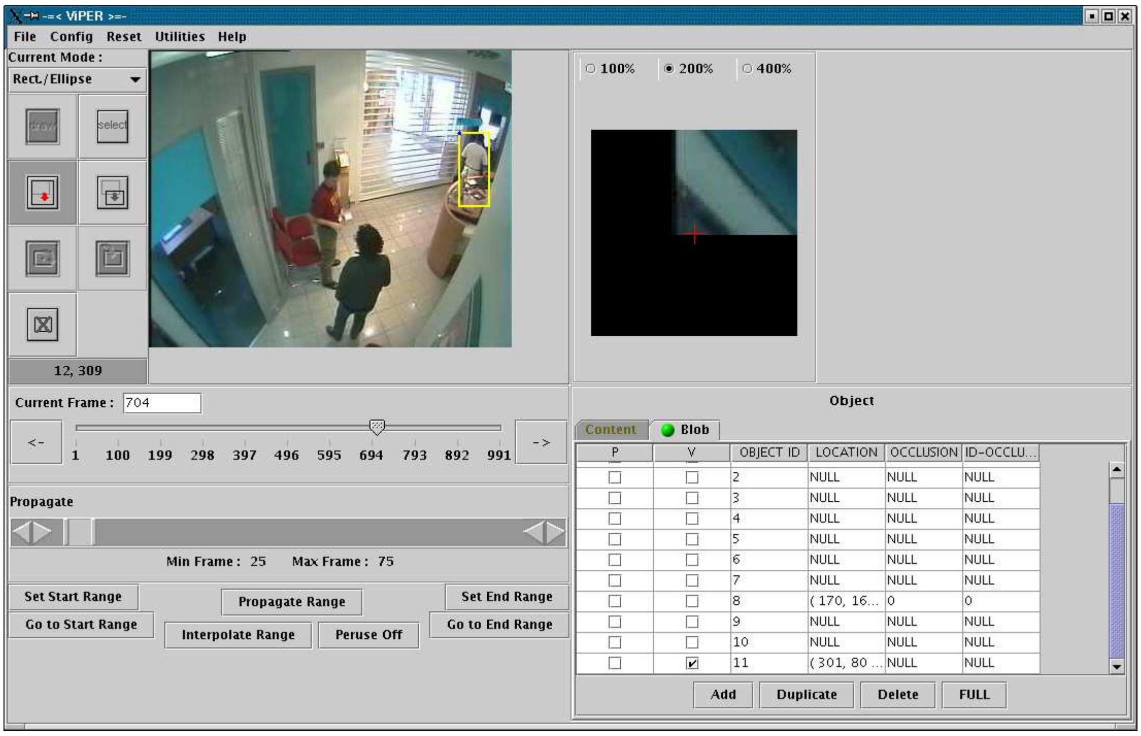
The ground truth definition is subjective. There are several key questions a user has to answer before he/she can define ground truth. Do we draw the bounding box of the whole person when the person is occluded? Do we draw two bounding boxes when two people walk very close to one another or do we draw only one for the group? These choices have to be made with respect to the target application and the features to evaluate. In counterpart, care must be taken to avoid introducing a bias when defining ground truth if, for example, the assessor is aware of typical tracking errors.

We have chosen to draw a bounding box as soon as we see an evidence that a person is present in the scene (a hand, a head or even a shadow). In other words, we draw the full bounding box for each person even when he/she is dynamically or statically occluded. The only exception to this rule appears when the person is on the camera border. In such a case, we do not draw the bounding box outside the image.

An example of a bounding box definition using the ViPER graphical user interface is illustrated in figure 4.1. The white person is statically occluded by the desk. Nevertheless, we can see that the full bounding box has been drawn, feet position being guessed.

These choices (to be the most precise as possible) have been made in order to be able to rank the most sophisticated algorithms. For simpler algorithms (that do not address strong requirements) it is always possible to compensate the detection errors by recomputing the observable parts of the mobile objects.

Due to these choices, we are able to determine whether the motion detection and classification process have correctly labelled the person as OCCLUDED. Imprecisions can appear since the coordinates of invisible parts are guessed. The evaluation process has to take into account this imprecision source.



Current Mode : Rect./Ellipse

12, 309

Current Frame : 704

Min Frame : 25 Max Frame : 75

○ 100% [✓ ● 200%] ○ 400%

P	V	OBJECT ID	LOCATION	OCCLUSION	ID-OCCLU...
☐	☐	2	NULL	NULL	NULL
☐	☐	3	NULL	NULL	NULL
☐	☐	4	NULL	NULL	NULL
☐	☐	5	NULL	NULL	NULL
☐	☐	6	NULL	NULL	NULL
☐	☐	7	NULL	NULL	NULL
☐	☐	8	(170, 16...	0	0
☐	☐	9	NULL	NULL	NULL
☐	☐	10	NULL	NULL	NULL
☐	✓	11	(301, 80 ...	NULL	NULL

Figure 4.1: Bounding box definition using ViPER graphical user interface

We store the following attributes called **ground truth attributes**: the 2D width and height, the 2D position and the type (PERSON, VEHICLE, ...). These attributes computed in 3D using camera calibration can also be used as ground truth. It makes a total of 7 ground truth attributes for mobile object detection.

4.2.2 Tracking

Once we have defined ground truth for motion detection, we are able to define the ground truth for tracking. It consists in giving the same identifier to a person throughout the whole sequence. As a result, we only store the identifier as attribute. So the total number of ground truth attributes for mobile object detection and tracking is 8.

4.3 Result Criteria

Thanks to these two ground truths, we are now able to evaluate mobile object detection and tracking algorithms. To present evaluation results to the user (function of his/her goal), we have defined five evaluation criteria types, each one having a specific purpose and ranging from the most global (1) to the most specific (5) criteria:

1. Global criteria: these criteria correspond to end user requirements. For instance, they generally want to know the number of persons detected and correctly tracked. It is also desirable to have a weak false alarm rate so that the user can trust the system.

2. Technical criteria: these criteria give us the percentage of ground truth attributes correctly computed for the objects. For instance, they enable to compute the percentage of mobiles which have a 3D position less than 1 meter away from the ground truth 3D position.

3. Detection condition criteria: these criteria allow us to balance the results according to a given difficulty characterizing the video, for instance the scene distance between the object and the camera. During the repair process, these criteria are useful to tune the level of repair. For example, in a first stage of repair we can be more tolerant to an error (loss of a track) if the object is far from the camera.

4. Tracking criteria: these criteria enable to detect and classify difficult cases of tracking (caused by mobile object detection) leading to a tracking error. Most trackers try to repair detection problems thanks to temporal coherency. For example, a good tracker should merge a head with the corresponding body of a person in an over-segmentation case. This classification will be used also in the repair process.

5. Repair criteria: these are the most specific criteria. Their goal is to classify tracking errors according to the tracking sub steps, or code parts, which generated them in order to repair these sub steps.

For criteria 1 to 3, there is another way to study the interaction/the relation between the two processes, mobile object detection and tracking. We can quantify the degradation of tracking results induced by bad detection. For that, we simply have to compare tracking outputs when we feed the tracker with the output of the

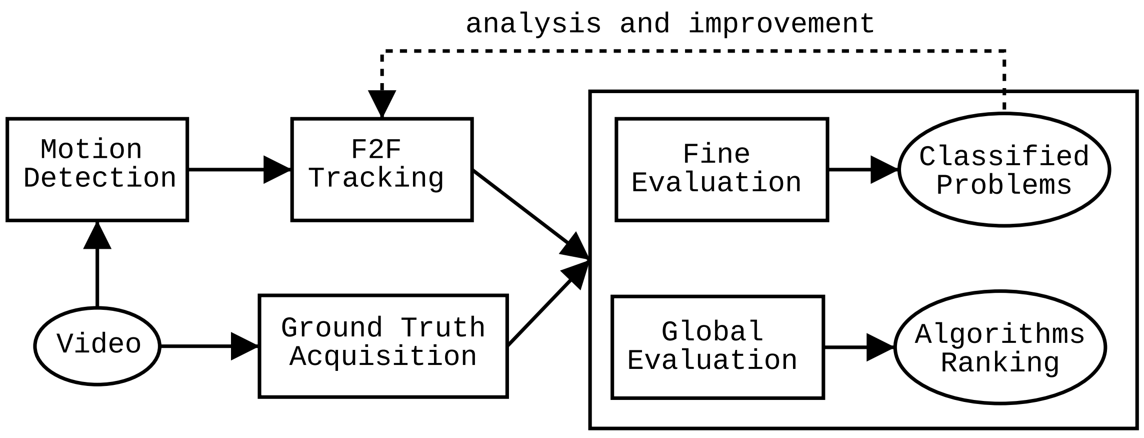


Figure 4.2: Off-line feedback loop to improve F2F tracking

motion detection or with detection ground truth. It is also a way to check how good the tracker is at solving detection problems.

4.4 Evaluation Algorithms

Once algorithm inputs and outputs are well defined and the necessary evaluation material are established, we can investigate the evaluation algorithm. This is illustrated in figure 4.2.

Evaluation can be done on two levels. The first one is a global evaluation process which is general and applied to both mobile object detection and tracking algorithms. At this level, these algorithms can be seen as a black box. The goal is to rank all types of algorithms using mostly global and technical criteria. To get a more precise global evaluation, detection and even tracking criteria can also be used. This is the topic of chapter 5. Chapter 6 will present the second level which is a fine evaluation based on the analysis of tracking sub steps and using mostly repair criteria. Currently, this second level is only applied to the F2F tracking algorithm. This is an iterative procedure. A classification of tracking errors is defined and the associated errors are identified automatically. Once this identification is done, the goal is to diagnose which parameter or which code of the algorithm generated a given error class. The last step consists in repairing the algorithm. Finally, the new tracking algorithm is re-evaluated.

Chapter 5

Global Evaluation

The goal of this global evaluation is to rank tracking processes by giving a global analysis of their performances. Evaluation results are produced with respect to global, technical, detection and tracking criteria by comparing tracking outputs and ground truth. The proposed method can be applied to most trackers since the tracking process is seen as a black box. It only requires tracking outputs with 2D information to produce the evaluation results. Its use is rather simple since both inputs are required to be coded in standard XML. This is a widespread and easy-to-use format.

The evaluation algorithm consists of two parts: mobile object detection evaluation and F2F tracking evaluation. The algorithm classifies detections made by the mobile object detection process and links made by the tracking process into three main categories: true positives (TP), false positives (FP) and false negatives (FN). True negatives (TN) are of no interest here. Each main category has several sub-categories defined by the technical criteria in order to evaluate object attributes. In order to facilitate the diagnosis, all results are further categorized according to the detection criteria. The type of occlusion (static or dynamic) is the first criterion. When there is no occlusion, results are broken down according to the camera distance (close, medium, far). Static occlusion happens when people, represented by their bounding boxes, are occluded by the static inventory of the scene (e.g., walls, furniture, etc), while dynamic occlusion occurs when people overlap. Finally, for tracking evaluation only, tracking criteria generate tracking errors due to mobile object detection. The two types of evaluation are described separately in the following.

5.1 Mobile Object Detection Evaluation

5.1.1 Algorithm

The classification into true positives, false positives or false negatives depends solely on the degree of overlap between the ground truth objects and the bounding boxes made by the mobile object detection process. A false negative is a ground truth object not sufficiently covered by a detected bounding box. A false positive is a detected bounding box not covered or not sufficiently covered by any ground truth object. False positives are registered in the same way as false negatives.

Detections which are not false negatives or false positives are true positives, i.e. bounding boxes sufficiently overlapping a ground truth object. For a true positive detection we register whether the 2D form of the bounding box agrees with the corresponding ground truth object, and also whether the 3D centres of gravity conform well. In addition we register to what degree the mobile object detection process chooses the right class (PERSON or GROUP_OF_PEOPLE) for the box. Finally we register whether the system correctly detects the presence of static occlusion (OCCLUDED_PERSON). These computation are done with the help of technical criteria.

	Mobile Object Detection		
	true positive rate	false negative rate	false positive rate
video V1	98.5%	1.5%	7%
video V2	98.8%	1.2%	1.8%
video V3	94%	6%	8%

Table 5.1: Mobile object detection: true positive rate, false negative rate and false positive rate for video sequences V1, V2 and V3

5.1.2 Results and Analysis using global criteria

We have computed two metrics. The first metric is the average percentage of the time people are detected. It is defined as the total number of true positives divided by the total number of expected detections (ground truth):

- For video sequence V1, the first person was detected 100% of the time. The second person which is dynamically occluded is detected 55% of the time.
- For video sequence V2, all persons were detected 85.6% of the time, on average. This is due to the high frequency of dynamic occlusion.
- For video sequence V3, all persons were detected 92% of the time, on average. This sequence emphasizes more on crossings

The second metric is the true positive rate (TP/(TP+FN)) for true positives, the false negative rate (FN/(TP+FN)) for false negatives and the false positive rate (FP/(FP+TP)) for false positives. This is illustrated in figure 5.1.

These two metrics can help us to compare various algorithms and can be indicative for an end user. But they are not sufficient to know if the process is reliable.

TP	2D form	3D position	class label
video V1	97%	45%	31.6%
video V2	93.6%	33%	50.7%
video V3	96.2%	35%	51.8%

Table 5.2: Mobile object detection: percentage of correctly computed attributes for true positives (TP) for video sequences V1, V2 and V3

5.1.3 Results and Analysis using technical criteria

We illustrate evaluation results using technical criteria in table 5.2 for the three video sequences. It represents, for true positives, percentage of the ground truth attributes correctly computed for objects. Columns show respectively the matching (with ground truth) of the 2d form of the bounding box, the 3D position of centres of gravity and the class label.

The purpose of mobile object detection evaluation is to verify that the chosen video sequences are sufficiently varied, and that they pose problems of differing nature both for mobile object detection process and tracking process (e.g., the class is wrong under occlusion). These evaluation results also serve to provide an impression of the overall performance of the mobile object detection procedure, and thereby a notion of the difficulties faced by the subsequent tracking.

In a true positive case, the 2D form of the bounding box is correctly computed most of the time (e.g., 97% of the time for video V1). In counterpart, we can see here that the class label is false half of the time for V2 and V3 and even worse for V1. It can be explained in part by frequent dynamic occlusions. For instance, one person in video V1 is most of the time dynamically occluded by a second person. In such a case, the mobile object detection procedure labels the only detected bounding box (for the two people) as PERSON whereas it should have labelled it GROUP_OF_PEOPLE. Nevertheless, this evaluation shows the inefficiency of the mobile object detection procedure to find people. For the 3D position, a careful analysis has shown that the criterion was too strict. In fact, the required precision was a 3D distance between expected (ground truth) position and computed (detected) position less than one meter.

This evaluation using technical criteria is more useful than the evaluation done with global criteria. It has highlighted weaknesses of the mobile object detection procedure.

5.1.4 Results and Analysis using detection criteria

We illustrate evaluation results using detection criteria in table 5.3 for the three video sequences. Rows correspond to classification by occlusion type or camera distance. Columns correspond to attributes computed for the objects, for each video V1, V2 and V3. In video V1 for example, for all true positives computed in a dynamic occlusion case, half a percent of them have a wrong 2D form for their bounding box and 47% of them have a wrong class label. The 3D position has not been represented since the criterion must first be changed.

By looking at the results, some general considerations can be done. In a dynamic occlusion case, the mobile object detection process has enormous difficulties to assign a correct class label attribute. This is a cue for improving the process (e.g., to add specific knowledge about dynamic occlusions in the 3D model). The 2D form of the bounding box attribute is quite good in a close up view but has still to be improved in a static occlusion case.

	video V1		video V2		video V3	
TP	2D form	class label	2D form	class label	2D form	class label
static occl.	0%	8.2%	3.3%	22.9%	1.4%	9%
dyn. occl.	0.5%	47%	0.7%	18.9%	0.6%	13.9%
close	0%	5%	0%	0%	0.2%	2.8%
medium	2.5%	8.2%	2.4%	7.5%	1.6%	22.5%
far	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.

Table 5.3: Mobile object detection: percentage of attributes non correctly computed for true positives with respect to detection criteria for video sequences V1, V2 and V3

	video V1		video V2		video V3	
	FN	FP	FN	FP	FN	FP
static occl.	10.3%	0%	6.9%	0%	65.2%	0%
dyn. occl.	89.7%	0%	89.7%	0%	20%	0%
close	0%	0%	0%	0%	0%	6.2%
medium	0%	0%	3.4%	72.7%	14.8%	93.8%
far	n.a.	100%	n.a.	27.3%	n.a.	0%

Table 5.4: Mobile object detection: false negatives (FN) and false positives (FP) percentage with respect to detection criteria for video sequences V1, V2 and V3

We also illustrate the repartition of false negatives and false positives with respect to detection criteria in table 5.4 for the three video sequences. We have to mention that figures are more significant for V2 and V3 than V1, as there are more observations. Obviously, performances in the medium field of view have to be improved drastically.

In conclusion, this evaluation is more precise than the preceding one (now, we can differentiate weaknesses coming from a close up view or from a medium view) but it still not sufficient to repair the algorithm. Nevertheless, it will be useful when repairing. For instance, we could first improve class labelling in the static occlusion case and then, in the dynamic occlusion case which is more complicated. Also, we could first try to eliminate false negatives in the medium field of view, before dealing with false negatives under static occlusion.

5.2 Tracking Evaluation

5.2.1 Algorithm

For F2F tracking, classification into positives or negatives depends both on the degree of overlap between ground truth objects and bounding boxes made by the tracking process, and also naturally on the presence of links between the boxes.

A true positive link is a link created by the tracking process combining two bounding boxes that both sufficiently cover a ground truth object at times t and $t+1$. For a true positive link, we register to what degree the two boxes represent a good detection of the underlying ground truth object. Imprecise detection is essentially a mobile object detection problem rather than a tracking problem. Though, it is studied in order to assess the frequency of occurrence in the links built by the tracking process. We also register whether the link made is the tracking process first choice (i.e. highest valued) of all the links associated with the two bounding boxes. A second link occurs when the tracking process computes several links and its second choice corresponds to the ground truth link. True positive link evaluation is not very useful in terms of identifying tracking errors, but gives an interesting overall view of tracking's confidence in its choice of links.

All links made by the tracking process which are not true positives are classified as false positives. For a false positive link between two bounding boxes, we register whether the boxes correspond to people or noise. This gives three sub-categories of false positive links: person-person, person-noise and noise-noise (where the objects are different).

A false negative link is a link missed by the tracking process. This is due to either partially detected bounding boxes (at time t or $t+1$ or both) or a missing link between correctly detected bounding boxes.

The chosen categories reflect interesting characteristics of the link and facilitates subsequent identification of tracking shortcomings. Most interesting in terms of tracking improvement are the false negative links. For each identified error, the evaluation produces a text file containing the frame number where this error is present.

5.2.2 Results and Analysis using global criteria

We have also computed two metrics. The first metric is the average percentage of the time people are tracked. It is defined as the total number of true positive links divided by the total number of expected links (ground truth):

- For video sequence V1, the first person was tracked 100% of the time. The second person which is dynamically occluded is tracked 26% of the time.
- For video sequence V2, all persons were tracked 63% of the time, on average.
- For video sequence V3, all persons were tracked 77% of the time, on average.

The second metric is the true positive rate (TP/(TP+FN)) for true positives, the false negative rate (FN/(TP+FN)) for false negatives and the false positive rate (FP/(FP+TP)) for false positives. This is illustrated in figure 5.5.

Again, these two metrics can help us to compare various algorithms and can be indicative for an end user. But they are not sufficient to know if the process is reliable.

	F2F Tracking		
	true positive rate	false negative rate	false positive rate
video V1	90%	10%	1%
video V2	84%	16%	4.4%
video V3	81.2%	18.8%	5%

Table 5.5: F2F tracking: true positive rate, false negative rate and false positive rate for video sequences V1, V2 and V3

TP	second link	first link
video V1	1.6%	98.4%
video V2	4.4%	95.6%
video V3	3.3%	96.7%

Table 5.6: F2F tracking: technical criteria percentage for true positives (TP) for video sequences V1, V2 and V3

Two other global criteria are informative (but they have not been computed yet). The first one is the maximum tracking time. For example, a person can be tracked 95% of the time but the number of track loss can be high. The second one is the number of identifier which have been attributed to a person.

5.2.3 Results and Analysis using technical criteria

We illustrate evaluation results using technical criteria in table 5.6 for the three video sequences. Technical criteria compute, for true positives, the percentage of links which were tracking process first choice or second choice. This is illustrated in the two columns (second or first link). It gives an overall view of tracking's confidence in its choice of links. We can observe that most of the links which have been found are first links.

	video V1		video V2		video V3	
TP	2nd link	1st link	2nd link	1st link	2nd link	1st link
static occl.	0%	8.1%	2.5%	36%	0.8%	24.1%
dyn. occl.	1.6%	53.5%	1%	28.3%	1.8%	20.9%
close	0%	6.5%	0%	0.3%	0%	3%
medium	0%	30.3%	0.9%	31%	0.7%	48.7%
far	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.

Table 5.7: F2F tracking: true positive (TP) attribute percentage with respect to detection criteria for video sequence V3

5.2.4 Results and Analysis using detection criteria

We illustrate true positive evaluation results using detection criteria for the three video sequences in table 5.7. Rows correspond to classification by occlusion type or camera distance. Columns show the number of links which are the tracker second or first choice, respectively.

Obviously, the tracking process has much more difficulties in the presence of occlusions. True positive results presented with both technical and detection criteria are useful to know in which situations the tracking process could have problems. For example, we can improve link creation in the medium field of view.

In table 5.8, we show the repartition of false negatives and false positives with respect to detection criteria for the three video sequences.

This classification will be useful during the repair process. For example, we could first try to repair the algorithm to avoid false negatives in the medium field of view. Then, in a second step, we could fix false negatives occurring in a static occlusion case.

	video V1		video V2		video V3	
	FN	FP	FN	FP	FN	FP
static occl.	15%	0%	56%	1%	44.9%	0.9%
dyn. occl.	70%	5.9%	23%	1%	21.6%	3.8%
close	0%	5.9%	0%	76.6%	2.2%	27.1%
medium	15%	0%	21%	1%	31.3%	3.4%
far	n.a	88.2%	n.a.	20.4%	n.a.	64.8%

Table 5.8: F2F tracking: false negatives (FN) and false positives (FP) percentage with respect to detection criteria for video sequences V1, V2 and V3

5.2.5 Results and Analysis using tracking criteria

A careful analysis of the evaluation using either global, technical or detection criteria has shown that there are two situations where the evaluation process is not sufficient to produce a correct error classification:

- Different errors of the tracking algorithm are classified as one error type.
- Similar errors are classified into different error types.

In the first case, the evaluation algorithm has to be modified to discriminate the given error type into more accurate subtypes by refining the existing evaluation criteria. The second case occurs when the classification does not match the real tracking errors. In this case, the error types have to be redefined using new criteria.

Evaluation has shown different types of errors classified as false negatives. We have refined these errors into four sub categories thanks to tracking criteria:

- Split of a person into several body parts. This error is due to an over-segmentation of a person: at time t the person is detected as one moving region and at time $t+1$ as two (or more) moving regions corresponding to different body parts

	Person		Group	
	Split	Merge	Split	Merge
V1	1	1	7	7
V2	20	12	92	108
V3	25	25	61	82

Table 5.9: Tracking errors for video sequences V1, V2 and V3

(e.g., head, body, feet). Usually in this situation, the main body part is tracked and the remaining parts are lost.

- Merge of body parts of a person into a well-detected person. This situation is similar to the previous one.

- Split of a group of people into distinct persons. Usually in this situation, one of the person is isolated and detected by a large moving region which is close to the detection of the group at the previous time. Then, the moving regions corresponding to the remaining persons are lost.

- Merge of separated persons into a group of persons. This situation is similar to the previous one.

Table 5.9 describes the four tracking errors subtypes. After analyzing all these error situations, we have found out that these four subtypes correspond to specific tracking errors.

During this process, it is of prime importance to solve the general problem without depending on one sequence. It is only possible to do so if the video sequences used for the evaluation are diverse and significant enough.

Chapter 6

Fine Evaluation

Global evaluation is useful to have an opinion about a tracker and to answer the question: how good is this tracker for my application? But this is not sufficient in most cases. We would like a way to improve the algorithm: a repair methodology. For that, we need a deep understanding of the algorithm and its limitations. The main idea is to divide the algorithm in sub steps, each one being evaluated easily. In addition, tracking errors are classified with respect to these sub steps, using repair criteria. This is the topic of this chapter. Currently, this method is only applied to the F2F tracking algorithm.

We begin this chapter by describing in section 6.1 our F2F tracker algorithm. Section 6.2 explains how we defined adequate repair criteria to be able to produce a classification of tracking errors. Finally, section 6.3 introduces the diagnosis and repair methodology.

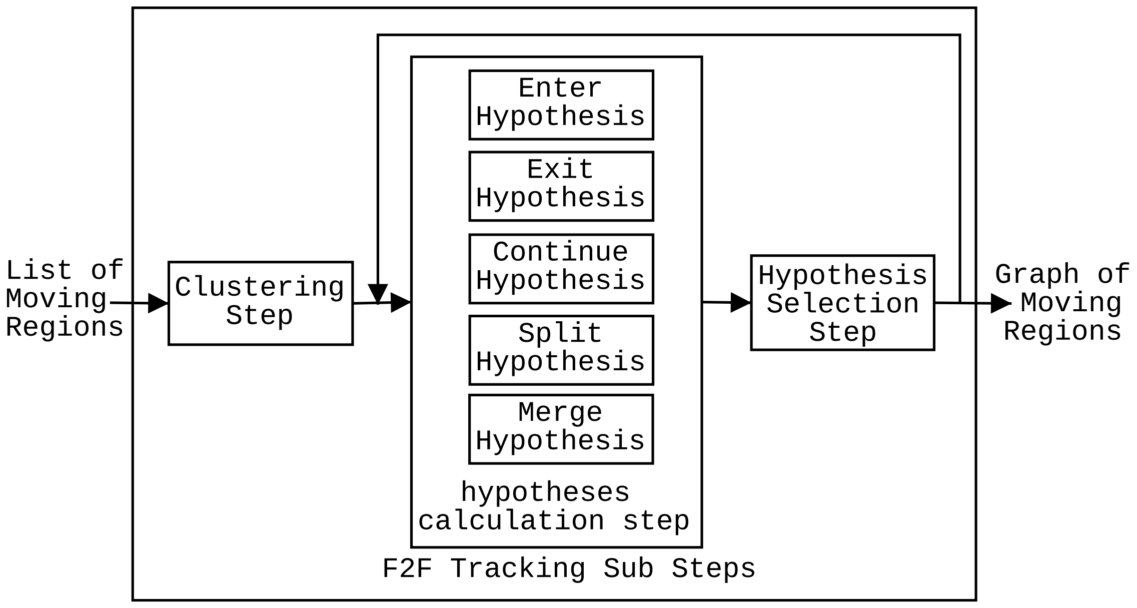


Figure 6.1: F2F tracking sub steps

6.1 F2F Tracker Algorithm

Our F2F tracking algorithm is made of 3 sub steps as illustrated in figure 6.1. The first one is a clustering step (filtering step) followed by a hypotheses calculation step. This second step is composed itself of 5 sub steps (corresponding to 5 hypothesis types). Finally, a hypothesis selection step is applied. This process is iterated until all moving regions have been linked. These steps are explained hereunder.

6.1.1 Computation of Clusters of Moving Regions

To obtain a real-time algorithm, we first apply a filtering stage necessary to limit an exponential computation time. We have a set of moving regions $N = \{n_1 ... n_p\}$ in the new frame and a set of moving regions $O = \{o_1 ... o_q\}$ in the old frame. Indeed, the set O (resp. N) has 2^q (resp. 2^p) subsets. So there are $2^{(q+p)}$ possible links between the two sets of moving regions O and N. We reduce the number of possible links by filtering out moving regions which cannot be in correspondence e.g., two blobs far away.

We compute a $p \times q$ combination matrix C to compare all elements of the sets N and O. The computation is done using a coarse 2D distance metric between bounding boxes of moving regions. Each element (i,j) of C is in the range [0..100] where zero indicates that there is no correspondence. We define that two moving regions, a new one denoted n_i (newly detected at time $t+1$) and an old one denoted o_j (already detected at time t), are in high correspondence if C(i,j) is higher than a threshold T, where T is a parameter of the algorithm.

We call a cluster of moving regions (CMR) a sub set of neighbour moving regions. Two moving regions of N are neighbours if they have at least one high correspondence in common with one moving region of O. Moreover, two moving regions are neighbours if they share a common neighbour with a third moving region (transitivity rule). Second, we extract from matrix C some CMR couples: one sub set of N called N_s and one sub set of O called O_s. These sub sets are the maximum sub sets of neighbour moving regions for which each moving region of N_s has a high correspondence with at least one moving region of O_s. A CMR couple (CMRC) contains one CMR at time t and its corresponding CMR at time $t+1$, that is $CMRC = (CMR^t, CMR^{t+1})$.

6.1.2 Discrimination Between Concurrent Hypotheses

We generate different types of hypotheses to estimate which situations can appear within a CMRC. A hypothesis corresponds to a phenomenon of the real world and attempts to explain an association between zero, one or several moving regions of the CMR^t, and zero, one or several moving regions of the CMR^{t+1}. Hypotheses are only computed for moving regions classified as PERSON or GROUP_OF_PEOPLE as hypotheses computed for NOISE or UNKNOWN will often have a lower probability. There are five hypothesis types: *enter*, *exit*, *continue*, *split* and *merge* which are computed as follows. For both CMR in a CMRC and for each moving region in a CMR, we register the number of moving regions in the other (corresponding) CMR whose bounding boxes are at least α percent in overlap, α being a parameter of the algorithm. This number determines the hypothesis type. For instance, for an already detected (old) moving region in CMR^t, we will compute an *exit* hypothesis if the number of moving regions in CMR^{t+1} which are at least α percent in overlap is 0, a *continue* hypothesis if this number is 1 and a *split* hypothesis if it is more than 1.

Once possible hypotheses within the CMRC have been determined, we evaluate accurately each hypothesis using additive knowledge such as contextual knowledge and 3D information. We have defined a function f: $O \times N \rightarrow \mathbf{R}$ between two moving regions to estimate wether the two moving regions correspond to the same mobile object. This function compares the 3D size of two moving regions, their type, their 2D distance and their 3D distance. This is the sensitive function of the algorithm and it has 11 parameters. This function is used to compute the likelihood $L(h, f)$ of each hypothesis h. The output range for L is [0..100]. This computation is specific for each hypothesis and is described below.

Finally, we choose the best scored hypothesis and we associate moving regions as described by the hypothesis. Then, we remove the processed (linked) moving regions from the CMRC. We iterate this procedure until the CMRC is emptied. We show hereunder some cues about the computation of the likelihood of each hypothesis type.

These types are defined according to the number of already detected moving regions (old) and the number of newly detected moving regions (new). Also, a *mixed* hypothesis (n olds - m news) could be investigated but is currently not for complexity reason.

enter **hypothesis (0 old - 1 new) and** ***exit*** **hypothesis (1 old - 0 new)**

Two situations can appear in an *exit* hypothesis: 1) the moving region o_a is on the camera border or on a contextual in/out zone, 2) the moving region is not exiting. In the first case, the score will be very good (i.e. $L(exit, f) = 100$) as no other available information (e.g., temporal one) at that processing stage can lead us to another conclusion. In the latter case, the score will be bad unless the moving region matches a noise n_b in the new frame. In this situation, we have $L(exit, f) = f(o_a, n_b)$. The *enter* hypothesis is similarly solved.

continue **hypothesis (1 old - 1 new)**

In a *continue* hypothesis, we compute the score between the old moving region o_a and the new moving region n_b: $L(continue, f) = f(o_a, n_b)$. We then try to further improve the score if there are some moving regions classified as NOISE in the CMR^{t+1}. We test whether the likelihood increases when we merge a noise with the person of the same frame. The better the merge is (if any) the better the hypothesis score is.

split **hypothesis (1 old - n news) and** ***merge*** **hypothesis (n olds - 1 new)**

Three main situations can occur for the *split* hypothesis: 1) we have a group at time t that splits up at time $t+1$, 2) we have a person a time t who is segmented in several parts at time $t+1$, 3) we have a group at time t that splits up at time $t+1$ with a person segmented in several parts. In addition, we can have the three previous situations with either a noise classified as a person.

Since we have one old moving region o_a, we first determine the best corresponding new moving region n_j such that $L(split, f) = max_j f(o_a, n_j)$. Then, as long as the score improves, we try to merge iteratively n_j with one of the remaining newly detected moving regions: $L^1(split, f) = max_k f(o_a, n_j \cup n_k)$. If $L^1 > L$ then $n_j = n_j \cup n_k$ and we iterate. During this process, we register all the involved newly detected moving regions.

Second, a 3D distance criterion helps us to determine which of the registered newly detected moving regions comes from an over segmentation of a person. The basic idea is that the 3D distance between several parts of the same person will be high as they will be badly projected. So we are able to distinguish which are the moving regions to be merged and which are the moving regions splitting up from a group. Noise is hardly detectable. The *merge* hypothesis is similarly solved.

6.2 Fine Evaluation Algorithm

The goal of the fine evaluation is to enable tracking process improvement by classifying and grouping tracking algorithm errors according to the code and by analyzing these errors.

The analysis means that we have to find which part of the tracking algorithm (e.g., which hypothesis or which parameter of the function f) is responsible of a given error class. This work is still in progress but we show hereunder results obtained for the clustering step.

CMR couples creation is done with a coarse 2D distance between objects. Based on the ground truth, we have designed a procedure which outputs the perfect CMR couples the algorithm should have computed. In other words, we have a procedure which is able to evaluate a primitive operator (the clustering step) of the complex operator (the F2F tracking). This first step indicated that the distance currently used to compute neighbours generates many errors. A typical error is, for instance, a missing mobile in a CMR preventing a correct hypotheses generation. The repair step will now consist in finding an appropriate criterion to modify the parameter value used for computing the distance (according to input data, context,...) or to design a new distance measure.

6.3 Repair

We present in this section, the repair methodology we are establishing to improve the tracking algorithm.

Once we have a satisfactory classification of the tracker errors and the associated diagnostic, we try to fix the tracking algorithm. We try to determine if the problem can be solved by an adequate change in a tracking parameter or if we need to introduce additive knowledge. Currently, this is a manual repair since we need a human intervention to modify the code or the parameter. For instance, to repair the clustering step we can either change the neighbourhood threshold or to redefine a new distance for computing neighbours.

While repairing, the main goal is to obtain a tracking algorithm made up by as many sections as the error classes which have been used for the evaluation. Each one of these sections aims to handle the situations which could lead to the errors belonging to the corresponding class. This correspondence between sections of the tracking algorithm and classes of errors is fundamental in order to easily and precisely determine - from the evaluation results - the parameter(s) or the need of additional knowledge which are responsible for each error.

Finally, the last step consists in re-evaluating the modified tracking process, first using the same set of video sequences and ground truth and second by extending this set with more challenging videos.

Chapter 7

Conclusion

To improve the video interpretation process, we have first focused on the weakest part of the process, the tracking algorithm. To reach this goal, we have oriented our work on the establishment of an evaluation and repair framework for tracking algorithms.

We have proposed a two level evaluation. The first one being a global evaluation which ranks tracking algorithms according to several criteria and which gives an idea of their overall performances. This first level has been shown to be general and can be applied to various tracking algorithms. The second level is a fine evaluation. The analysis of the tracking algorithm leads to a classification and diagnosis of tracking errors which enables to improve and repair the tracking process. This methodology can help tuning the numerous tracking parameters. Thanks to this method, validation on a large number of video sequences is also simplified.

This method has currently been applied to human tracking in a video surveillance application context. The studied tracker is now able to track correctly multiple people in cluttered environment. A demonstration is available on the web page http://www-sop.inria.fr/orion/personnel/Benoit.Georis/index.html.

Currently, this technique could be used in the particular case of a new camera setup. The tracking algorithm could be adjusted and parameterized according to ground truth until acceptable performances are reached. After this configuration mode, the tracker would run without supervised evaluation.

The next major goal to achieve is the design of an automatic repair process. For example, future work will emphasis on automatic parameters tuning or automatic subcode selection. Another research line will investigate the extension of this method to other tracking algorithms (long term tracking) and other parts of the interpretation process (fusion algorithm).

Chapter 8

An Evaluation Framework for Video Interpretation

The promising results obtained with our evaluation and improvement methodology applied to the F2F tracking lead us to investigate two research axes in the future. This is illustrated in figure 8.1. First, we would like to study the possibility to have a more automatic repair process, as explained in section 8.1. Second, we would like to generalize this framework to other algorithms in the video interpretation chain, as described in section 8.2.

8.1 Automatization

First, we would like to go a step further in the evaluation and repair process. Our aim is to organize the code in such a way we can easily evaluate it and repair it. A first utilization consists in tuning algorithm parameters automatically, given a video set and the associated ground truth. A second utilization is to obtain a library of functions. Each of these functions would be dedicated to manage a specific tracking

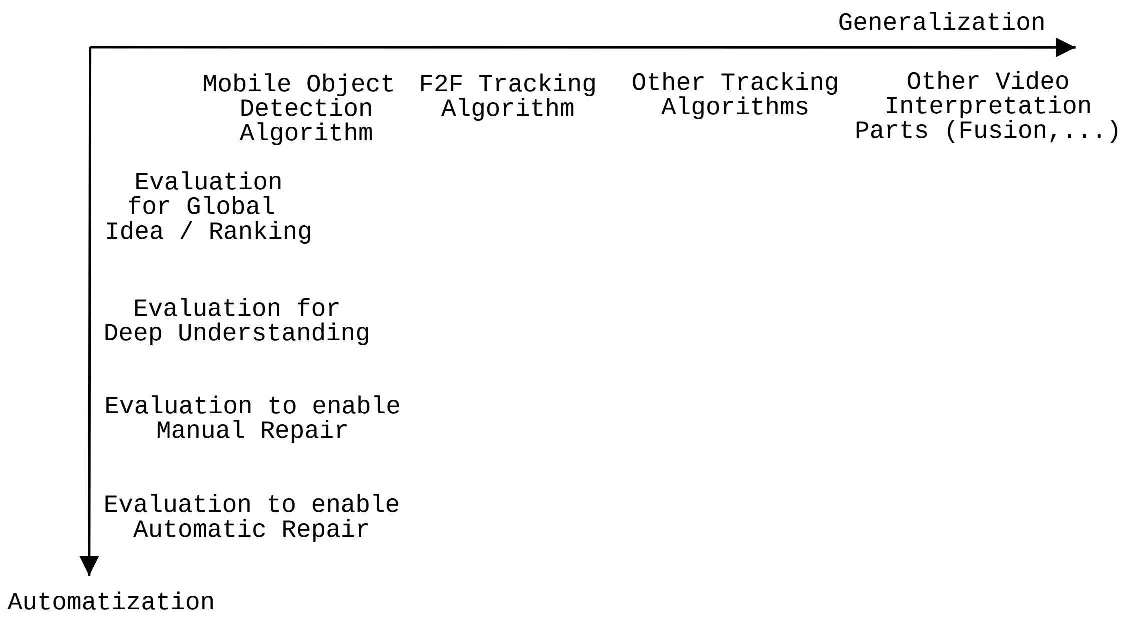

Generalization
Mobile Object Detection Algorithm
F2F Tracking Algorithm
Other Tracking Algorithms
Other Video Interpretation Parts (Fusion,...)
Evaluation for Global Idea / Ranking
Evaluation for Deep Understanding
Evaluation to enable Manual Repair
Evaluation to enable Automatic Repair
Automatization

Figure 8.1: Future research axes

problem. Then, these functions could be dynamically selected and executed with appropriate parameters.

Second, we would like to evolve to an automatic ground truth generation. For instance, a process could generate ground truth as described in section 4.2 by taking as only input the maximum number of people in the scene. There is a compromise to find between the accuracy of the ground truth we give (and the accuracy of the evaluation and repair we can do) and the degree of automatization of the ground truth acquisition process we have.

8.2 Generalization

As a matter of test and validation of our approach, we want to extend the global evaluation process to other tracking algorithms, for instance the long term tracking module or the F2F tracker coming from the Multitel lab. We also want to evaluate other parts of the video interpretation system, for example the fusion module.

Moreover, we have drawn a general methodology even for fine evaluation and repair. This methodology needs to be validated on other tracking algorithms. By doing a fine evaluation of other tracking algorithms, we could select advantages of each algorithm and use these advantages to obtain a better tracking process (e.g., a library to manage difficult cases of tracking). The evaluation and repair methodology can also be extended to other parts of the video interpretation system.

Bibliography

[1] J. Bullier, “Architecture du système visuel,” in *Proc. of the NSI’97, La perception visuelle, Aspects Multi-Disciplinaires*, May 1997.

[2] C. Wren, A. Azarbayejani, T. Darell, and A. Pentland, “Pfinder: Real-time tracking of the human body,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, 1997.

[3] A. Baumberg and D. Hogg, “An efficient method for contour tracking using active shape models,” in *Proc. of IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, (Austin), pp. 194–199, 1994.

[4] I. Cox and S. Higorani, “An efficient implementation of reid’s mht algorithm and its evaluation for the purpose of visual tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, 1996.

[5] J. Ferryman, ed., *Proceedings of the First IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, (Grenoble, France), March 31 2000.

[6] T. Ellis, “Performance metrics and methods for tracking in surveillance,” in *Proceedings of the Third IEEE Workshop on PETS*, (Copenhagen), June 2002.

[7] L. di Stefano, G. Neri, and E. Viarani, “Analysis of pixel-level algorithms for video surveillance applications,” *CIAP01*, 2001.

[8] P. Correia and F. Pereira, “Standalone objective evaluation of segmentation quality,” *WIAMIS01*, 2001.

[9] C. Jaynes, S. Webb, R. Steele, and Q. Xiong, “An open development environment for evaluation of video surveillance systems,” in *Proceedings of the Third IEEE Workshop on PETS* (J. Ferryman, ed.), (Copenhagen), June 2002.

[10] R. Vincent, M. Thonnat, and J. Ossola in *Proc. of the International Conference on Imaging Science, Systems, and Technology CISST'97*, Program Supervision for Automatic Galaxy Classification 1997.

[11] C. Shekhar, S. Moisan, R. Vincent, P. Burlina, and R. Chellappa, “Knowledge-based control of vision systems,” *Image and Vision Computing*, vol. 17, pp. 667–683.

[12] S. Moisan and M. Thonnat, “Knowledge-based systems for program supervision,” in *1st International Workshop on Knowledge Based systems for the (re)Use of Program Libraries*, pp. 4–8, Nov 1995.

[13] M. M. Lopez, *Vrification et Validation de Systmes Base de Connaissances en Pilotage de Programmes.* PhD thesis, Universitat Jaume I, February 1999.

[14] V. Clement and M. Thonnat, “A knowledge-based approach to integration of image procedures processing,” *CVGIP: Image Understanding*, vol. 57, pp. 166–184, Mar 1993.

[15] M. Thonnat, S. Moisan, and M. Crubézy, “Experience in integrating image processing programs,” in *Proceeding of the 1st International Conference on Vision Systems* (Henrik Christensen, ed.), Lecture Notes in Computer Science, (Las Palmas, Gran Canaria, Spain), Springer-Verlag 1998.

[16] S. Moisan, A. Ressouche, and J.-P. Rigault, “Behavioral substitutability in component frameworks: a formal approach,” in *8th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications*, (California), October 2003.

[17] F. Cupillard, F. Brémond, and M. Thonnat, “Behaviour recognition for individuals, groups of people and crowd,” in *IEE Proc. of the IDSS Symposium - Intelligent Distributed Surveillance Systems*, (London), February 2003.

[18] http://lamp.cfar.umd.edu/media/research/viper/.