TEC2011-25995 EventVideo (2012-2014)

Strategies for Object Segmentation, Detection and Tracking in Complex Environments for Event Detection in Video Surveillance and Monitoring

D5.1

DESCRIPTION OF DEMONSTRATORS AND APPLICATIONS

Video Processing and Understanding Lab

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Supported by

AUTHOR LIST

Marcos Escudero-Viñolo	marcos.escudero@uam.es
Álvaro García Martín	alvaro.garcia@uam.es
Rafael Martín Nieto	rafael.martinn@uam.es
Fulgencio Navarro Fajardo	fulgencio.navarro@uam.es
Diego Ortego Hernández	diego.ortego@uam.es

CHANGE LOG

Version	Date	Editor	Description
0.0	7-12-2014	Fulgencio Navarro	1, 2.1, 2.2.2, 2.2.5 and 2.3
0.2	12-12-2014	Rafael Martín Nieto	2.2.4
0.3	16-12-2014	Marcos Escudero-Viñolo	2.2.1
0.4	20-12-2014	Álvaro García Martín	2.2.3
0.5	26-12-2014	Diego Ortego	Corrections and 3
1.0	28-12-2014	José M. Martínez	Version 1

CONTENTS

1. Introduction

In a previous document, we described a framework for designing and developing distributed video processing algorithms in which the video feed can be obtained from various sources available in the Escuela Politécnica Superior of the Universidad Autónoma de Madrid (e.g., live cameras or video repositories). This framework was called DiVA (Distributed Video Analysis).

In this document we present an integration methodology which has been designed in order to standardize the integration of different algorithms and applications on this DiVA framework. Also the applications developed on this framework as demonstrators are here presented. The document is concluded with some improvements aimed at expanding the application scenario.

1.1. Document structure

This document is composed of the following chapters:

Chapter 1: Introduction to this document.

Chapter 2: Contributions developed.

Chapter 3: Conclusions and future work.

2. Contributions

This chapter compiles the contributions developed in the VPU in the scope of this project.

2.1. DiVA enhancements

2.1.1. DiVA integration methodology

DiVA platform was developed in the VPULab (Video Processing and Understanding Lab). It is framed in the third generation of video-surveillance and multi-camera distributed processing systems. It was developed to provide an environment where implement and test video-surveillance algorithms developed in the laboratory.

The main criteria followed in the design of the platform where the efficiency, the generality and the scalability. All those criteria where satisfied in the original design, however the integration of new algorithms in the platform required a great command of the system by the programmer.

To simplify the algorithm integration process an integration methodology was designed and implemented in the system [1]. It is composed of the following levels:

Level 1: algorithm design standardization

A modular scheme has been designed for an easy integration of the algorithms. However, some algorithm implementation requirements must be met in order to match with the system methodology. Those requirements are the following:

- The algorithm must be written in C++ language.
- The whole algorithm must be contained in a single class, though it may contain or use other classes.
- The algorithm class must contain these methods: a process method which receives an image from the camera, a show results method, a save results method and the constructor and destroyer methods.

This first level of integration looks as shown in the Figure 1.

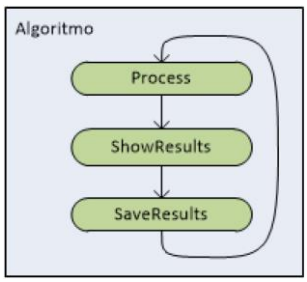

Figure 1. Integration level 1 scheme

Level 2: algorithm integration methodology

The lower level, the algorithm, is encapsulated in this level in a DiVA algorithm class which is designed to connect the different methods of the algorithm with the methods of the DiVA architecture.

The algorithm programmer just has to choose the camera to run the algorithm and the methodology will do the rest: establish the connection with the camera, give the frames to the algorithm, receive its results and show them. This level scheme looks as shown in Figure 2.

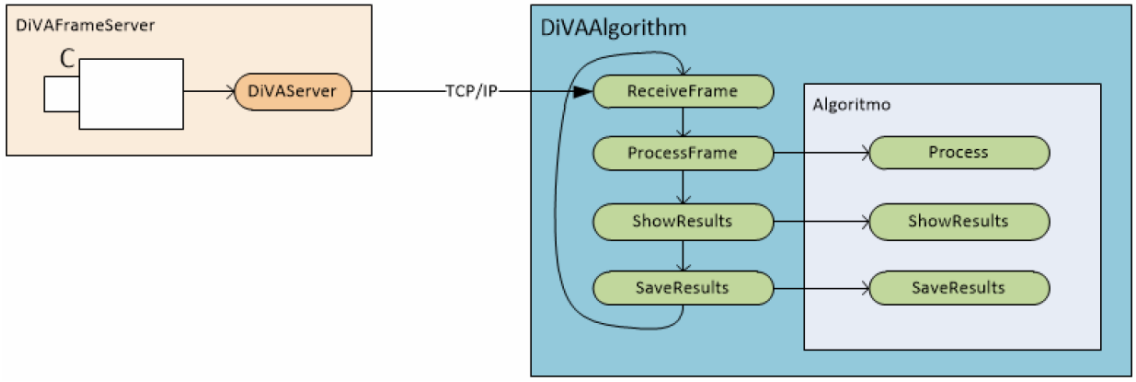

Figure 2. Integration level 2 scheme

Level 3: demonstrator graphic interface (GUI)

Once the algorithm is fully integrated and working in the system is necessary to show the results in a proper way and sometimes interact with the algorithm via changing parameters at the beginning or during the execution.

In this level, to be able to show results in the preferred way, three sublevels have been defined: developer's application, client's application and client's light application.

The developer's application allows choosing the frame server, to choose which results to show (partial and final) and to choose and modify the parameters of the algorithm, everything in the same execution interface.

The client's application is not specialized-public oriented, so it is much simpler. It allows to choose the frame server and to modify the parameters in a different window. It shows the result in the same interface.

Both sublevels present a similar scheme that is shown in the Figure 3.

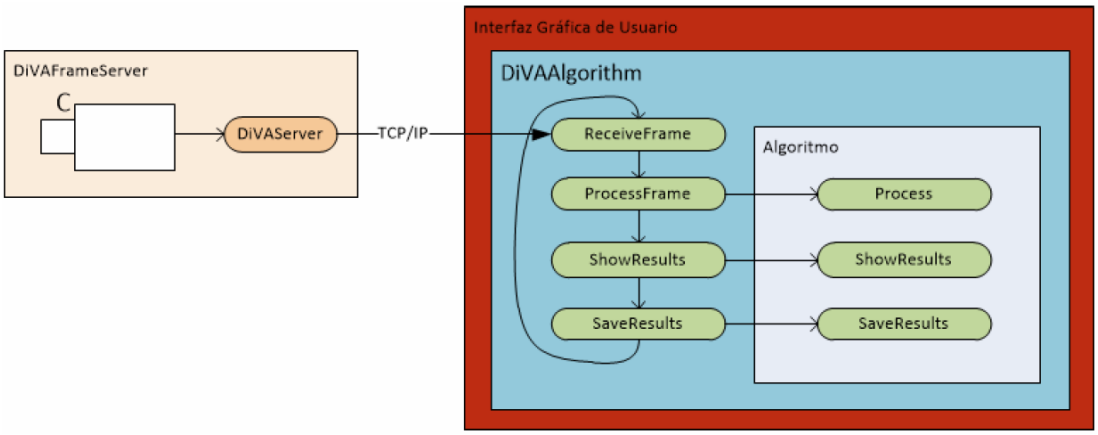

Figure 3. Integration level 3 for developer's and client's application scheme

Finally, the client's light application makes a higher use of the distributed process of the DiVA platform. It divides the complete process in at least two modules. On one side there are the algorithm modules and in the other side the demonstrator modules. Their communication is client-server oriented, and the main idea behind this distribution is to separate the algorithm process from the visualization. This allows having several algorithms running in different machines, and connecting the demonstrator application to any of them to see its results. It allows to connect and create frame servers, and to configure both the algorithm parameters and the frame server parameters. The resulting architecture is shown in Figure 4.

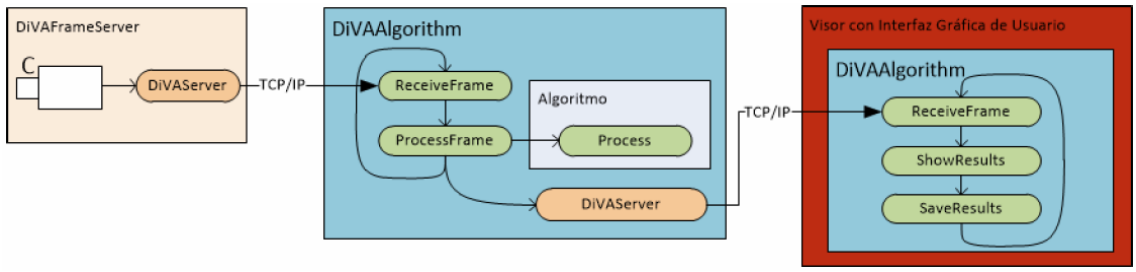

Figure 4. Integration level 3 client's light application

This integration methodology implemented on DiVA architecture improves the scalability of the system as it simplifies the work of algorithms developers. It is also a useful tool for presenting results of those implemented algorithms as it will be shown in the subsequent subsection. A predefined template is also developed in order to guarantee that all the applications present a homogeneous aspect, see Figure 5.

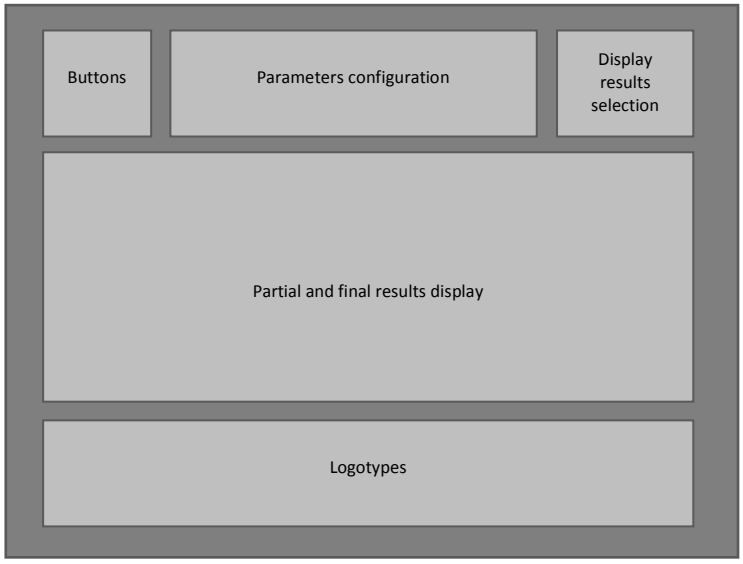

Figure 5. Applications general layout

2.1.2. Approach description

The integration methodology described above has the objective of improving the scalability and the efficiency of DiVA and also of reducing the costs of working with the platform.

To this aim, some drawbacks on the original DiVA platform were detected and corrected. The main improvements introduced are related to image cost management. The efficiency is a main task in a distributed system. If the distributed system works with video, then the efficient image management is a major objective. To manage the cost associated to each frame transmission codification options are included. The selected option must be fixed when the frame server properties are defined. The possible formats included are *portable network graphics* (PNG) and *joint photographic experts group* (JPEG).

2.2. DiVA applications

2.2.1. Background modelling

This algorithm defines an outdoor intrusion detection system in real time using a background subtraction technique based on a pixel-driven analysis.

The proposed information models rely in the use of a non-parametric scheme that stores samples of the background and the foreground using a multilayer approach. On the other hand, the proposed initialization scheme relies in a pixel classification stage to provide local-updating mechanisms under a confidence-based selection mechanism. In particular, for the updating process, the system is guided by the confidence matrix or frequency of occurrence of the modes. In turn, this confidence matrix is used to drive a selective updating process by using pixel's classes. The confidences are updated by an exponential mechanism driven by an adaptable restrictiveness parameter which evolves with the likelihood of a new mode being a representation of a modelled pixel. The models are temporarily adapted using a parametric selective updating process which controls their evolution.

Finally, in the comparison-to-models stage, two specific state-of-the-art schemes are integrated: a scheme to account for the pixel spatial environment in the comparison, and an adaptable cone-trunk-shaped pixel's mode description that provides robustness to shadows and reflections.

A user interface implemented that allows instantaneous interaction with the system has been also designed. Next section aims to describe the proposed interface. For a better understanding of the algorithm and the involved parameters please refer to [2].

The designed interface provides the user-system-interface to on-line change system's configurable parameters as well as to select the nature of the input data to the system. Next paragraphs are devoted to explain its operation. The below mentioned areas are those shown in the general layout Figure 5. Description would be done according to these areas.

The buttons area defines the input to the system: either a Camera (Camara) or a video file (Explorar) can be used; as well as activates (Iniciar) or inhibits (Detener) its operation. The parameters area provides the tools to on-line change aforementioned parameters. To the changes to be performed, they should be applied (Aplicar). The display selection area provides the tools to select / deselect the showed information. Current version includes: current frame, best background model, best model and best confidence images, foreground model, class image with static background (blue), dynamic background (green), static foreground (red) and dynamic foreground (yellow), and the binary mask with the foreground pixels (both static and dynamic) activated. Finally, the display area shows the selected information to display, see Figure 6.

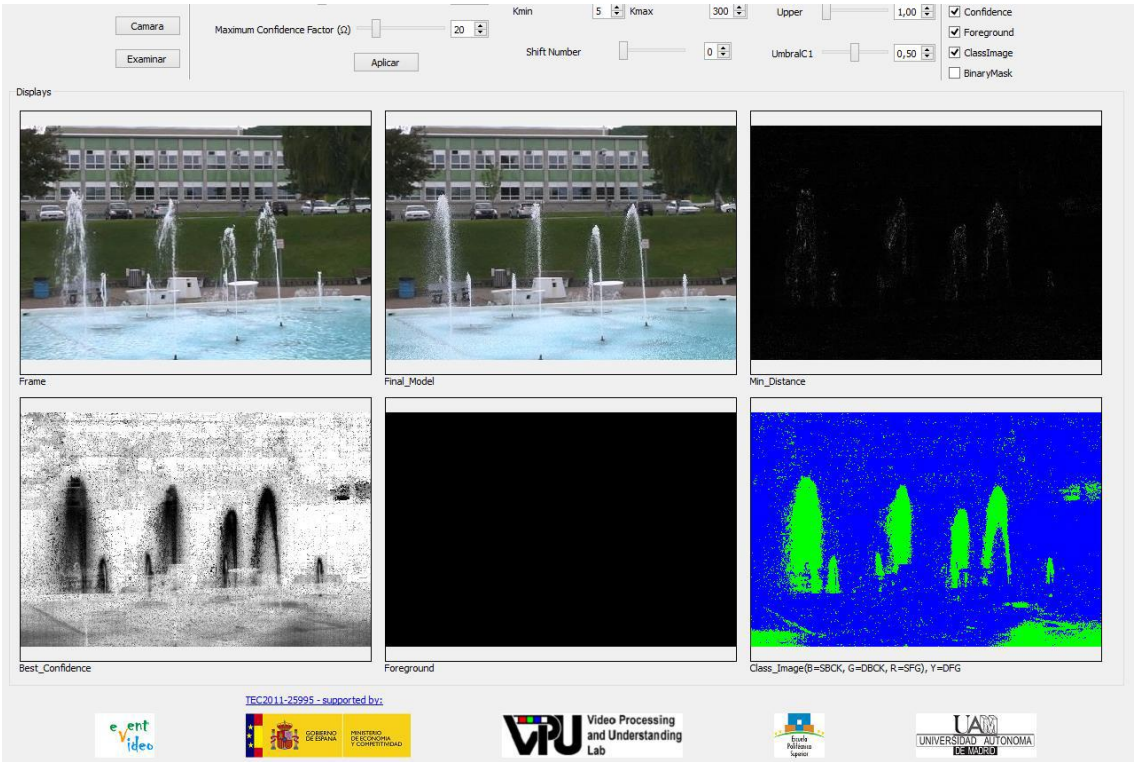

Figure 6. Representation area in Background modelling GUI

2.2.2. Stationary regions detector

This application, detailed in [1], has been developed based on a static region detector from the VPULab. This algorithm is able to extract the static regions in a video sequence. To this aim, it combines the information of three different features: structural similarity, foreground and motion.

Firstly, a background subtraction approach, which is able to learn slight background changes and global scene changes, provides foreground information. Secondly, a motion feature is included in order to handle foreground static regions caused by objects moving constantly in the same area. Finally, a structural similarity feature is included to cope with the lacks of the background subtraction approach in terms of illumination changes. The partial and final results are shown in Figure 7.

The application interface showing the algorithm working and allowing or not to modify its parameters is built in three levels. In the first level, the interface is meant for a specialized audience. The second level is meant for a non-specialized public. Finally, the third level is a division of the interface in two modules: the processing module and the representing module. Each of the three levels are detailed below.

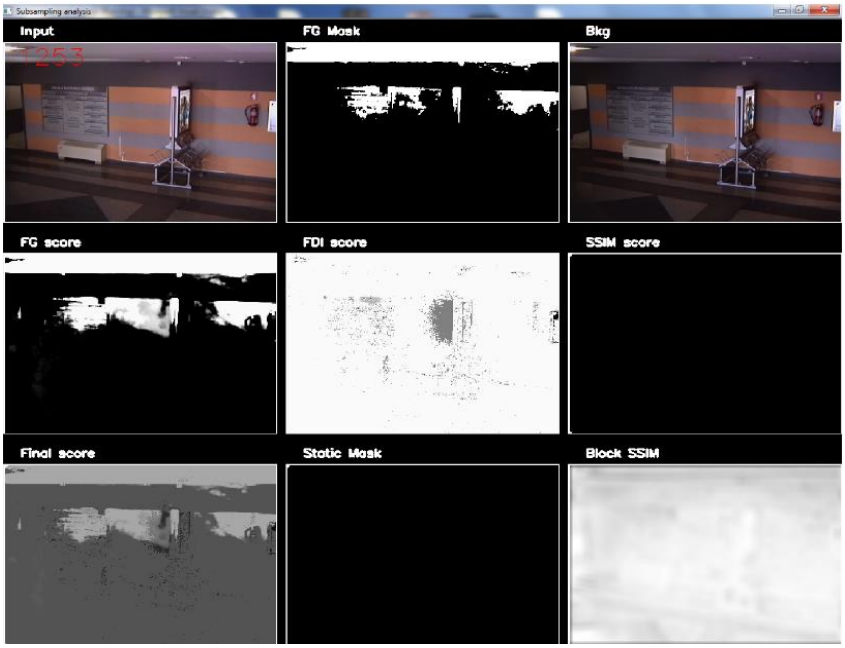

Figure 7. Input, features and static mask (algorithm output) example.

Specialized application

The specialized application layout, similar to those presented in the subsection 2.1.1, is shown in the Figure 8.

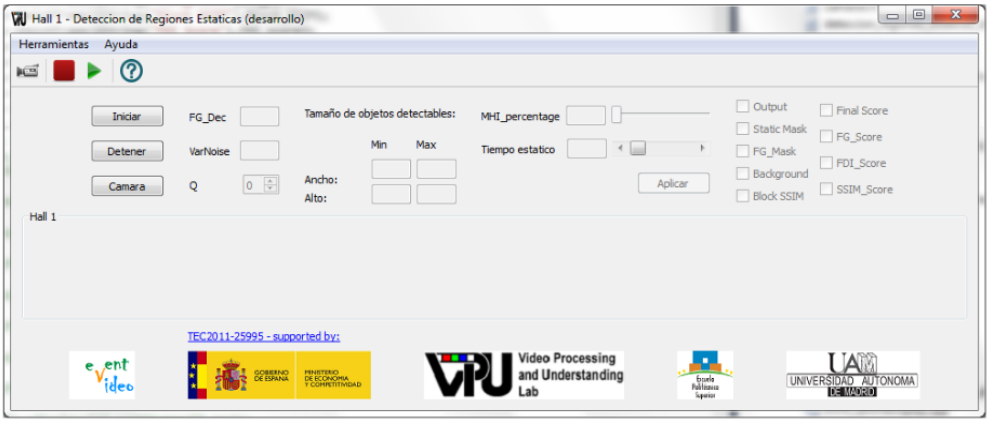

Figure 8. Static background application layout. Specialized user template.

Subsequently, each part of the window layout is detailed. In the menu bar, the option of selecting the input camera and the windows of help and about are included. In the left top corner, just below the menu bar, shortcuts to these options and to start and stop the application execution are presented.

In the buttons area, we find the same buttons of start, stop and camera selection. In the parameters area, we see a big set of parameters that can be modified in the algorithm, and a button of apply which will save and load the introduced values. At the right side, in the display selection area, a total of nine checkboxes are included to select which outputs the user want to show.

The previously selected outputs will be shown in the display area. In this area, a display size control is included, so the selected outputs will be rescaled in order to show them as big as possible depending on the space and the number of outputs selected to show. The logotypes section is included at the bottom.

The parameters included in the interface are related to the aforementioned features included in the algorithm. They control the background subtraction approach, the size of the detected object or the time necessary to consider an object static. More details about these parameters are included in the deliverable about event detection.

Non-specialized application
This application is very similar to the specialized application. The working is similar, but the interface is simplified in order to make it more accessible to non-specialized users, see Figure 9
.

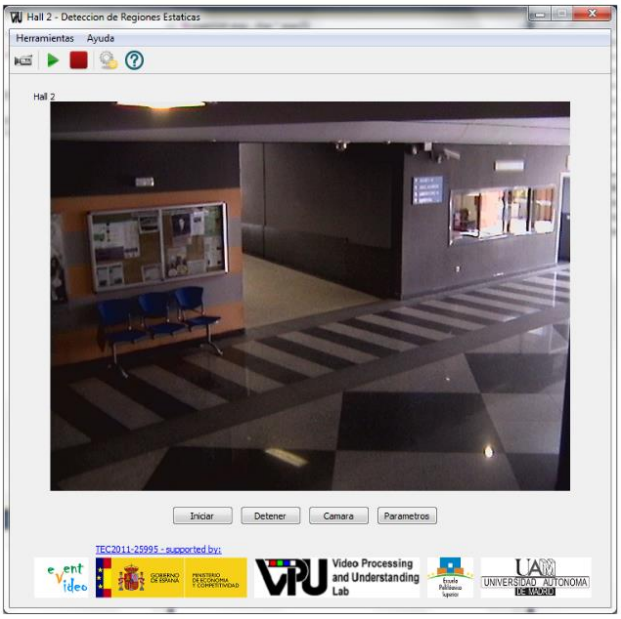

Figure 9. Static background application layout. Non-specialized user template.

The parameters area is removed, and is included in a new window accessible from the menu bar, and so it is the display selection area. The rest of the layout is presented as shown in Figure 9. The display area is place in the centre, and the buttons of start, stop, camera selection and parameters are placed below. The logotypes presented at the bottom.

Another option is included in this interface. In the menu bar, in the tools tab, an option to present a compact mode is now available. When activated, all the extra information apart from the output display is erased, see Figure 10.

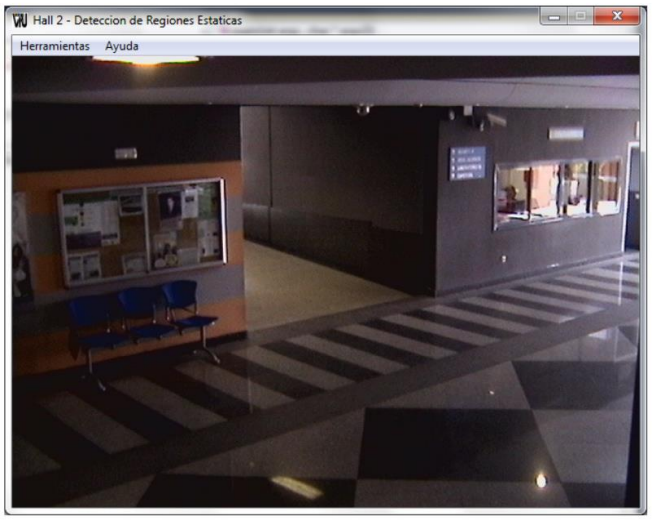

Figure 10. Static background application layout. Compact mode activated.

Two modules application

In video surveillance applications is usual to process the different algorithms in a computer and display the results in different monitors or televisions. To this aim, a two module application has been developed.

The first module is controlled by the computer where the algorithm will be running. It allows selecting the input camera and the output device. It also allows changing the parameters of the algorithm. The second module is the representing module. It is a generic interface, which connects to the first module to present the output that it is sending. It includes the compact mode option, and the buttons of start, stop and camera selection. It also includes the button to change parameters. As it is a generic interface, it will present a label informing of which algorithm's results it is presenting. An example is shown in Figure 11.

Figure 11. Static background application layout. Generic interface of the second module.

2.2.3. People detection

This application was developed using three different algorithms: Edge, Fusion and HOG. The Edge and Fusion detectors follow the same structure and main configuration parameters, therefore the graphic user interface is practically the same.

The Edge detector [3] combines segmentation and exhaustive search in order to achieve robustness and real time operation. It is a real time adaptation of the people detection approach [4]. An individual human is modelled as an assembly of natural body parts. The main idea consists of identifying characteristic edges of each body part and generating four edge models of body parts (body, head, torso and legs). The initial objects candidates to be person are extracted using background subtraction and then those selected candidates are scanned with four independent edge feature detectors previously trained.

The Fusion detector [5] is a real time detection approach based on segmentation and a holistic person model. The initial objects candidates to be person are extracted using background subtraction and the holistic person model is the combination or fusion at decision level of three simple person models: ellipse fitting, ghost and aspect ratio.

The HOG detector [6] is based on exhaustive search and a holistic person model. It consists in scanning the full image looking for similarities with the chosen person model, evaluating different detection windows with a classifier at multiple scales and locations. The chosen person model is based on appearance information using the Histogram of Oriented Gradients.

The graphic user interface (GUI) main window and parameters' window are common for both implementations (Edge and Fusion detector). The main window is shown in Figure 12. The parameters' window is shown in Figure 13.

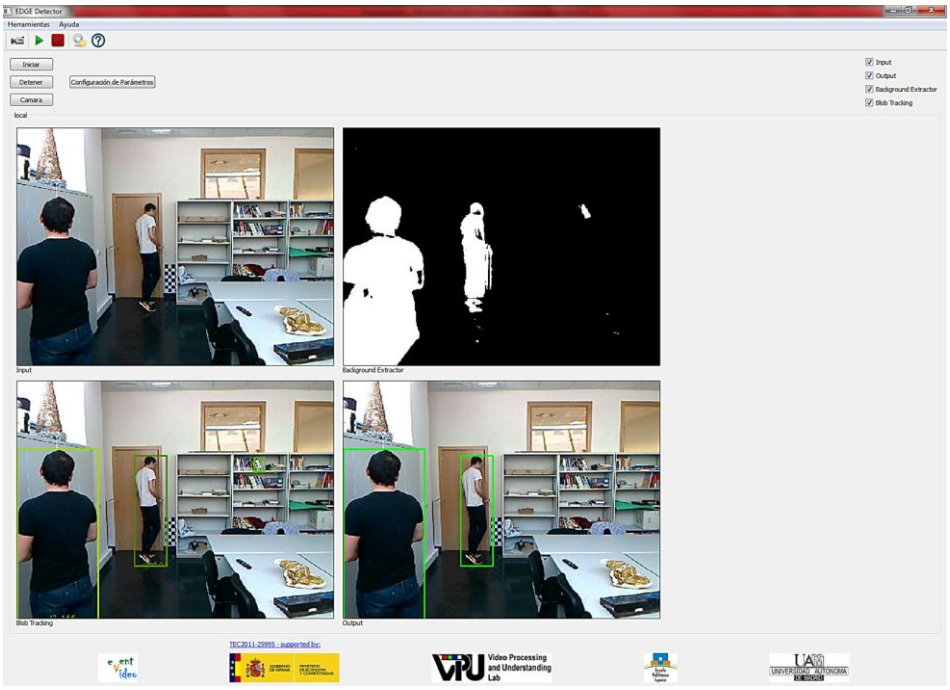

Figure 12. Edge and Fusion detector main window

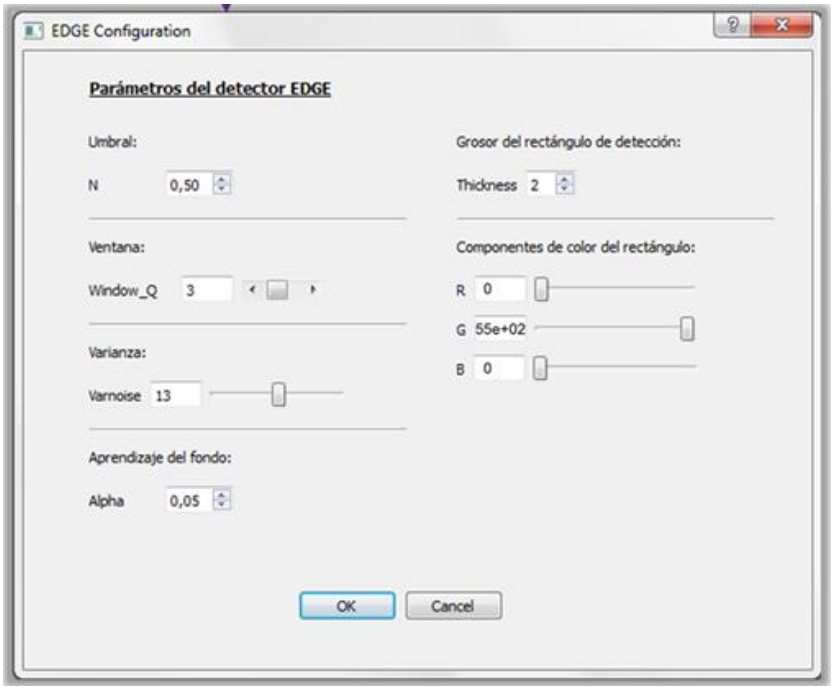

Figure 13. Edge and Fusion detector parameters' window

The graphic user interface (GUI) main of HOG detector is shown in Figure 14. Whilst the parameters' window is shown in Figure 15.

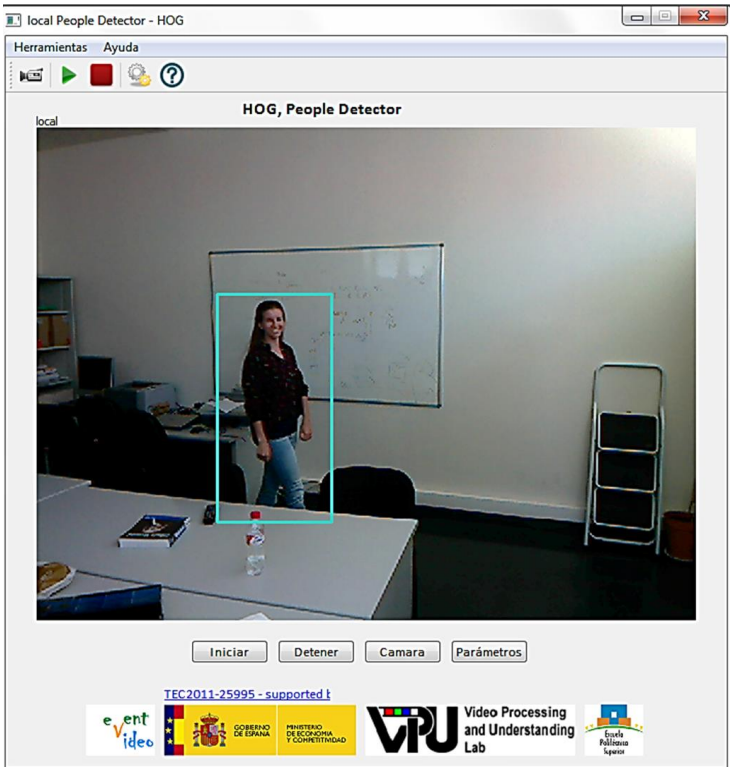

Figure 14. HOG detector main window

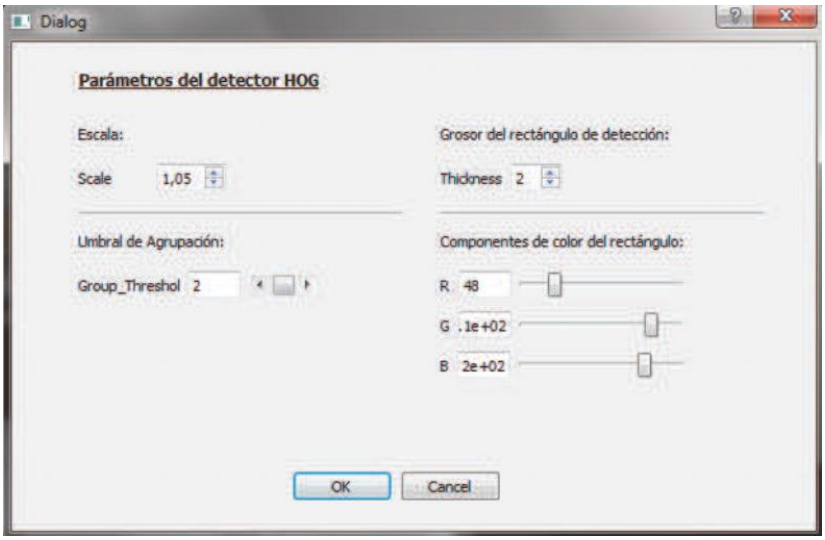

Figure 15. HOG detector parameters' window

2.2.4. Tracking

This application was developed using two different algorithms. First, a Template Matching [7] tracking algorithm (TM) was selected with the aim of establishing a methodology and a structure for later implementations. This technique is one of the simplest used in video object tracking. It main idea is to select an image patch, which may correspond to a region of interest of a frame of the sequence, and it will be used as a template. Once the template is selected, the intensities of the pixels are compared with the possible positions you can take in the next frame. Finally, the candidate position that provides greater confidence using a measurement system, typically pixel sum difference, correlation or convolution, is selected.

Moreover, a more complex algorithm was also selected. The second application uses the Point-based Lukas Kanade-Tomasi Colour Filter Tracker algorithm [8] (PKLTF). It was originally developed by Antonio González Huete in VPULab, as part of his final master thesis [9], in 2013. This tracker is composed of two phases. The first phase uses the Kanade Lukas Tomasi approach [10] (KLT) to choose the object features (using colour and motion coherence) in order to track relatively large object displacements. The second phase uses mean shift gradient descent [11] to take the bounding box to the exact position of the object model, using the object features provided by the KLT learning model.

The graphic user interface (GUI) main window is common for both implementations. It contains a robust mouse input method which allows defining a bounding box for the algorithm initialization. This window is shown in Figure 16.

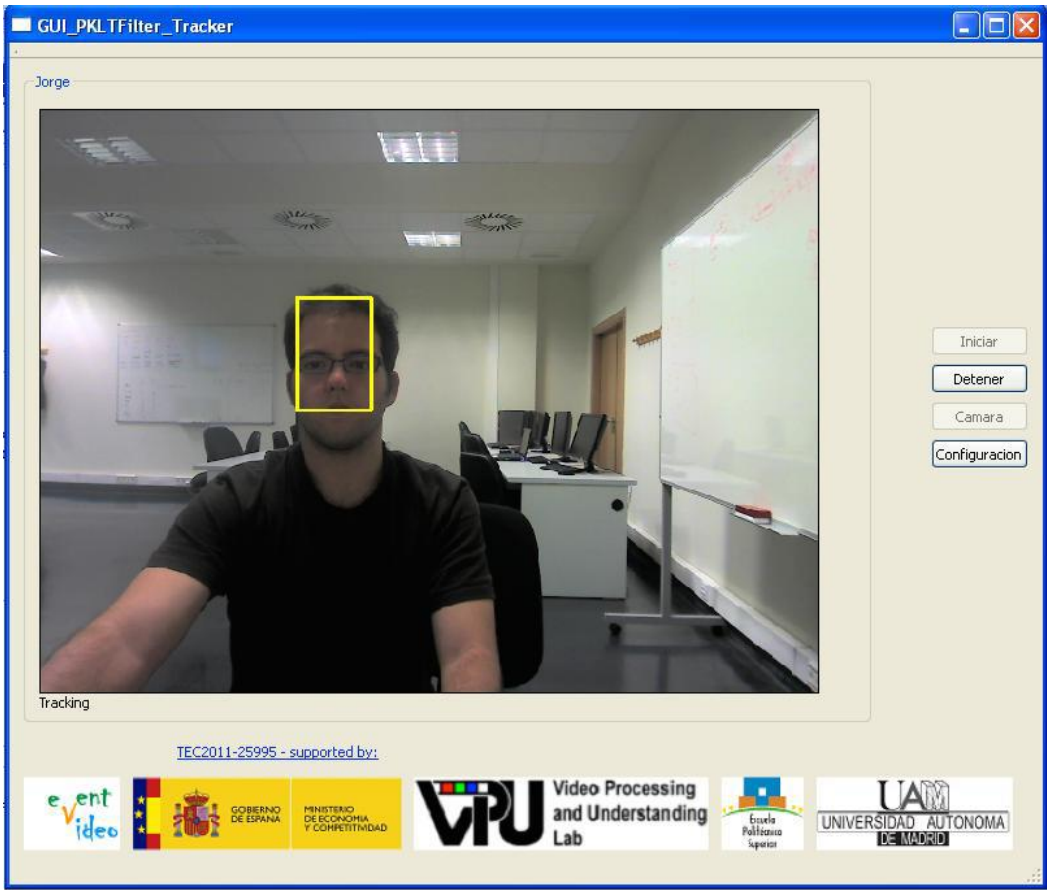

Figure 16. Main window

About the TM configuration window, only object search area is configurable. The parameter window is shown in Figure 17.

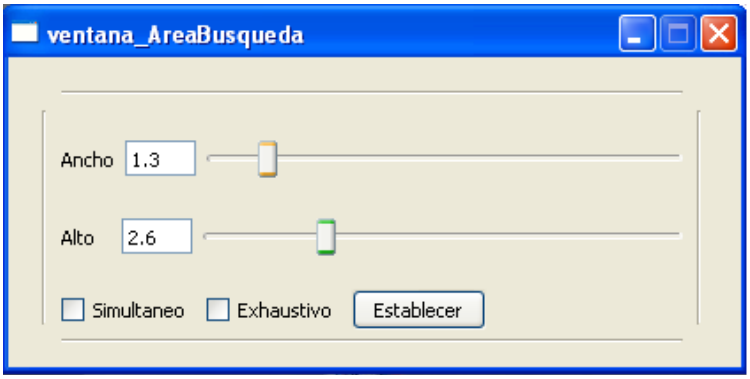

Figure 17. Template matching configuration window

In the PKLTF configuration window you can select to display the particles with which the algorithm is working, modify the parameters of each algorithm blocks, and to restore the default values stored in the configuration file of the algorithm. This parameter window is shown in Figure 18.

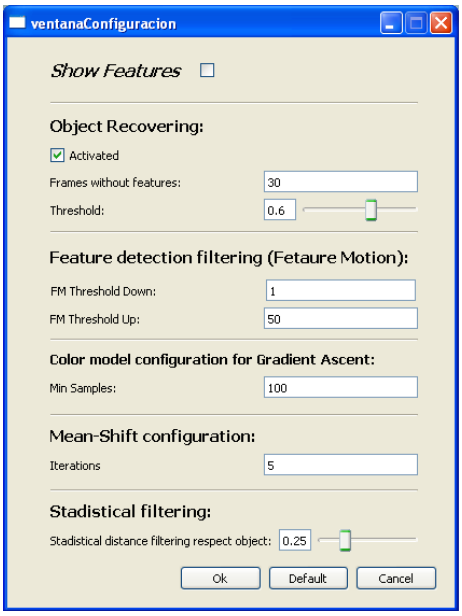

Figure 18. PKLTF configuration window

2.2.5. Anomaly detection

In this application we have implemented an existing approach from the literature [12] described in a previous documen[13]. The main contribution of the algorithm is the proposal of a framework where it models the normal t, see activity in the scene. This model is named "background behaviour image". For the generation of that background behaviour image, a train stage with normality events is first performed. That normally or anomaly activity in the scene is modelled at pixel level in motion terms. Anomalous events are detected by comparing each incoming frame of a video sequence with the generated background behaviour image, where the normal activity, i.e. scene normal motions, is trained.

Furthermore, the application algorithm includes improvements to the base algorithm for challenging scenarios. A modified size descriptor that is invariant to spatial scale is used, improving the system behaviour in challenging situations as those with small objects involved.

This application interface differs from previous ones as it needs to include a window for the training stage. This training can be done from this window, see Figure 19, or simply loaded from file, i.e. from a previously generated background behaviour image.

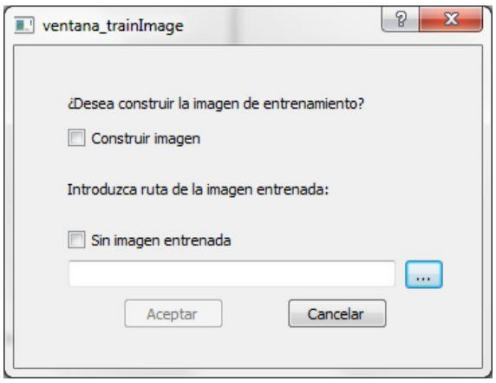

Figure 19. Training window

The main window follows the general layout for the non-specialized users. As it is shown in Figure 20, it presents the display area in the centre and below the push buttons to control the execution, the camera and the parameters settings. As it is defined in the general layout for non-specialized users, the parameters setting are displayed in another window when "Parametros" button is pressed, see Figure 21. For a detailed description of the parameters and its use see [14].

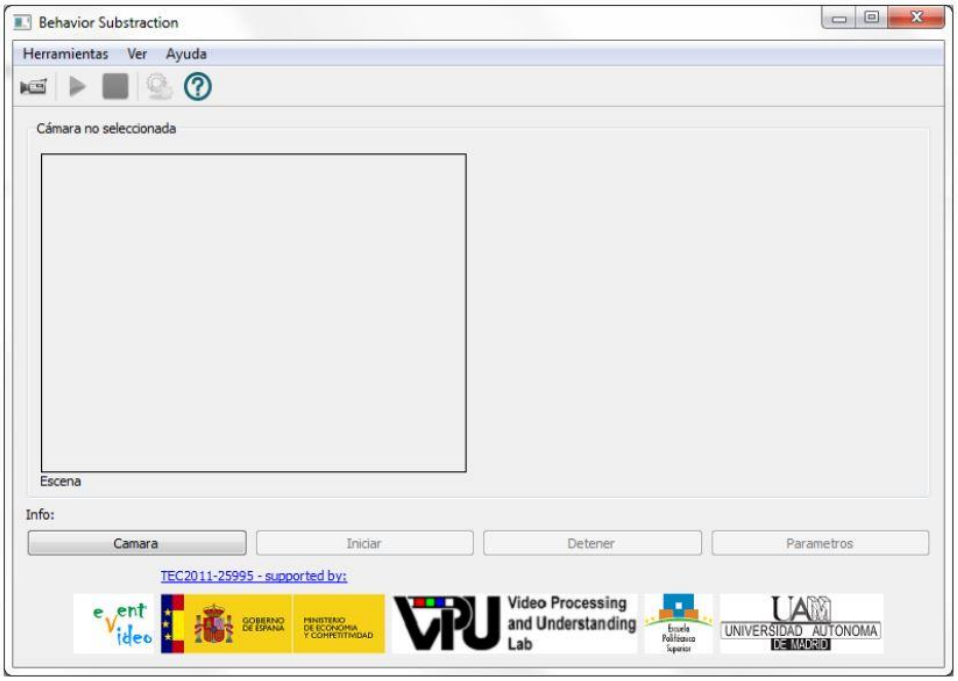

Figure 20. Main window

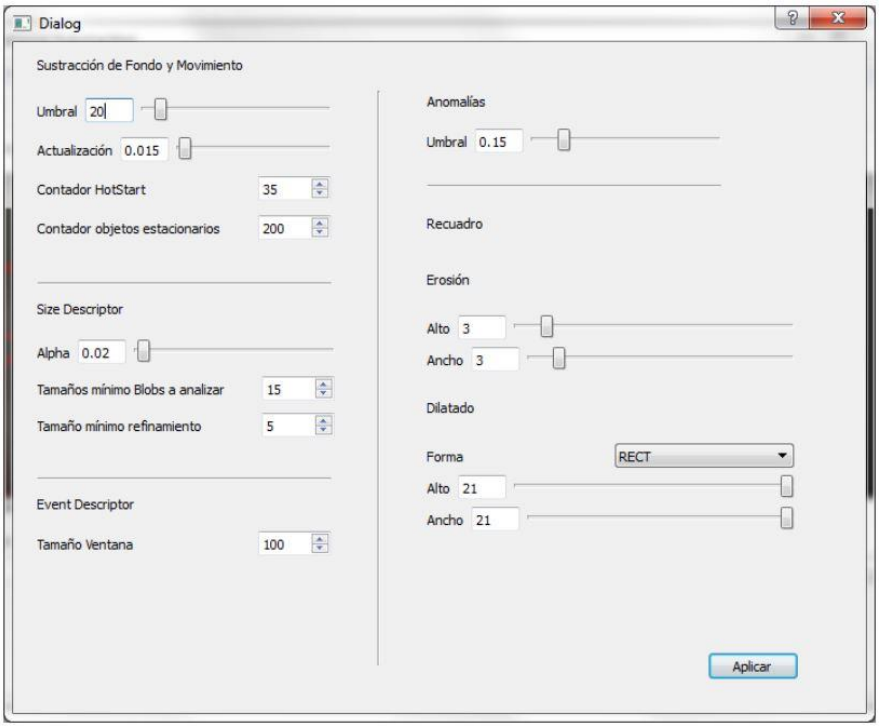

Figure 21. Parameters control window

2.3. Application scenario expansion

The utility of many techniques of computer vision in real scenarios or its extensive use is sometimes biased by the cameras' properties. In one hand, static cameras present the perfect conditions for the most algorithms' performance, but they are limited by the small field of view. Full mobile cameras in the other hand present a larger field of view. However they introduce several handicaps in the image processing as dealing with jitter, sudden camera motion or big scale changes.

To find a trade-off between static and full mobile cameras, pant-tilt-zoom cameras (PTZ) are a good choice. In the following section, a developed algorithm for panoramic generation with PTZ cameras is presented [15], and an example of the integration of a computer vision algorithm in this panoramic sweeping camera is described.

2.3.1. Panoramic generation and swapping

DiVA system is composed of several architectures all connected in one platform. One of those architectures is the capture architecture. It is composed of several cameras some of which are PTZ cameras. Those cameras are the input of this stage.

Once the video input is clear, the system disposes of a sequence of frames partially shifted with respect to the previous one. This shifted is caused by the camera continuous scanning. The idea of this improvement is to generate a panoramic background image, and then scan it continuously in order to apply other video processing algorithm.

To perform the panoramic generation a linear process based on points of interest (PoI) and homographies estimation is followed. The main stages are those showed in the following scheme (see Figure 22). The panoramic generation is developed in several stages:

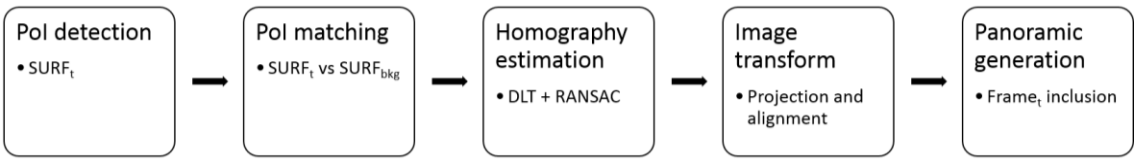

Figure 22. Panoramic generator scheme.

The PoI detection is performed via SURF keypoints. In applications where image transformations via homographies are necessary, the most precise keypoints are chosen, despite its possible high computational cost. However, in this application, the continuous scanning of the camera requires a keypoint method able to work close to real time. SURF keypoints are presented in the state-of-the-art as precise and with low computational cost, so they are chosen because of fitting the required properties.

The homography is done with a DLT method. This simple method yield good results when the points used are reliable. The selection of those points is refined with RANSAC in order to avoid the possible false positive of the PoI matching stage. The transformation of an image with respect an original one is done via a concatenation of the previous homographies (see Figure 23). A homography H_n is estimated as $H_{n-1}*H_{n-1,n}$.

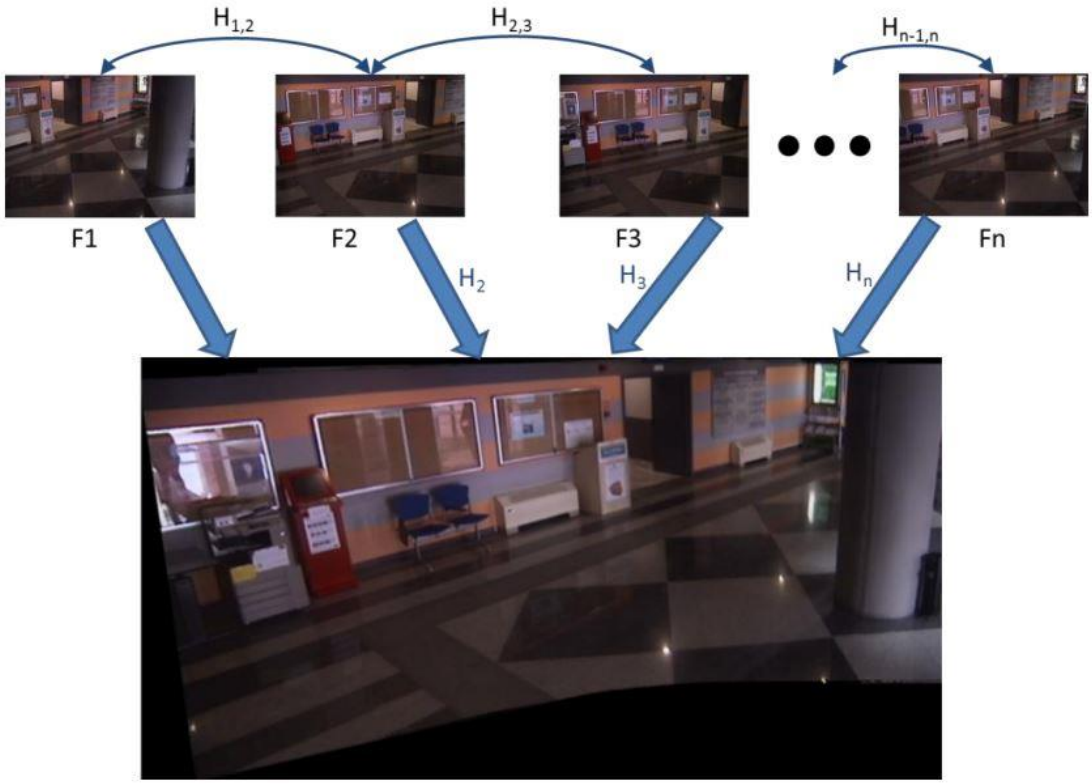

Figure 23. Composition of the panoramic transforming each frame F_1 to F_n with the homographies H_1 to H_n.

Once the homography is estimated, and the image is transformed, the next stage is the inclusion of that frame in the panoramic. The inclusion of new frames in a panoramic must be done carefully. As overlapping areas exist between the incoming frame and the panoramic that is being generated or swept, a decision has to be made about the values of the pixels on the overlapping areas. Averaging their values will result in a low quality panoramic, as the real values of the image will be smoothed. The fusion is made using the mode measure. The resulting values of each pixel are existing values so the quality of the resulting panoramic is higher.

The PTZ camera keeps scanning the visual field after the panoramic is generated. The new incoming frames are included in the panoramic so the background can be updated. In this point, the new application scenario is ready to include new algorithms on it.

A testing application is included as a demonstrator of the capacities of this improvement. A method for background subtraction is included. The algorithm is based in the gamma method. It is integrated in the different DiVA levels exposed in the previous chapters, and some modification on its process timeline are included in order to process a background composed of a set of images. The result is a background subtraction algorithm running in a scenario that covers a wider area than a static camera, but with the same reliability. An example is show in Figure 24.

Figure 24. Example of an incoming frame included on the panoramic and the result of the background subtraction algorithm.

3. Conclusions and Future Work

This document has described the methodology designed to ease the integration of video-surveillance algorithms into the DiVA framework. This methodology standardizes the integration process via three adaptation levels. In the first level, the algorithm is adapted to contain three methods. In the second level the algorithm is encapsulated in the DiVA architecture and its three methods are connected with the DiVA methods. Finally, in the third level, the algorithm results are properly shown via graphic interfaces which have different designs and functionalities depending on the kind of final user.

Furthermore, six examples of integration using this methodology have been included. Algorithms for background subtraction, stationary region detection, anomaly detection, people detection and tracking have been successfully integrated in the DiVA architecture, thus demonstrating the suitability of the proposed methodology.

For the future, other extensions and improvements will be made on the global system, like adaptation of the system to work under Linux (using POSIX threads for multitasking scheduling).

In the line of displaying results, an adaptation to portable devices can be included, i.e. android or IOS support. The option of showing the results in different devices at the same time is possible with the displaying mode: *two modules*, but it has not been tested already.

The data acquisition system can also be improved. Currently, information from RGB cameras is managed. However, it is interesting to be able to manage information from other types of cameras as IR or depth cameras. Also the connectivity to these cameras can be improved. Connection to IP cameras, FireWire cameras and USB cameras are included in this version, but Wi-Fi or Bluetooth control could be an interesting field to explore.

References

[1] C. Sánchez Bueno and JM. Martínez, "Entorno de desarrollo de aplicaciones de video-seguridad multicámara", (tutor: José M. Martínez), Proyecto Fin de Carrera (Master Thesis), Escuela Politécnica Superior, Univ. Autónoma de Madrid, Jun. 2014

[2] D2.1 Segmentation for static cameras. TEC2011-25995 EventVideo.

[3] A. Garcia-Martin and J.M. Martínez. Robust real time moving people detection in surveillance scenarios. In Proc. of AVSS, pages 241-247, 2010.

[4] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In ICCV, pages 90–97, 2005.

[5] V. Fernandez-Carbajales et al. "Robust People Detection by Fusion of Evidence from Multiple Methods", Proc. of Int. Workshop on Image Analysis for Multimedia Interactive Services, pp. 55–58, 2008.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Proc. of CVPR, volume 1, pages 886–893, 2005.

[7] R. Brunelli. Template Matching Techniques in Computer Vision: Theory and Practice. Wiley Publishing, 2009.

[8] R. Martín and J.M. Martínez: An automatic system for sports analytics in multi-camera tennis videos. Activity Monitoring by Multiple Distributed Sensing (AMMDS) Workshop 2013. In Proc. of Int. Conf. on Advanced Video and Signal-based Surveillance (AVSS). pp 438-442. 2014

[9] A. González, "Seguimiento y producción automática mediante cámaras PTZ en entornos de red", (tutor: Jesús Bescós), Proyecto Fin de Máster (Master Thesis), Escuela Politécnica Superior, Univ. Autónoma de Madrid, 2013.

[10] J. Shi and C. Tomasi. Good features to track. In Computer Vision and Pattern Recognition, 1994," in Computer Vision and Pattern Recognition, pp. 593-600, 1994.

[11] D. Comaniciu, V. Ramesh, and P. Meer, Real-time tracking of non-rigid objects using mean shift. In Computer Vision and Pattern Recognition, pp. 142–149, 2000.

[12] L. Caro, L" Anomaly Detection in Video Sequences", (tutor: Juan Carlos San Miguel), Trabajo Fin de Master (Master Thesis), , Escuela Politécnica Superior, Universidad Autónoma de Madrid, Oct. 2013.

[13] D3.2 Events detections in dense environments. TEC2011-25995 EventVideo.

[14] A. Huélamo, "Detección de anomalías en tiempo real", Trabajo Fin de Grado (Graduate Thesis), Escuela Politécnica Superior, Universidad Autónoma de Madrid, Jul. 2014

[15] A. Palero, "Detección de intrusion con cámaras móviles en tiempo real". (tutor: Jesús Bescós), Trabajo Fin de Grado (Graduate Thesis), Escuela Politécnica Superior, Universidad Autónoma de Madrid, Jun. 2014.