TEC2014-53176-R HAVideo (2015-2017-2018)

High Availability Video Analysis for People Behaviour Understanding

D4.2.2 v2

Applications

Video Processing and Understanding Lab

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Supported by

AUTHORS LIST

Álvaro García Martín	alvaro.garcia@uam.es
José M. Martínez	josem.martinez@uam.es
Marcos Escudero Viñolo	marcos.escudero@uam.es
Rafael Martín Nieto	rafael.martinn@uam.es
Juan Carlos San Miguel Avedillo	juancarlos.sanmiguel@uam.es

HISTORY

Version	**Date**	**Editor**	**Description**
0.1	10 November 2017	Álvaro García Martín	Initial draft version
0.9	10 December 2017	Álvaro García Martín	Final Working Draft
1.0	12 December 2017	José M. Martínez	Editorial checking
1.1	3 December 2018	Rafael Martín Nieto	Initial draft version 2.0
2.2	5 December 2018	Marcos Escudero Viñolo	Contributions
2.2	11 December 2018	Álvaro García Martín	Contributions
2.1	15 December 2018	Rafael Martín Nieto	Contributions
2.2	17 December 2018	Juan Carlos San Miguel Avedillo	Contributions
2.3	19 December 2018	Rafael Martín Nieto	Final Working Draft
2.0	20 December 2018	José M. Martínez	Editorial checking

CONTENTS:

1. Introduction

1.1. Motivation

The work package 4 (WP4) aims at evaluating and integrating the algorithms developed within WP1, WP2 and WP3, in order to conform the global analysis chain to provide solutions to long-term video analysis for people behaviour understanding. In particular, this deliverable describes the work related with the task T.4.2: Use Cases and Demonstrators.

The objective of this task is the development of demonstrators of the algorithms developed in the project, providing applications both for developers (for evaluation and testing) as well as for final users (use cases to be defined with the help of the Observing Partners). The use cases will focus on the applications areas of the surveillance and people monitoring challenges targeted by the project (e.g., outdoor surveillance, people monitoring in malls, in-home monitoring).

1.2. Document structure

This document contains the following chapters:

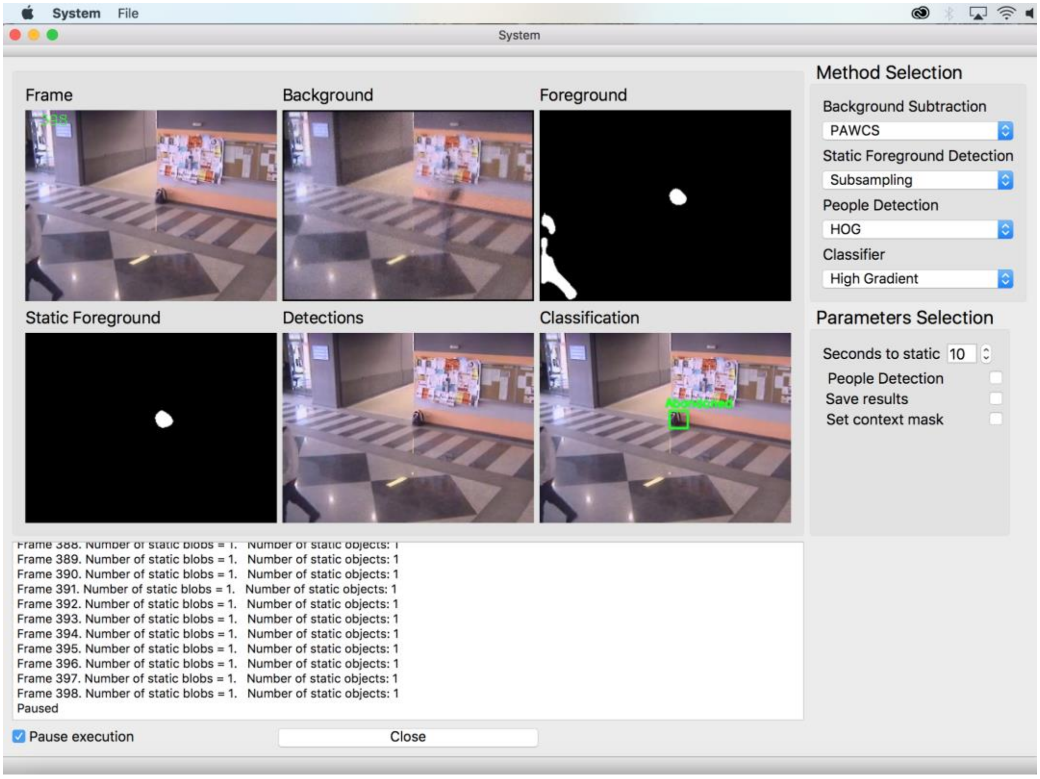
- Chapter 1: Introduction to this document
- Chapter 2: Set of applications
- Chapter 3: Conclusions

2. Applications

2.1. A complete abandoned object detection system demonstrator

A graphical user interface (GUI) has been developed to act as a demonstrator of the complete abandoned object detection (AOD) system. This GUI allows the user to check the system functionality in a visual way and to manually set the system algorithms and parameters [19].

The development environment used for the interface creation has been Qt Creator (version 4.2.1) based on Qt 5.8.0. An example of the working interface is shown in the picture below (see **Figure 1**):



Method Selection

Background Subtraction: PAWCS

Static Foreground Detection: Subsampling

People Detection: HOG

Classifier: High Gradient

Parameters Selection

Seconds to static: 10

People Detection □

Save results □

Set context mask □

```
Frame 388. Number of static blobs = 1.   Number of static objects: 1
Frame 389. Number of static blobs = 1.   Number of static objects: 1
Frame 390. Number of static blobs = 1.   Number of static objects: 1
Frame 391. Number of static blobs = 1.   Number of static objects: 1
Frame 392. Number of static blobs = 1.   Number of static objects: 1
Frame 393. Number of static blobs = 1.   Number of static objects: 1
Frame 394. Number of static blobs = 1.   Number of static objects: 1
Frame 395. Number of static blobs = 1.   Number of static objects: 1
Frame 396. Number of static blobs = 1.   Number of static objects: 1
Frame 397. Number of static blobs = 1.   Number of static objects: 1
Frame 398. Number of static blobs = 1.   Number of static objects: 1
Paused
```

☑ Pause execution

Figure 1. Graphical user interface or demonstrator interface.

1. **GUI elements.**

- Menu bar. This bar is placed on the top of the screen and the "file" option allows the user to select the desired vile file to process from the file explorer. Several formats (*.mpg, *.avi, *.mov, *.mp4, ...) are supported.

- Inside the main window of the interface the user can find:
    - Display area. This area allows the user to visualize the results of the system after each stage thereof. On top row from left to right, the current frame, the computed background and the foreground mask are displayed. On bottom row, the static foreground mask is displayed first, the people and object detections and finally the stolen/abandoned object classification.
    - Algorithm selection. Allows the user to choose a different algorithm for each module of the system by selecting it in a drop-down menu. Implemented algorithms at each stage of the system are listed below.
- Parameters selection:
    - Seconds to static. This option allows the user to modify the number of seconds until an object is considered as stationary. It is set to 10 seconds by default.
    - People detection. If this checkbox is activated, the people detection is running throughout the full sequence, otherwise it only will be run when something static is detected.
    - Save results. If it is checked a .xml file with the detections found along the sequence will be created and saved.
    - Set context mask. This option allows the user to select an area of the frame where the detections will be ignored (non-interest area).
- A dialog box where results are displayed in real time.
- Pause and close execution options.

2. Current functionalities.

- Reading stored video sequences in several video formats (*.mpg, *.avi, *.mov, …).
- Algorithms selection for each system stage.
- Parameters selection.
- Displaying results after each system stage.

- Saving results in .xml files.

3. Implemented algorithms.

a. Background subtraction module

[1] Mixture Of Gaussians (MOG) background subtraction models the background by using several Gaussian distributions at each pixel location.

[2] K-Nearest Neighbors background subtraction is an improved version of MOG.

[3] Kernel Density Estimation (KDE) is a non-parametric modeling method that estimates the density function directly from the data.

[4] Independent Multimodal Background Subtraction (IMBS) algorithm is based on the discretization of the color distribution of each pixel by applying a grouping algorithm to generate the background model.

[5] LOcal Binary Similarity segmenTER (LOBSTER) based on Local Binary Similarity Pattern (LBSP) features and color information.

[6] Pixel- based Adaptive Word Consensus Segmenter (PAWCS) is an improved version of LOBSTER and based also on background word consensus.

[7] Self-Balanced SENsitivity Segmenter (SUBSENSE) is on the same basis than PAWCS, but in addition it includes automatic adjustments of local sensitivity.

b. Static foreground detection module

[8] Temporal accumulation of the pixel’s persistence.

[9] Subsampling approach also analyzes persistence via foreground subsampling.

[10] History Images is a multi-feature detector combining foreground, motion and structural information.

[11] Dual Background Model computes static foreground by comparing two background models (short and long-term)

[12] Triple Background Model computes static foreground by comparing three background models (short, medium and long-term)

c. People detection module

[13] Histogram of Oriented Gradients (HOG) applies exhaustive search based on holistic appearance descriptors along the whole image.

[14] Haar-like features classifier is based on a trained holistic person model.

[15] Deformable Part-based Model (DPM) is a part-based person model.

[16] Aggregated Channel Features (ACF) is a detector also based on exhaustive search and a holistic model.

d. Abandoned object classification module

[17] High Gradient (HG) and Color Histogram (CH) approaches. The former analyzes high-gradient value points along the object shape while the second only considers color information.

[18] Pixel Color Contrast (PCC) method combines edge and color information.

2.2. A multi-camera pedestrian detector with semantic constraining demonstrator

The main functionality of the developed application is to facilitate the use and configuration to the final user of the different algorithms and strategies implemented in the final application. This application aims to implement a multi-camera pedestrian detector with semantic constraining. Two main interfaces have been created depending on the final user [20].

The developer interface version allows to:

First, visualize all the video sequence from the video cameras. These sequences may be loaded through the dedicated menu for opening video files, or cameras IP can be loaded so network streams are used in the case of IP cameras (see **Figure 2**).

In addition to the video sequence, the interface allows to visualize both intermediate results and outputs from the developed algorithms (pedestrian detection, semantic segmentation, usage rate extraction…) over the camera frames. This information is intended for the developer user.

Also, the interface represents the common reference plane with all the projected information that is used in the multi-camera system. In this plane one can observe projected pedestrian in addition to the desired semantic information. In the image it has been decided to project on to the reference plane those path-labelled areas by the semantic segmentation algorithm.

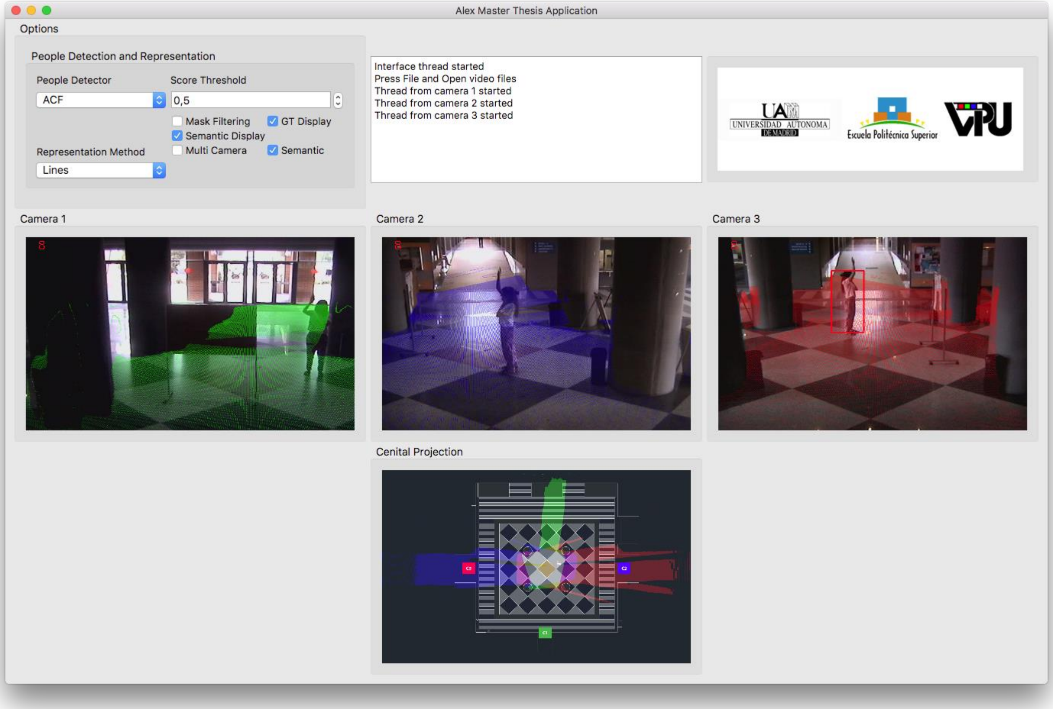
Finally, the developer version has a configuration area that allows the user to:

Choose the desired configuration to perform pedestrian detection task. Configuration includes both the algorithm and the online selection of the detection threshold. In the options one can choose between several algorithms such as Histogram of Oriented Gradients (HOG), Deformable Parts Models (DPM), Aggregate Channel Features (ACF) or Fast-RCNN. The interface also allows to enable or disable the semantic constraining or the multi-camera fusion algorithm.

From the interface one can modify the projection representation for pedestrian detections onto the reference plane. Detections may be represented as lines obtained from the bounding boxes or as Gaussian functions.

Furthermore, and if files have been previously loaded, the interface allows the user to visualize simultaneously ground-truth information.

Finally, there exists an information area from where the user gets all the necessary information to proceed if the interface is in idle state. When the application is working this area will display several information about the video files, the algorithms and the output paths.



Alex Master Thesis Application

Options

People Detection and Representation

People Detector: ACF

Score Threshold: 0,5

□ Mask Filtering ☑ GT Display
☑ Semantic Display
□ Multi Camera ☑ Semantic

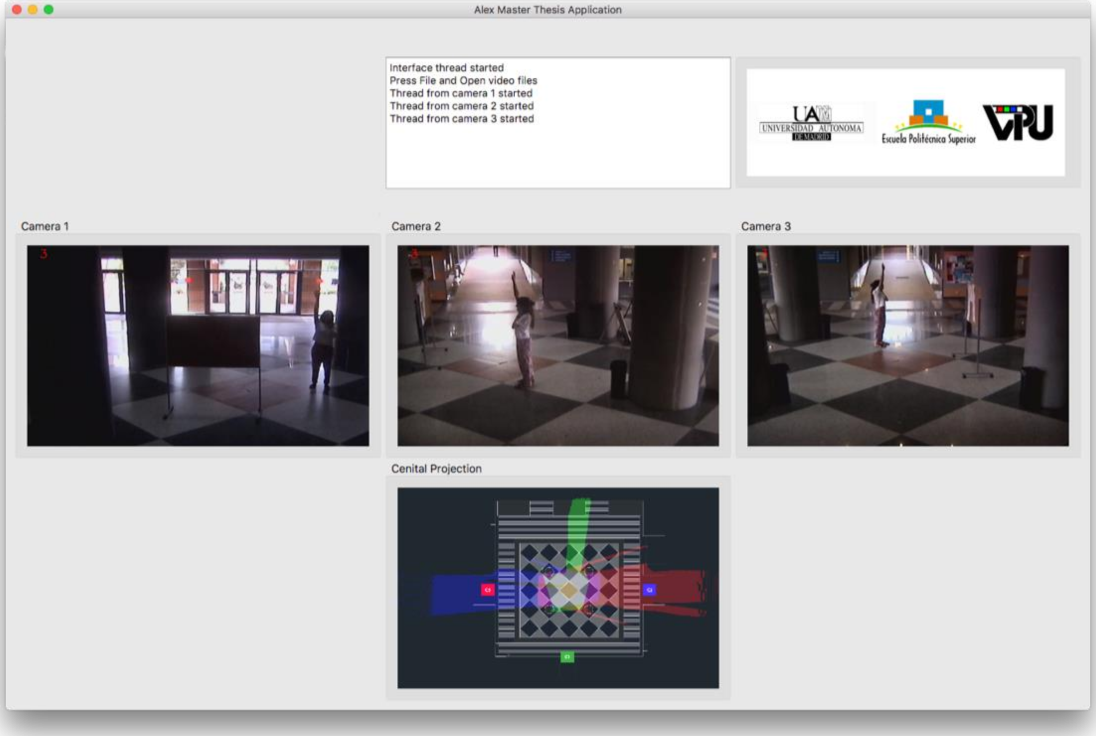
Representation Method: Lines

```
Interface thread started
Press File and Open video files
Thread from camera 1 started
Thread from camera 2 started
Thread from camera 3 started
```

Figure 2. Developer demonstrator interface.

Client version:

This version of the interface is intended for the final user. It allows the user to run the program without any knowledge about the algorithms behind the application. Due to this reason, it only contains the display and the information area. The configuration area has disappeared, and all the tuneable (see **Figure 3**).



Interface thread started
Press File and Open video files
Thread from camera 1 started
Thread from camera 2 started
Thread from camera 3 started

Figure 3. Client demonstrator interface.

2.3. A long-term tracking with target re-identification demonstrator

The objective of this work is to improve the performance of an existing tracker, called PKLTF (Point-based Kanade Lucas Tomasi colour-Filter). A newly improved tracker is designed considering the problems that affect the base tracker. Several improvements are tested, some of which are integrated into the proposed version SAPKLTF (Scale Adaptive Point-based Kanade Lucas Tomasi colour-Filter) 32[21].

A demonstrator application is developed in order to show the operation of the tracker in real situations, to facilitate the understanding of the algorithm and the influence of the parameters on the algorithm performance (see **Figure 4**).

In this case, the PKLTF demonstrator developed in a previous work [22] is used as starting point. This application has been updated with the SAPKLTF tracking algorithm.

The application gives a simple way to interact with the algorithm. Firstly, the user must choose the camera from which the application receives the video streaming; it can be a local camera or an IP camera. Once the camera, is selected the user can start running the demonstrator. To initialize the algorithm, the user must select the bounding box that defines the target with the mouse.

Through the configuration button, the user can change the parameters of the tracker, as well as different display options.

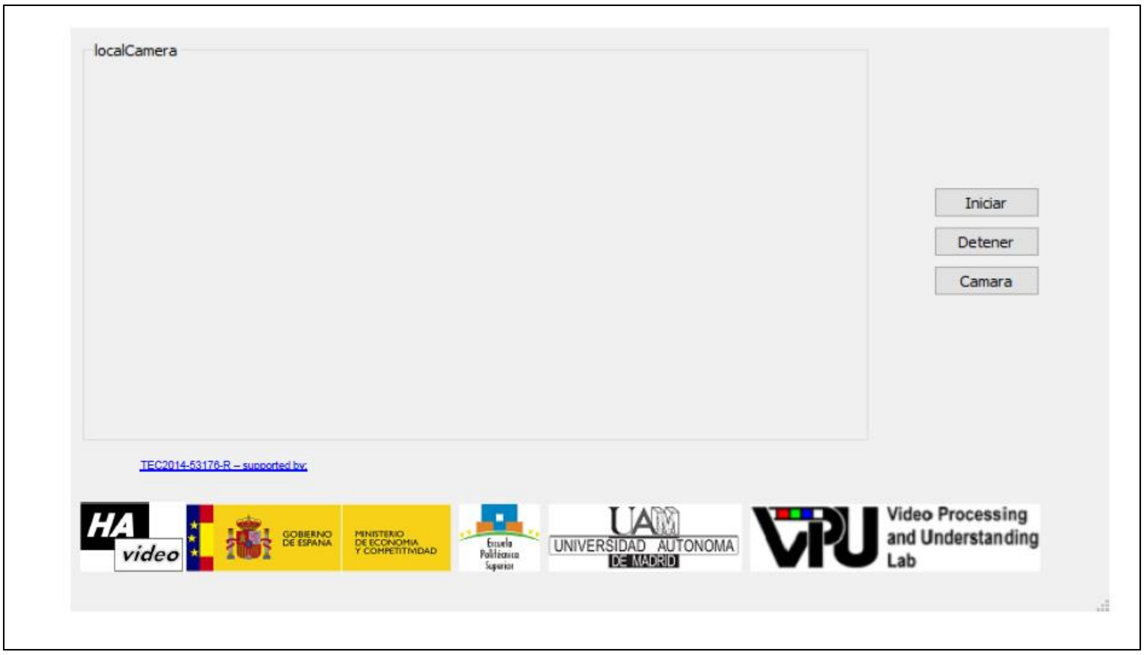

localCamera
Iniciar
Detener
Camara
TEC2014-53176-R – supported by:
HA video
GOBIERNO DE ESPAÑA
MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
Escuela Politécnica Superior
UAM UNIVERSIDAD AUTONOMA DE MADRID
Video Processing and Understanding Lab

Figure 4. SAPKLTF demonstrator interface.

New features added to the demonstrator:

- Frame size change (for speed-up purposes)

A scale variable has been added that indicates the scaling that will be performed to the frames before processing them. It is also necessary to scale the ground truth annotations in the same way and rescale them in the output to maintain the original size display.

- Support to orientation changes (rotated bounding-box)

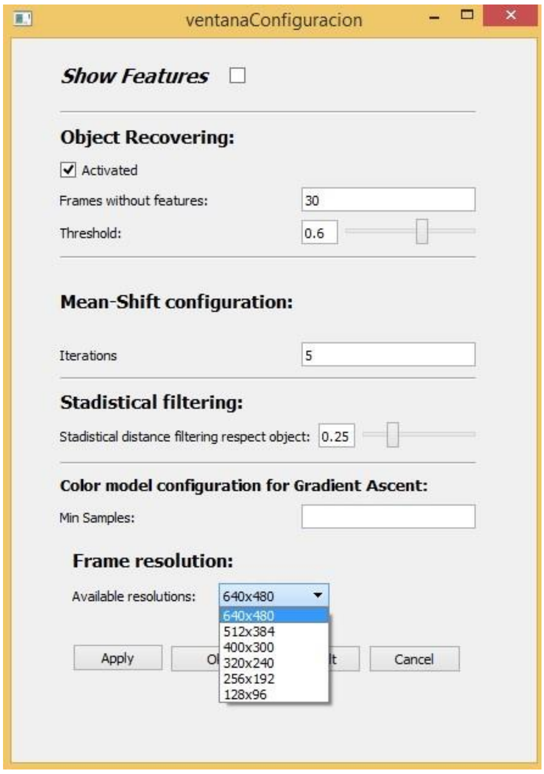
The changes have been made on the videoMain function in which the new points are calculated from the angle. Starting from the previously implemented bounding-box, its corners are rotated the corresponding angle to obtain the new points of the rotated bounding box.

- Webcam resolution change through the interface

The resolution change of the frame has been developed in the function that processes the frames. If there is a selected object, the resolution is processed, otherwise, an array of the frame size with the new scale is created.

If the function of updating the display detects that the user wants to change the resolution through the interface, it turns off the camera with the old resolution, and connect a new camera with the new resolution.

In the same way, a ComboBox has been added in the configuration window with the available resolutions. The result is presented below.



ventanaConfiguracion

Show Features ☐

Object Recovering:

☑ Activated

Frames without features: 30

Threshold: 0.6

Mean-Shift configuration:

Iterations 5

Stadistical filtering:

Stadistical distance filtering respect object: 0.25

Color model configuration for Gradient Ascent:

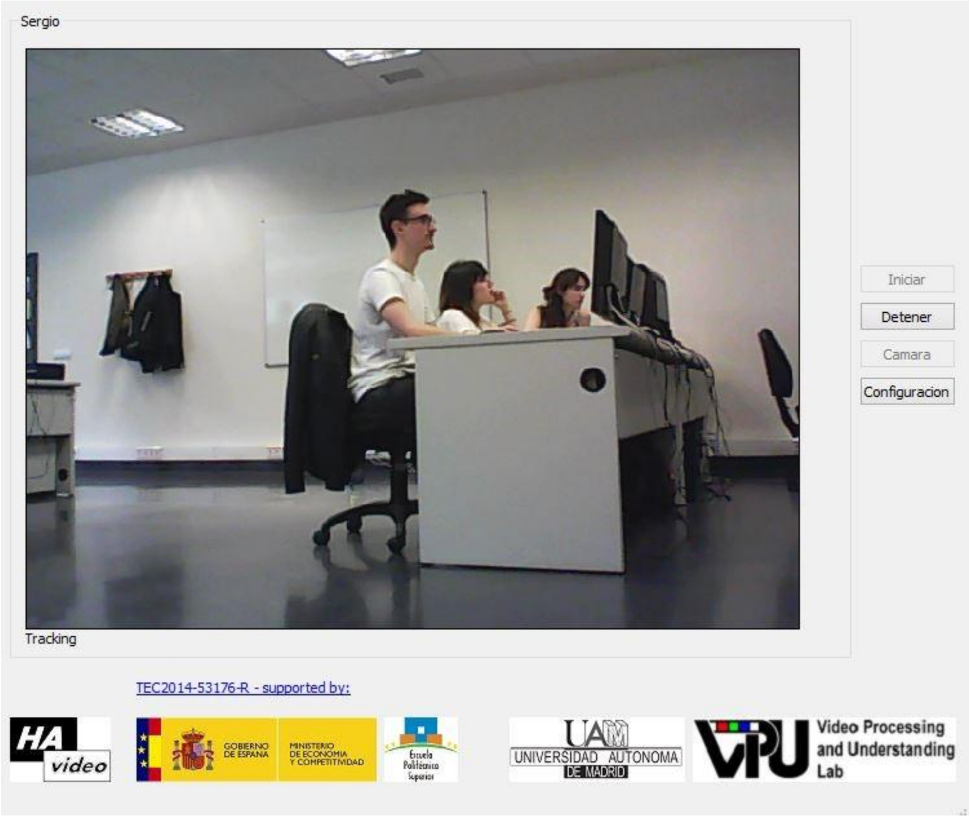
Min Samples:

Frame resolution:

Available resolutions: 640x480

640x480
512x384
400x300
320x240
256x192
128x96

Figure 5. Graphical user interface or demonstrator interface.


Sergio
Iniciar
Detener
Camara
Configuracion
Tracking
TEC2014-53176-R - supported by:
HA video
GOBIERNO DE ESPAÑA
MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
Escuela Politécnica Superior
UAM UNIVERSIDAD AUTONOMA DE MADRID
Video Processing and Understanding Lab

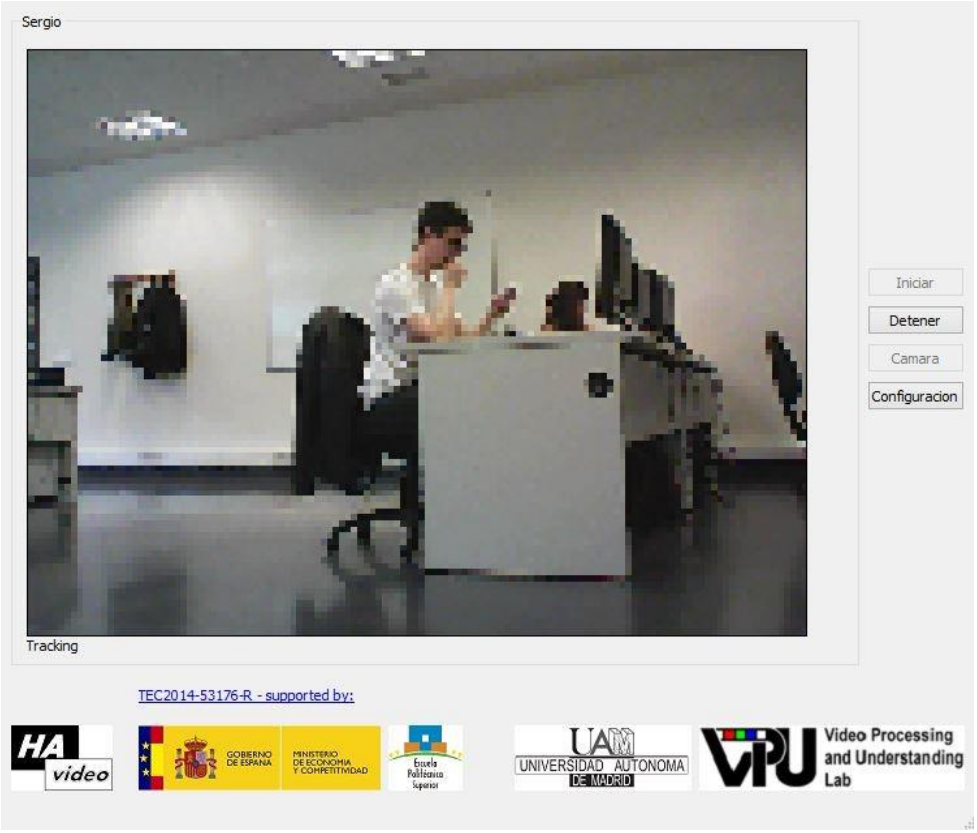

Sergio
Iniciar
Detener
Camara
Configuracion
Tracking
TEC2014-53176-R - supported by:
HA video
GOBIERNO DE ESPAÑA
MINISTERIO DE ECONOMÍA Y COMPETITIVIDAD
Escuela Politécnica Superior
UAM UNIVERSIDAD AUTONOMA DE MADRID
Video Processing and Understanding Lab

Figure 6. Example of resolution changes of the webcam by graphic interface

- Initialization of histograms on non-vertical ground truth

The original code, when extracting the characteristics from the ground truth (initialize the object), considered a rectangle aligned with the axes of the frame. In this implementation this has been changed so that it is now done on the exact area marked by the ground truth obtained.

Two new functions have been implemented. These two functions are responsible for obtaining the histograms of the points contained in the area, bounded by the polygon (which will match the coordinates obtained for the ground truth), foreground, and the area around it, background, which will be used to perform the background correction (CBWH). In the following images the corresponding mask is applied to obtain the desired pixels.

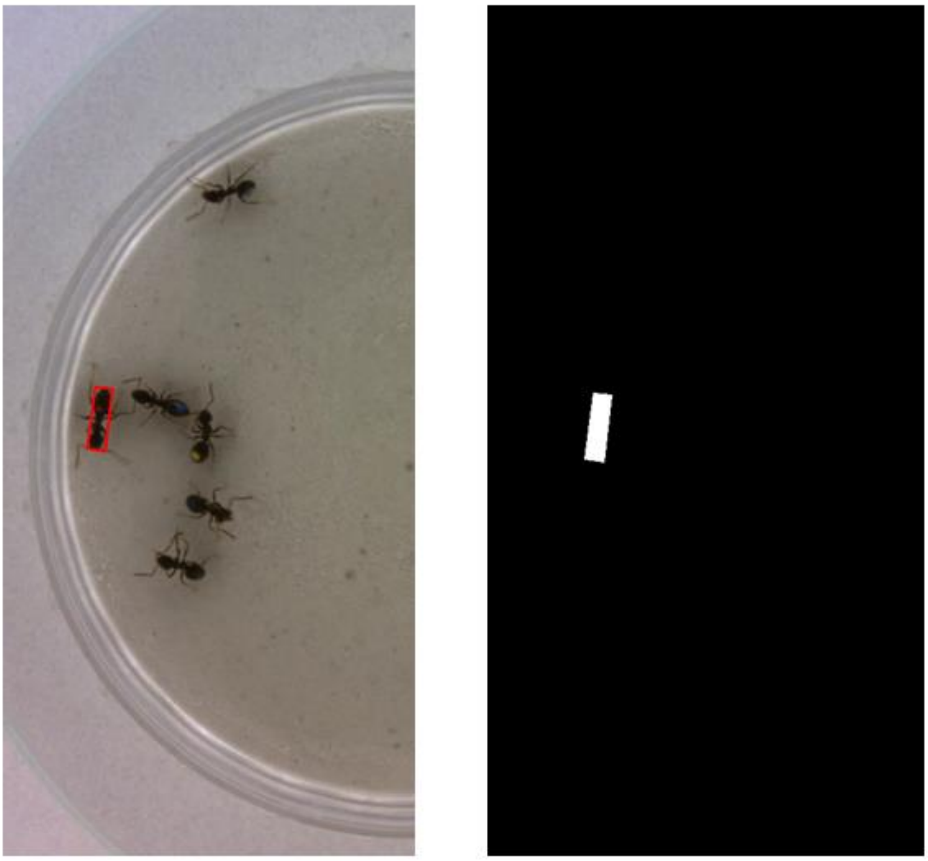


Figure 7. Ground truth (right) and obtained mask (left)

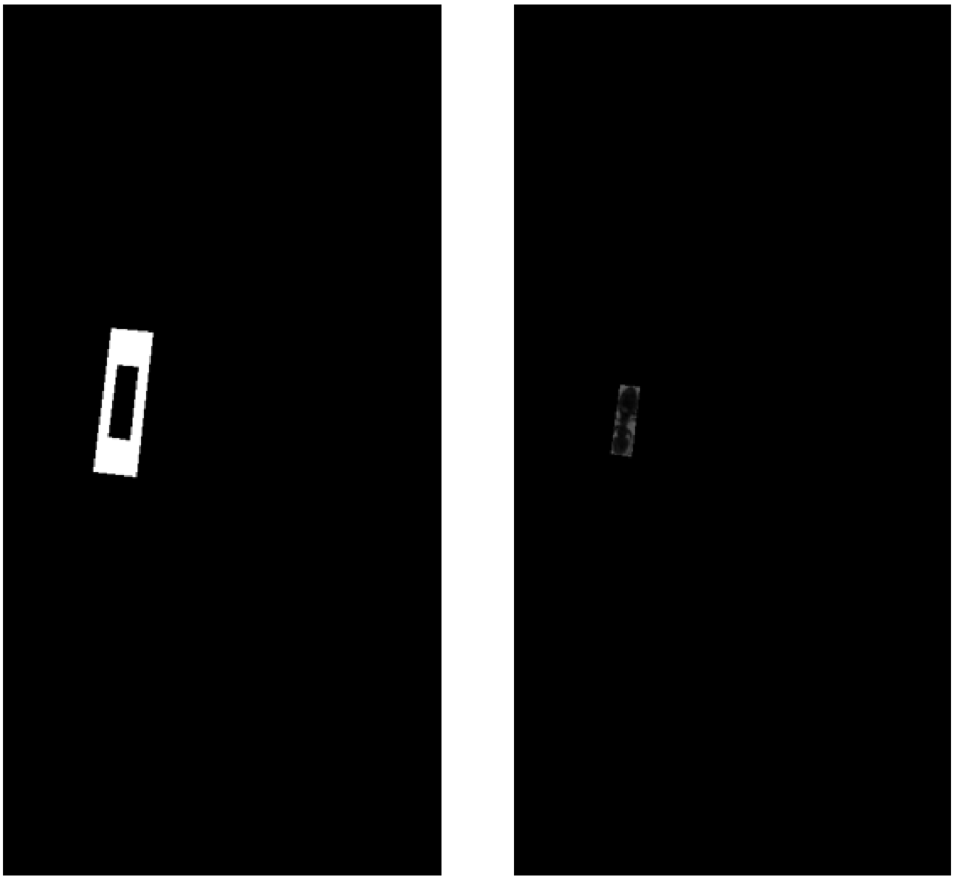


Figure 8. Mask considered to obtain the background (left) and region obtained for the object (right)

2.4. Pedestrian Density estimation in multi-camera scenarios.

This method starts from the solution for multi-camera pedestrian detector with semantic constraining described in section **¡Error! No se encuentra el origen de la referencia.** and provides new functionalities to estimate and demonstrate pedestrian density estimation in multi-camera scenarios.

First, unique scene-wise pedestrian detections are generated. To this aim, per camera detection are projected onto a common ground plane by means of homography transformations. Based on semantic cues from the scenario, the ground-plane is role-annotated creating thereby, an automatic focus area which allows to filter-out false pedestrian detections (commonly known as *ghosts*). Besides, a novel scheme to fuse per-camera pedestrian detections is proposed. Using graph theory, connected components are established between projected per-camera detections on the ground-plane, generating unique and global pedestrian detections.

Second, the application has been enhanced so it can operate on both video files and on video streams from Pan-Tilt-Zoom (PTZ) cameras. *OpenCV* capabilities have been used to load a video stream from an IP camera into the application. *OpenCV VideoCapture* class permits to load URL streams in the following format: *IP_PROTOCOL://user:password@IP/XXX*. The IP (Internet Protocol) PTZ cameras (Sony IPELA SNC-RZ60P) available in the Escuela Politécnica Superior are accessible via *HTTP*, providing a *MJPEG*-compressed video stream. Using the *OpenCV VideoCapture.open()* method, this stream is accessible via: *http://<user>:<password>@192.168.11.21/mjpeg*. The *VideoCapture* class is driven by a stand-alone buffer of fixed size which in case of overflow discards frames to flush computer's memory.

Third, the application has been improved and now deep learning architectures and modules can be loaded. Now there is the possibility of including both detectors and semantic segmentation algorithms by the use of the *Deep Neural Network (DNN)* module by OpenCV. In the first place, a newer version of Faster-RCNN object detection algorithm has been integrated in the application via the *OpenCV FasterRCNNPeopleDetection()* method. In the baseline application, object detection by Fast-RCNN was pre-computed offline and then loaded into the application. Using the *OpenCV* DNN module, Faster-RCNN is integrated to run online within the application. Integration has been done so Faster-RCNN can be used under both VGG-16 and ZF architectures with weights pre-trained on the ImageNet dataset. Computational times when using DNN module in an Intel® Core™ i3-4150 CPU @ 3.50GHz × 4 computer are around 25 seconds per frame.

Pedestrian density estimation is driven by a novel *density_estimation* method which operates on the fused detections on an image representation of the ground plane. The global detections are considered punctual and represented by active pixels on this ground plane image. A 3-dimensional Gaussian kernel with temporal scale observation T and spatial scale observation σ is used to convolve successive ground plane images, both scale parameters can be dynamically changed during execution.

A demonstrator application is developed to illustrate these functionalities. This application is supported by two versions of a GUI: the **developer** (see **Figure 9.** Developer version of the Pedestrian Density Estimation application interface: layout (left) and example (right).

2.5. Application for potential distractors detection in video object tracking.

This application aims to demonstrate the capabilities of the methods created along the HA-Video project to detect, given a target object, a scored-set of potential distractors in the same video sequence. To this aim, some of the methods have been adapted to this scenario, and a new application with associated interface is developed.

Given a target, the protocol is as follows. First, the target is described, and a configurable searching area is defined around it. Potential objects on this area can be defined by an *objectness* analysis. The application also allows to bypass this stage, searching on a sub-area partition of the whole searching area under an equal-scale premise. In both cases, objects and subareas are described by the same procedure as the target. Distractor scores are obtained by comparing these descriptions with the target one. In order to describe the target, objects and subareas, convolutional neural networks are used.

Two methods have been integrated in the application. In the first method, a ResNet50 architecture pre-trained with the ImageNet dataset is used to characterize the target and a searching area around it. In particular, the 22nd layer of the network is used, yielding a 512-dimensional characterization vector for each pixel. Using all the target pixels as a template, we rely on normalized cross-correlation on the searching area to locate the centers of potential distractors. This scoring scheme shares philosophy with recent Siamese-network methods.

The second method relies on a preliminary detection of potential objects in the searching area. To this aim, the EdgeBoxes algorithm is used. The target and each detected object are then characterized by the last layer of an AlexNet architecture also trained with the ImageNet dataset. Up to four measures are available for description comparison: cosine, Euclidean, earth's-mover and chi-square *distances* are included in the application. Results of this method can be visualized in **Figure 10**.

Both methods are accessible via Graphical Interface Unit (GUI) see **11**. Through the GUI, a potential user can also select the input data and parameters of both methods. In particular, the current version of the application allows to analyze all the videos in the VOT2016 challenge and to configure the following parameters:

- Number of distractors: defines the maximum number of distractors to detect. These are sorted according to their resemblance to the target.
- Threshold: defines the maximum distance between the target and a potential distractor to consider this last a plausible one.
- Distractor radio: allows to define the searching area around a target. The size of this area is defined proportional to the target's size.

- Overlap: Parameterizes a Non-Maximum Suppression method to remove potential distractors before their description.

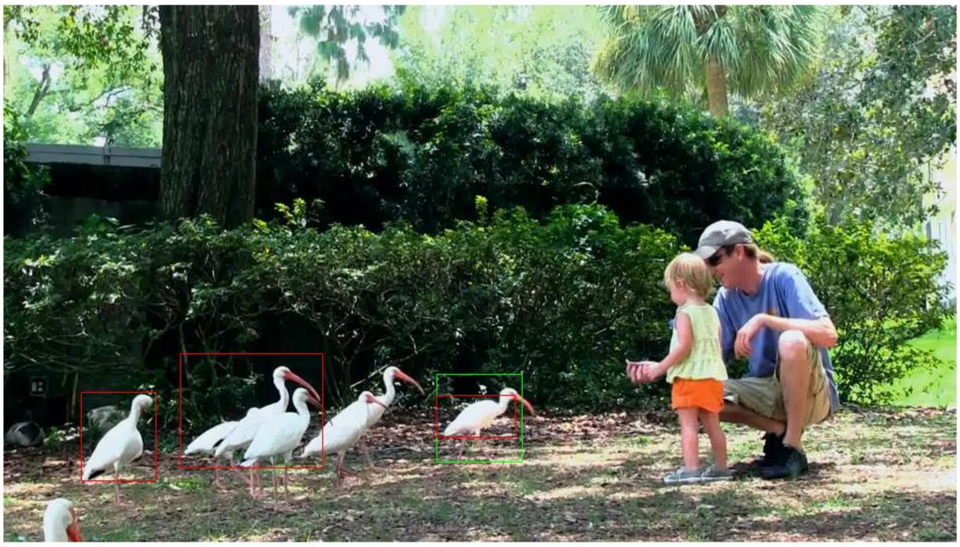


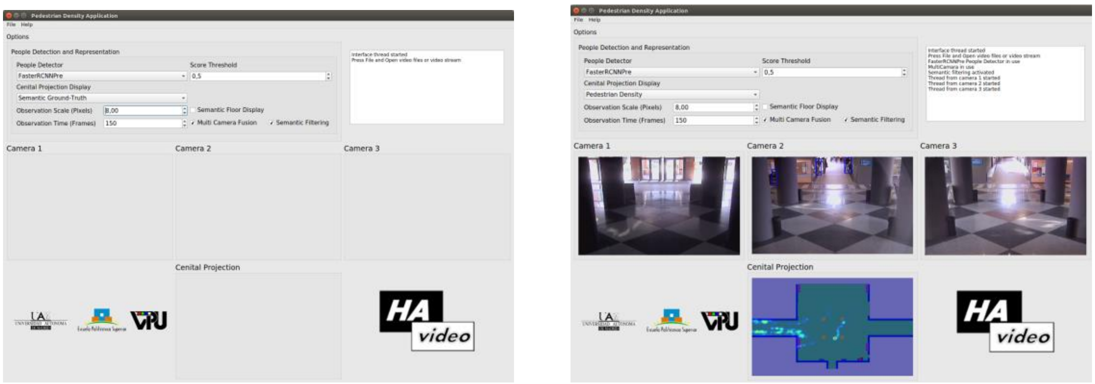
Figure 10. First three potential distractors—in red—given a target—in green—. Here, the first distractor is detected inside the target and only included for illustrative reasons.

Figure 11. Layout of the distractor detection application.

2.6. Application for people detection evaluation

) and the **client** version. Both versions allow to demonstrate the method both on a pre-recorded video file and on an IP video stream.

The **developer** version permits to online change pedestrian detection methods and the parameters of the new pedestrian density estimation method (T and σ). Several pedestrian detections are integrated in the demonstrator: Histogram of Oriented Gradients (HOG), Deformable Parts Model (DPM), Aggregate Channel Features (ACF), Fast-RCNN and YOLO. The **client** version allows the user to run the program without any knowledge on the algorithms behind the application. Due to this reason, it only contains the display and the information area. The configuration area is disabled, and all the tunable parameters are default set.


Pedestrian Density Application
File
Help
Options
People Detection and Representation
Camera 1
Camera 2
Camera 3
Cenital Projection
HA video

People Detector: FasterRCNNPre
Score Threshold: 0,5
Cenital Projection Display: Semantic Ground-Truth
Observation Scale (Pixels): 8,00
Observation Time (Frames): 150
□ Semantic Floor Display
✓ Multi Camera Fusion
✓ Semantic Filtering

Interface thread started
Press File and Open video files or video stream

People Detector: FasterRCNNPre
Score Threshold: 0,5
Cenital Projection Display: Pedestrian Density
Observation Scale (Pixels): 8,00
Observation Time (Frames): 150
□ Semantic Floor Display
✓ Multi Camera Fusion
✓ Semantic Filtering

Interface thread started
Press File and Open video files or video stream
FasterRCNNPre People Detector in use
MultiCamara in use
Semantic filtering activated
Thread from camera 1 started
Thread from camera 2 started
Thread from camera 3 started

Figure 9. Developer version of the Pedestrian Density Estimation application interface: layout (left) and example (right).

2.7. Application for potential distractors detection in video object tracking.

This application aims to demonstrate the capabilities of the methods created along the HA-Video project to detect, given a target object, a scored-set of potential distractors in the same video sequence. To this aim, some of the methods have been adapted to this scenario, and a new application with associated interface is developed.

Given a target, the protocol is as follows. First, the target is described, and a configurable searching area is defined around it. Potential objects on this area can be defined by an *objectness* analysis. The application also allows to bypass this stage, searching on a sub-area partition of the whole searching area under an equal-scale premise. In both cases, objects and subareas are described by the same procedure as the target. Distractor scores are obtained by comparing these descriptions with the target one. In order to describe the target, objects and subareas, convolutional neural networks are used.

Two methods have been integrated in the application. In the first method, a ResNet50 architecture pre-trained with the ImageNet dataset is used to characterize the target and a searching area around it. In particular, the 22nd layer of the network is used, yielding a 512-dimensional characterization vector for each pixel. Using all the target pixels as a template, we rely on normalized cross-correlation on the searching area to locate the centers of potential distractors. This scoring scheme shares philosophy with recent Siamese-network methods.

The second method relies on a preliminary detection of potential objects in the searching area. To this aim, the EdgeBoxes algorithm is used. The target and each detected object are then characterized by the last layer of an AlexNet architecture also trained with the ImageNet dataset. Up to four measures are available for description comparison: cosine, Euclidean, earth's-mover and chi-square *distances* are included in the application. Results of this method can be visualized in **Figure 10**.

Both methods are accessible via Graphical Interface Unit (GUI) see **11**. Through the GUI, a potential user can also select the input data and parameters of both methods. In particular, the current version of the application allows to analyze all the videos in the VOT2016 challenge and to configure the following parameters:

- Number of distractors: defines the maximum number of distractors to detect. These are sorted according to their resemblance to the target.
- Threshold: defines the maximum distance between the target and a potential distractor to consider this last a plausible one.
- Distractor radio: allows to define the searching area around a target. The size of this area is defined proportional to the target's size.
- Overlap: Parameterizes a Non-Maximum Suppression method to remove potential distractors before their description.

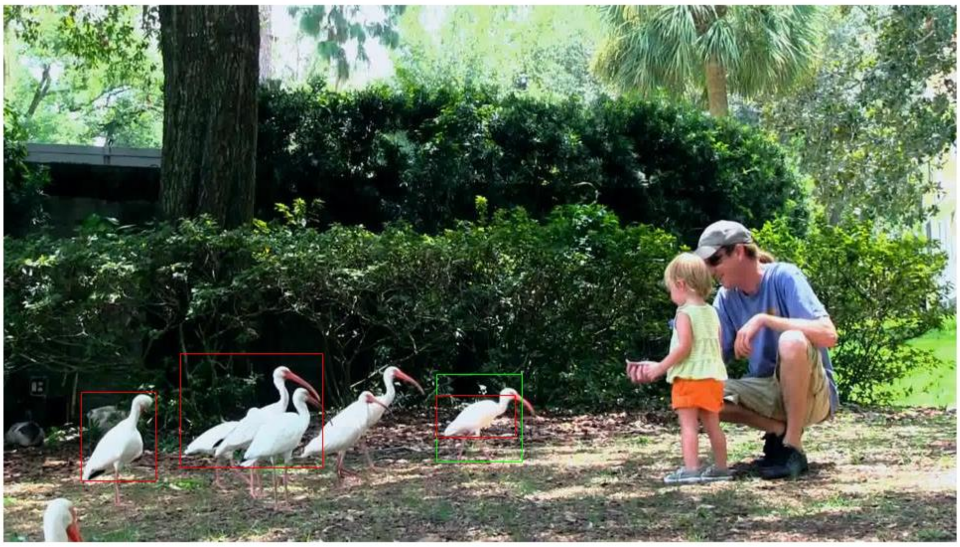
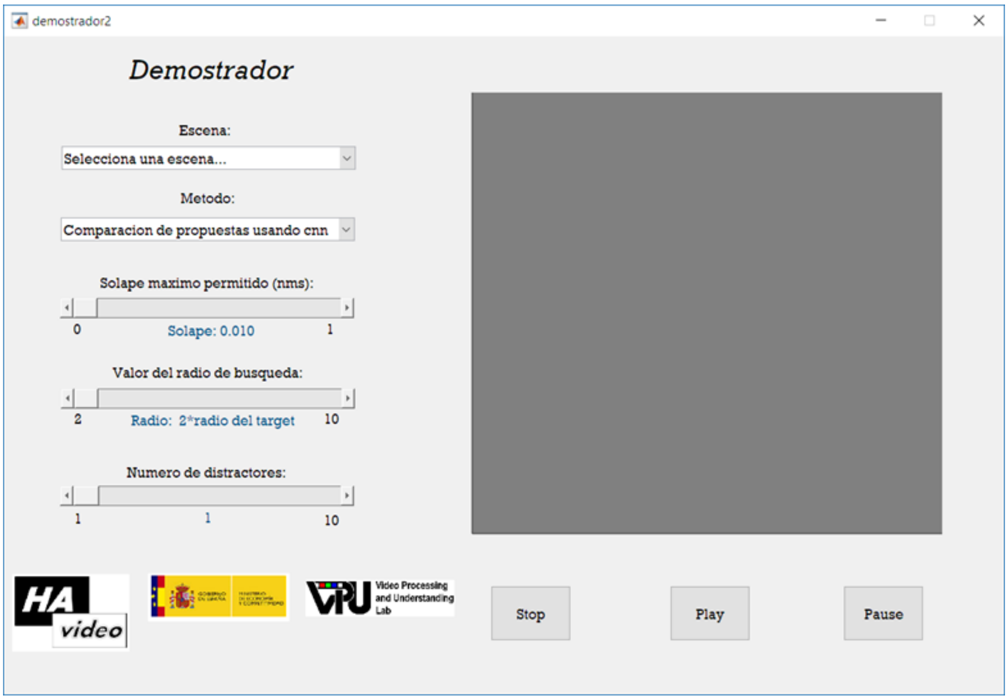


Figure 10. First three potential distractors—in red—given a target—in green—. Here, the first distractor is detected inside the target and only included for illustrative reasons.



demostrador2

Demostrador

Escena:

Selecciona una escena...

Metodo:

Comparacion de propuestas usando cnn

Solape maximo permitido (nms):

0 Solape: 0.010 1

Valor del radio de busqueda:

2 Radio: 2*radio del target 10

Numero de distractores:

1 1 10

Figure 11. Layout of the distractor detection application.

2.8. Application for people detection evaluation

People detection is a low-level task used in many computer vision and video processing applications. Examples include video surveillance (e.g. people density estimation, people counting, action recognition, anomaly detection and action recognition), smart environments (e.g. room monitoring and fall detection), and content retrieval (video annotation, event detection, object tracking). Typically, these applications need to start with the detection of the objects of interest (which are people in the previous scenarios) to then realize their purposes.

PDbm [23] benchmark proposed to the researchers send by email their results in order to obtain the evaluation. In this way, they couldn't know how their algorithm works until they made the final submit. The new approach provides a standalone application which allows users to evaluate their algorithms from their own computer, and compare the output with the results of the detectors published on the web site from the state of the art. For this task, the files which the application is going to need are the documents for the sequences with all the bounding boxes detected in the specified format. Another important feature of this tool is that it allows to update the results of the algorithm to the platform making a final submission, that is going to contain the files with the bounding boxes detected, the researchers information and the output in terms of average precision (AP). For this reason, we will maintain and update a rank list of the most accurate people detection algorithms for years to come.

In order to make the evaluation, the dataset selected has been extracted from the Change Detection Benchmark [24] dataset 2012/2014. The sequences have been selected to easily compare any people detection approach from the state of the art. They contain a range of challenges present in the real world (e.g. baseline, dynamic background camera jitter) and include accurate ground truth for people detection.

All the evaluation code is integrated in the standalone application that provides the challenge. In order to reach a greater number of users, we have created a version for Windows and another version for Ubuntu.

The participants must install the executable following the instructions that are available in the same zip folder. The toolkit needs to know the path of the MATLAB Runtime. Therefore, for each operating system we have provided a version that includes an installer which downloads the MATLAB Runtime from web and installs it along with the deployed MATLAB application, and another version that includes an installer that already has the MATLAB Runtime installer. If the user already has MCR (MATLAB Compiler Runtime), it is important to check if the version is compatible with the application (R2017b). In case they are not, the Runtime included in the application zip must be installed. If the user does not have MATLAB, is not necessary to install it or the license, only the runtime must be installed.

This application evaluates as much video sequences as the participants want from the database, so first of all, we will select them as we can see in **Figure 12**. **Figure 13** shows one example of the application results over the sequence “tramstop”.

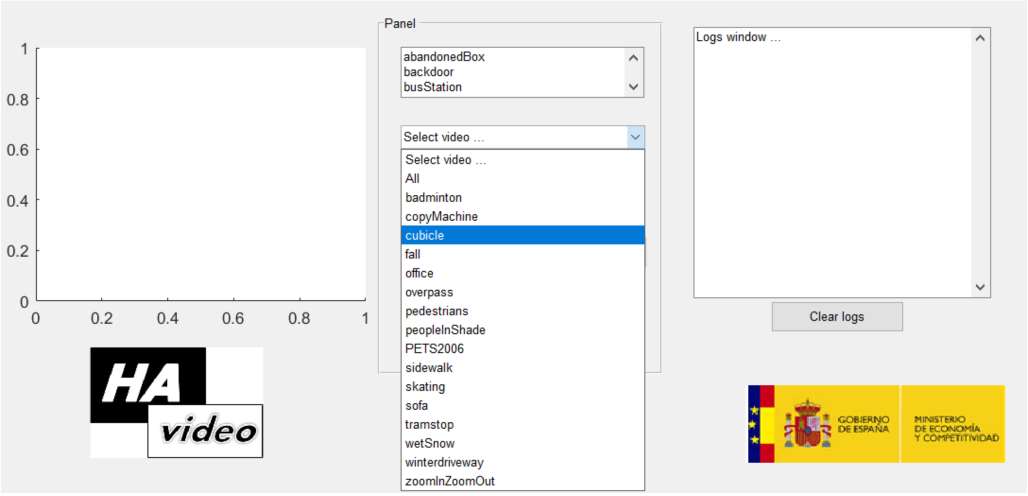
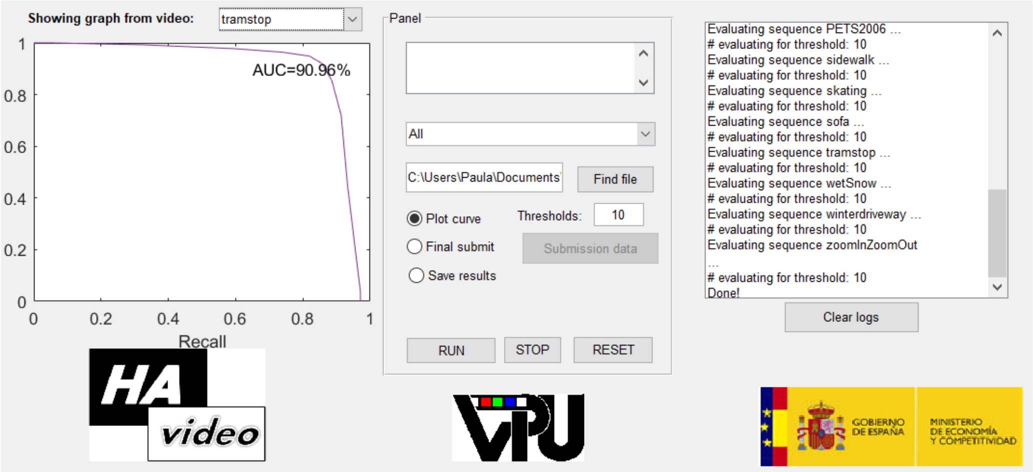


Figure 12. Application in the step of "select video sequences"


Showing graph from video: tramstop

AUC=90.96%

All

C:\Users\Paula\Documents

● Plot curve Thresholds: 10

○ Final submit

○ Save results

```
Evaluating sequence PETS2006 ...
# evaluating for threshold: 10
Evaluating sequence sidewalk ...
# evaluating for threshold: 10
Evaluating sequence skating ...
# evaluating for threshold: 10
Evaluating sequence sofa ...
# evaluating for threshold: 10
Evaluating sequence tramstop ...
# evaluating for threshold: 10
Evaluating sequence wetSnow ...
# evaluating for threshold: 10
Evaluating sequence winterdriveway ...
# evaluating for threshold: 10
Evaluating sequence zoomInZoomOut
...
# evaluating for threshold: 10
Done!
```

Figure 13. Application with the final results showed on the interface.

2.9. Multi-camera video surveillance system based on smartphones

Over the past years, video surveillance has become a prominent issue in order to satisfy such a basic need as security. Currently, complex hardware surveillance systems and object detection algorithms are being used to meet this need. However, the demand for more affordable, portable and versatile systems is continuously increasing. Therefore, the motivation for this project is to provide users with a more accessible video surveillance system by exploiting the assets of modern cell phones (Android smartphones), since they are available to most people nowadays.

This project continues the work presented in [25]. The proposed system uses Android smartphones as cameras, which are controlled remotely by the user with a client application installed in a computer. In addition, a server is implemented, in the same computer, in order to control and manage every communication between users and cameras.

Thereby, the main goal of this project is to build a surveillance system by which the user is able to display the images captured by the smartphones in a computer. For that purpose, the user will send orders from the client application, which will reach the cameras through the server, by a certain WiFi connection. Then, the cameras will start to capture images and send them back to the user. Aside from this main feature, the user can control the cameras by using other predefined commands such as: see and set certain camera parameters (resolution, FPS, Bluetooth status, WiFi status, etc), start and stop the image capturing process by the cameras and demand either the last frame from a camera (Get-Frame) or every frame captured for a certain amount of time (Get-Video).

In order to evaluate the system performance, diverse scenarios were tested, with different number of simultaneously connected cameras and image resolutions. Thus, measures of frames per second (FPS), CPU load, network saturation and use of RAM have been made. The results of these tests show some issues regarding the number of FPS received at the computer when dealing with high resolution images. Therefore, one of the goals of this project is to find the causes of this matter and try to implement a solution.

Therewith, it has been proven that the main bottlenecks of this system are due to the image processing capacity of the used cell phones and the network bandwidth. In addition, the number of FPS also decreased when dealing with multicamera scenarios, since the client application was only able to manage the frame request of one camera at a time. In order to enhance the system performance, this section has been improved by making the client application being able to manage the video request from several cameras simultaneously, which leads to much better results.

Furthermore, new system applications regarding object detection and tracking have been started. This first approach to include tracking in the system functionality is still in progress, so future work will include the development of this kind of features and their testing. Finally, more improvements regarding image processing and transmission could be made in order to obtain a more efficient system.

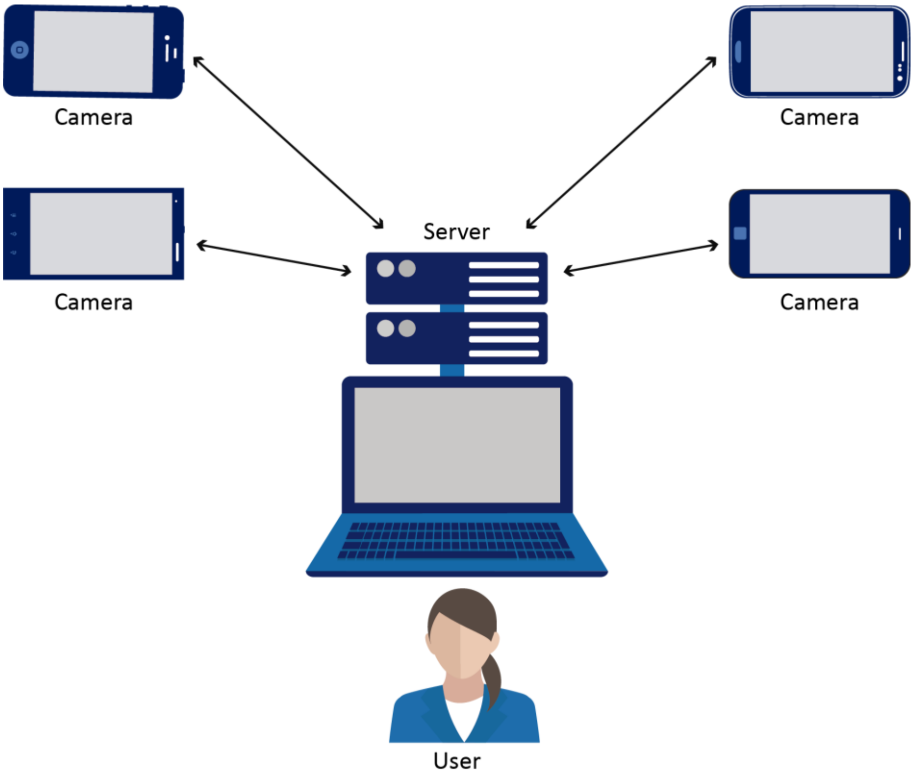


Figure 14. Visual example representing the different elements of the system.

2.10. Live streaming of classes and seminars using automatic video tracking

The idea of global communication, which nowadays is present in all the areas of our life, makes technology and the telecommunication systems indispensable. If we talk about education and how we could enhance it by integrating telecommunications system throughout the teaching and learning environments, we could think about connecting students and classes from Internet, using on-demand video streaming as a tool for multiple learning experiences.

Thus, the main motivation of this project is to automate the camera tracking of a teacher in a classroom to generate a portable system that could stream classes in real-time using on-demand video streaming, so that students who, for various reasons, couldn't physically attend the classroom could be part of it from their devices.

This project is a continuation of several previous works. On one hand, [21][26] which result in two algorithms that, based on the sequence of images captured by an IP fixed camera, track the teacher's position in real time the classroom and oriented in the direction an IP PTZ camera. In the other hand, [27], which achieve a fully functional broadcast system via live streaming for classes.

The previously algorithms implemented were designed for fixed systems, which could not be transported and, therefore, used in different facilities. Therefore, to give more possibilities and comfort, the main objective is to create a complete portable system.

As we can see in **Figure 15**, our portable system is composed of a cameras system, a router and a laptop.

- Cameras System: It uses two cameras; a Pan-tilt-zoom IP camera and a webcam, both with a broad range of vision, which will be connected to the router and the laptop, respectively.
- Laptop: It uses a laptop which supports live broadcast and the software that enables the functioning of the cameras system.
- Router: It supports the connection between the PTZ camera and the laptop, and the live broadcast.

The speaker will use the laptop to run the tracking algorithm code, first it will detect the position of the person who is speaking and track him, and next it would send the guidelines of movement to the PTZ camera. After that, the PTZ camera will send the video trough the router to the laptop, who will stream the video.

In the following picture we can see a model of the system and how it works.

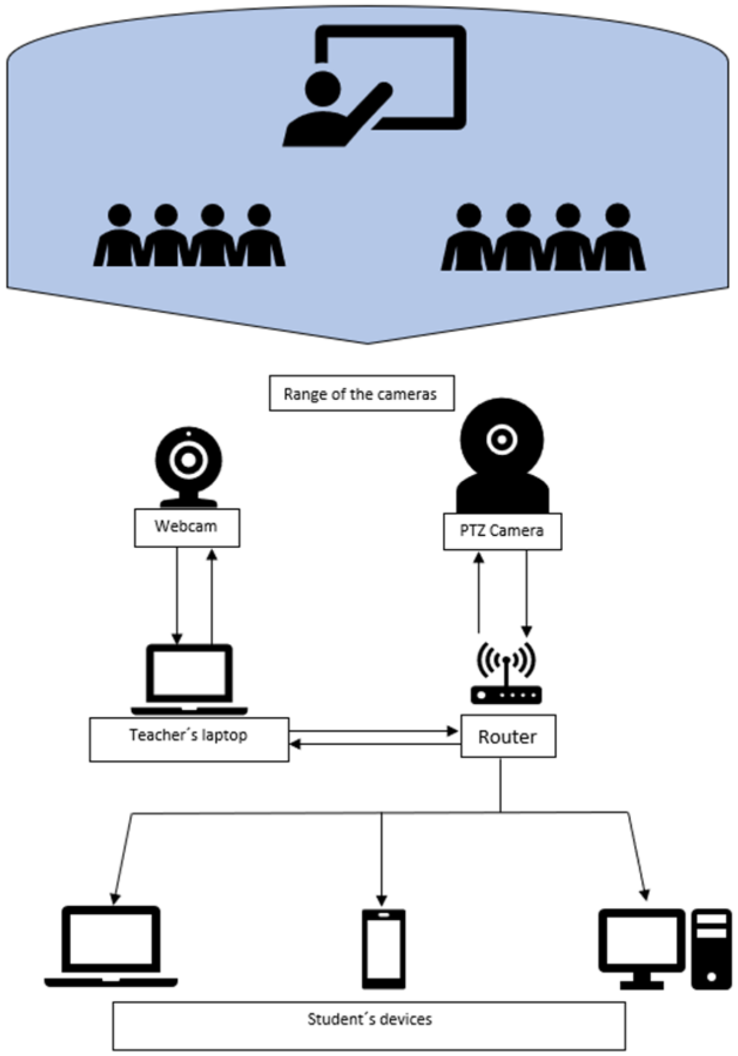


Figure 15. Visual example representing the different elements of the system.

We have two fundamental parts in the development of the system:

TRACKING- On the one hand we have the tracking of the speaker, that will be made by the PTZ camera that will move consistent with the instructions received from the webcam, based on the homography matrix. On the other hand, we have the webcam that achieve the tracking instructions of the lecturer due to a HOG first person detection and an algorithm about follow of distinctive points.

STREAMING- To do this part of the job we are based on the idea of using a streaming page as YouTube that haves a huge digital support. Taking advantage of this, we make a private streaming allowing only the people who is participant in the Moodle grade of the subject to see the streaming. This allows us to get more privacy and the option of get the participation of the students in the streaming with the Moodle blogs.

2.11. Multi-Camera Pedestrian Detection Benchmark Application

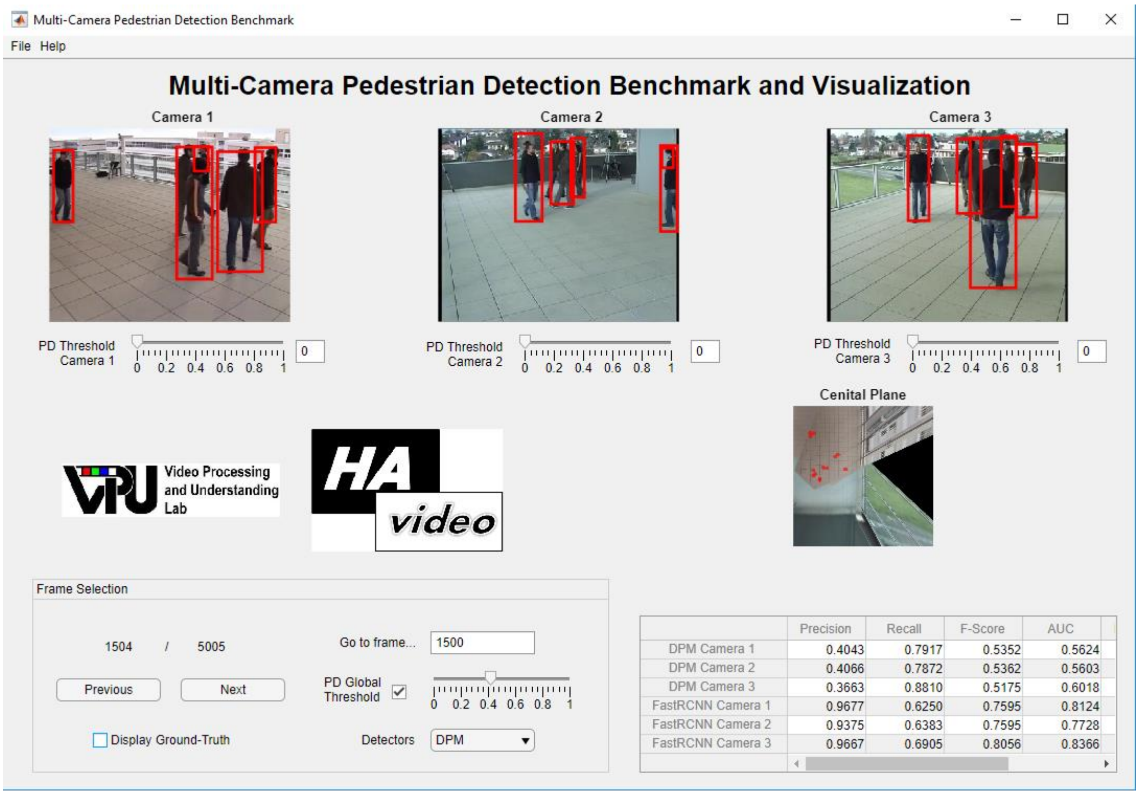
The multi-camera pedestrian detection benchmark application starts from the application described in section 2.2 and enhances it with functionalities to ease:

- The synchronous evaluation of several multi-camera pedestrian detectors.
- The quantitative evaluation of multi-camera pedestrian detectors by multiple state-of-the-arts indicators on a per-frame basis.
- The qualitative evaluation of multi-camera pedestrian detectors with respect to ground-truth data both on the image and the floor planes.

Functionalities. The application allows the users to quantitatively evaluate both state-of-the-art and new pedestrian detectors, providing per-frame and synchronized performance indicators for all the evaluated algorithms. Evaluation routines are included within the application. Specifically, the following multi-camera pedestrian detection performance indicators [28][29] have been included: Precision, Recall, F-Score, Area Under the Curve, N-MODA, N-MODP. The overall values of these indicators are provided for each camera and for each pedestrian detector (see bottom-right part of **Figure 16**). Furthermore, the application provides visualization tools to simplify and signify qualitative evaluation. Detection results—in the shape of bounding-boxes—are overlaid onto each camera frame together with ground-truth information. Moreover, global (camera-aggregated) detections are overlaid on the floor plane.

The application provides several visualization tools to navigate and parametrize performance, these are: move forward or backward in the video sequence, jump to a desired frame number, select operative the pedestrian detectors, enable ground-truth displaying, and configure global and per-camera pedestrian detection thresholds (see bottom-left part of **Figure 16**).

Performance indicators can be saved to a MATLAB file (.mat) and be used for later benchmarking with minimum revaluation.



Multi-Camera Pedestrian Detection Benchmark and Visualization

PD Threshold Camera 1: 0

PD Threshold Camera 2: 0

PD Threshold Camera 3: 0

Frame Selection

1504 / 5005

Go to frame... 1500

PD Global Threshold ☑

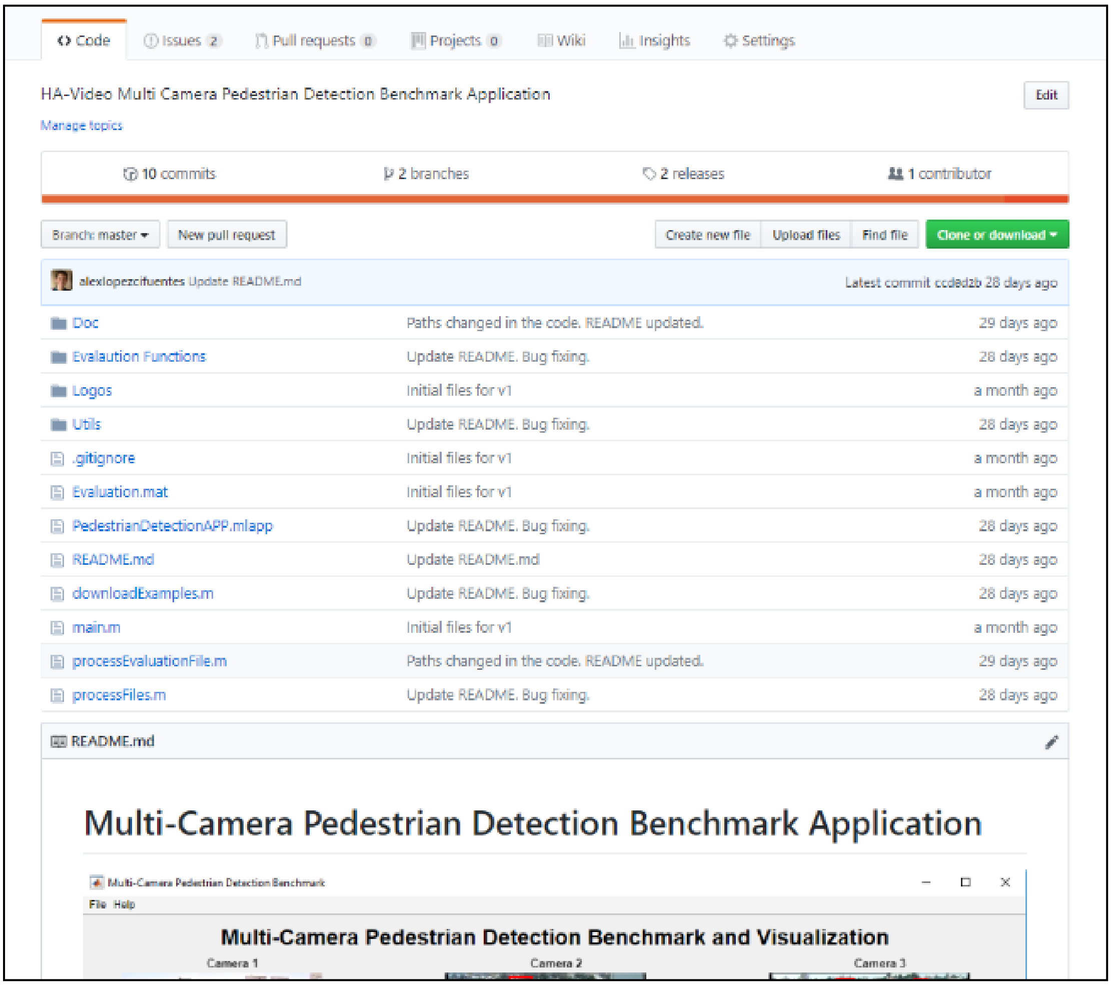
□ Display Ground-Truth

Detectors: DPM

	Precision	Recall	F-Score	AUC
DPM Camera 1	0.4043	0.7917	0.5352	0.5624
DPM Camera 2	0.4066	0.7872	0.5362	0.5603
DPM Camera 3	0.3663	0.8810	0.5175	0.6018
FastRCNN Camera 1	0.9677	0.6250	0.7595	0.8124
FastRCNN Camera 2	0.9375	0.6383	0.7595	0.7728
FastRCNN Camera 3	0.9667	0.6905	0.8056	0.8366

Figure 16. Multi-camera pedestrian detection benchmark application. The example depicts results for the DPM [31] and Faster R-CNN [32] pedestrian detectors on the EPFL Terrace dataset [30].

Use. The application requires—in pre-specified but well-stablished formats—the following information to operate: per-camera detections (bounding-boxes) for each detector, camera frames, ground-truth files and calibration parameters for each dataset. Currently, the default version of the application is setup for EPFL Terrace [30] dataset. The application is fully operative and has been uploaded along with the necessary documentation to a GitHub repository (currently private) for its future use (see **Figure 17**).



HA-Video Multi Camera Pedestrian Detection Benchmark Application

alexlopezcifuentes Update README.md — Latest commit ccdedzb 28 days ago

Doc	Paths changed in the code. README updated.	29 days ago
Evalaution Functions	Update README. Bug fixing.	28 days ago
Logos	Initial files for v1	a month ago
Utils	Update README. Bug fixing.	28 days ago
.gitignore	Initial files for v1	a month ago
Evaluation.mat	Initial files for v1	a month ago
PedestrianDetectionAPP.mlapp	Update README. Bug fixing.	28 days ago
README.md	Update README.md	28 days ago
downloadExamples.m	Update README. Bug fixing.	28 days ago
main.m	Initial files for v1	a month ago
processEvaluationFile.m	Paths changed in the code. README updated.	29 days ago
processFiles.m	Update README. Bug fixing.	28 days ago

Multi-Camera Pedestrian Detection Benchmark Application

Multi-Camera Pedestrian Detection Benchmark
File Help
Multi-Camera Pedestrian Detection Benchmark and Visualization
Camera 1
Camera 2
Camera 3

Figure 17. GitHub repository containing the fully operative application code and documentation for future use.

2.12. Visualizer and controller for multi-camera video surveillance simulator

This application starts from the work done in WP1 for multi-camera system simulators (MSS simulator).

We have created an an interface that shows a multicamera scenario created by the MSS simulator in which you can test and compare artificial vision algorithms. WE have used MFC C ++ technology and OpenCV library for Computer vision.

The result is an interface where a videowall with different cameras is shown. In the same interface, different options for the use of cameras by buttons are included such as rotation left/right, movement up/down and zoom in/out.

The following figure shows an example of the interface

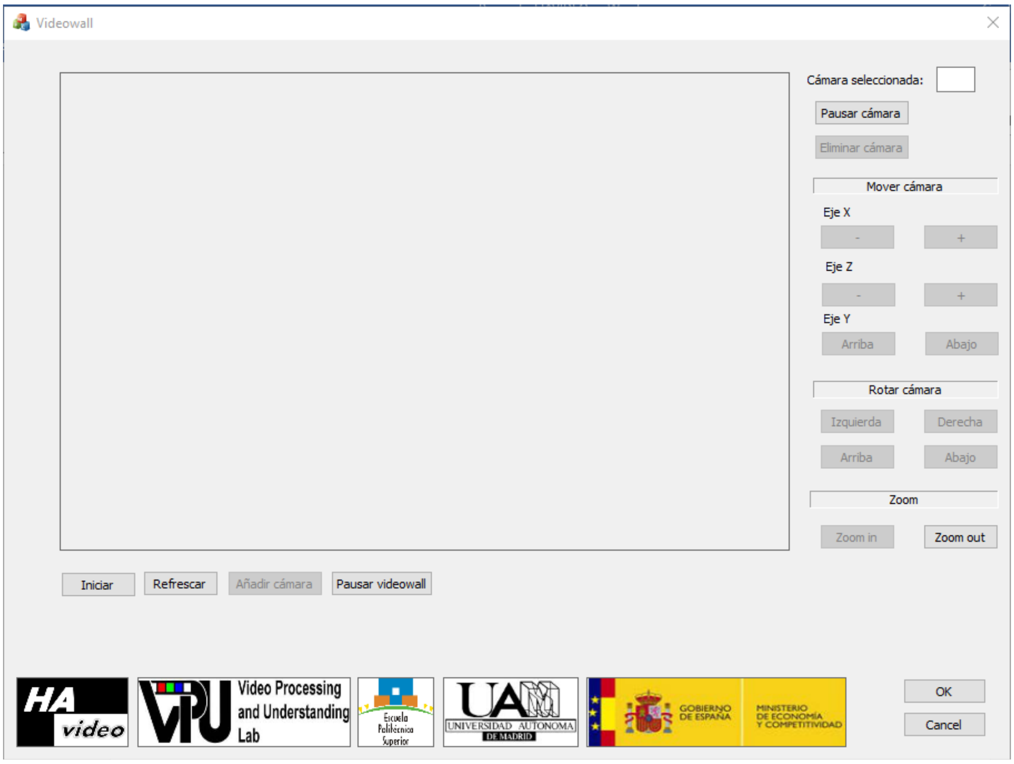
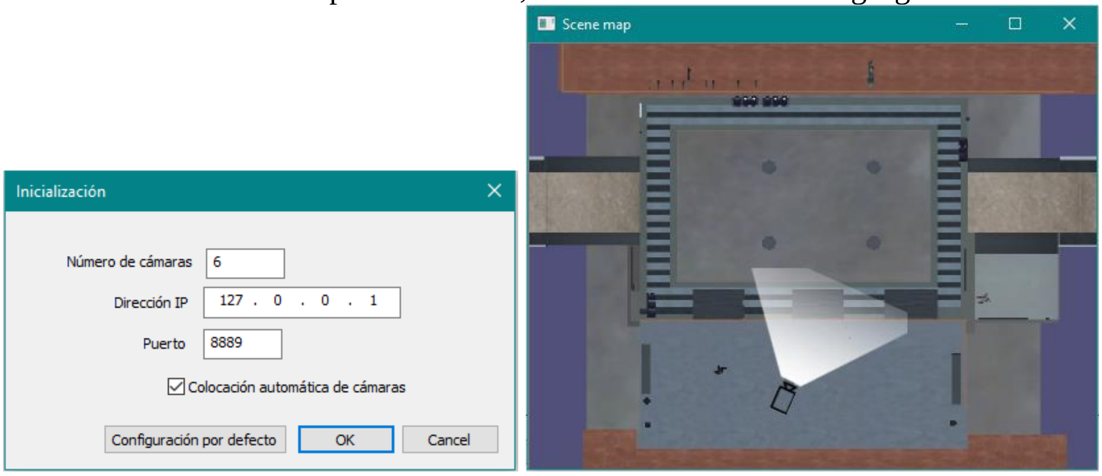


Figure 18. Visual interface for the MSS simulator

Moreover, additional interfaces have been created where you can setup the camera IP address and the camera spatial location., as shown in the following figure:



Número de cámaras 6

Dirección IP 127 . 0 . 0 . 1

Puerto 8889

☑ Colocación automática de cámaras

Figure 19. Additional visual interfaces

Finally, we include an example of the application running

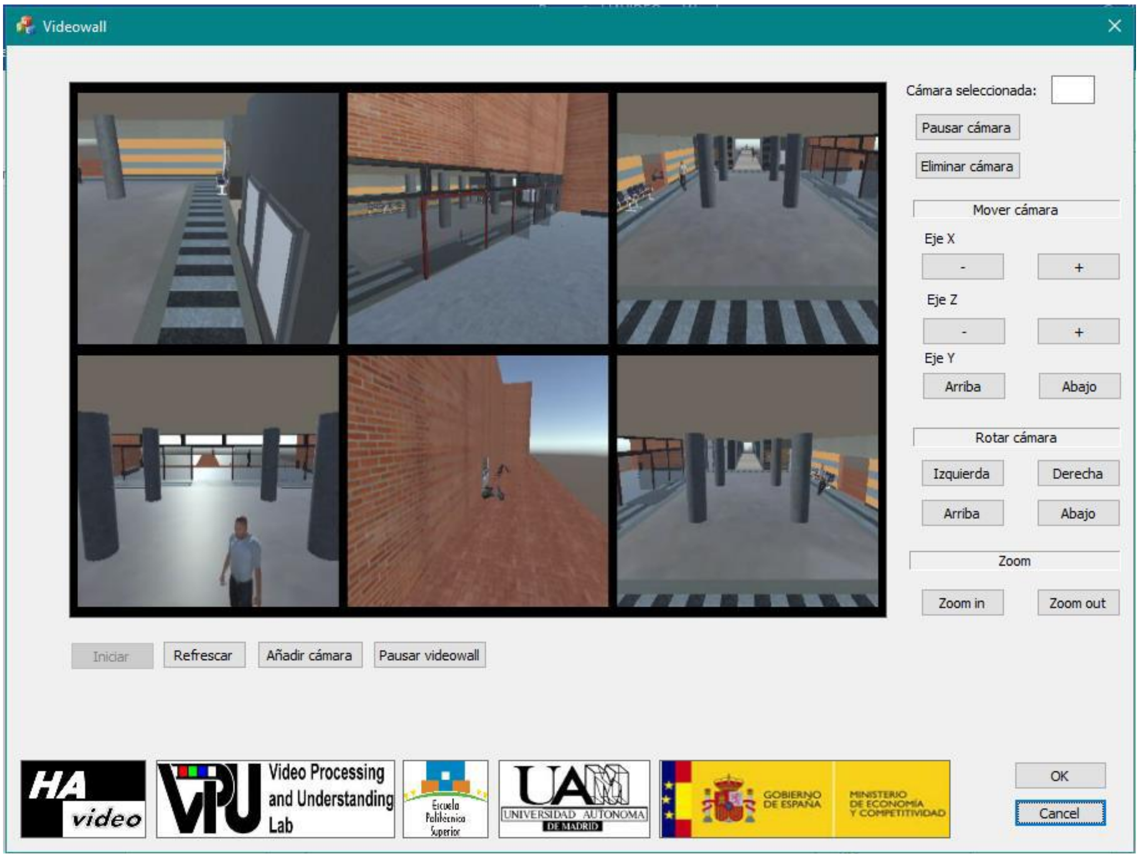


Figure 20. Running example of the visual interface for the MSS simulator

3. Conclusions

This deliverable describes the work related with the task T.4.2: Use Cases and Demonstrators. Following the guidelines for the development of applications and demonstrators in relation with the project. This document includes the description of the different evaluation methods, applications, and demonstrators developed during the last period of this project. 2.1 proposes a complete abandoned object detection (AOD) system demonstrator. 2.2 presents a multi-camera pedestrian detector with semantic constraining demonstrator. 2.3 describes a long-term tracking with target re-identification demonstrator, which is used in 2.10 for a complete streaming system for classes and seminars, and is complemented with 2.7 which presents an application for potential distractors detection. 2.4 describes the GUI developed for pedestrian density estimation in multi-camera scenarios. 2.9 extends a system for Multi-camera video surveillance which consider smartphones as autonomous cameras able to process images and video. Finally, two evaluation methods are presented in 2.8 and 2.11.

4. References

[1] Z. Zivkovic, “Improved adaptive gaussian mixture model for background subtraction,” in 17th International Conference on Pattern Recognition (ICPR 2004), vol. 2, pp. 28–31, 2004.

[2] Z. Zivkovic and F. Van Der Heijden, “Efficient adaptive density estimation per image pixel for the task of background subtraction,” Pattern recognition letters, vol. 27, no. 7, pp. 773–780, 2006

[3] S. Y. Elhabian, K. M. El-Sayed, S. H. Ahmed, Moving object detection in spatial domain using background removal techniques-state-of-art. Recent patents on computer science, 1(1), pp. 32-54, 2008.

[4] D. D. Bloisi, A. Grillo, A. Pennisi, L. Iocchi, and C. Passaretti, “Multi-modal background model initialization,” in International Conference on Image Analysis and Processing, pp. 485–492, Springer, 2015

[5] P.-L. St-Charles and G.-A. Bilodeau, “Improving background subtraction using local binary similarity patterns,” in IEEE Winter Conference on Applications of Computer Vision (WACV 2014), pp. 509–515, 2014.

[6] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin, “A self-adjusting approach to change detection based on background word consensus,” in IEEE Winter Conference on Applications of Computer Vision, pp. 990–997, 2015

[7] P. L. St-Charles, G. A. Bilodeau, R. Bergevin, (). Flexible background subtraction with self-balanced local sensitivity. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops pp. 408-413, 2014.

[8] S. Guler, J. A. Silverstein, I. H. Pushee. “Stationary objects in multiple object tracking. In Advanced Video and Signal Based Surveillance”, pp. 248-253, 2007.

[9] H.-H. Liao, J.-Y. Chang, and L.-G. Chen, “A localized approach to abandoned luggage detection with foreground-mask sampling,” in IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance, AVSS 2008., pp. 132–139, 2008

[10] D. Ortego and J. C. SanMiguel, “Stationary foreground detection for video- surveillance based on foreground and motion history images,” in IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2013, pp. 75–80, 2013

[11] F. Porikli, Y. Ivanov, T. Haga. “Robust abandoned object detection using dual foregrounds”. EURASIP Journal on Advances in Signal Processing, 2008

[12] C. Cuevas, R. Martinez, D. Berjón, N. Garcia. “Detection of stationary foreground objects using multiple nonparametric background-foreground models on a finite state machine”. IEEE Transactions on Image Processing, 26(3), pp. 1127-1142, 2017

[13] N. Dalal, B. Triggs, “Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition”, 1, pp. 886-893, 2005

[14] P. Viola, M. Jones, “Rapid object detection using a boosted cascade of simple features”. In Computer Vision and Pattern Recognition, Vol. 1, 2001.

[15] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan. "Object detection with discriminatively trained part-based models". IEEE transactions on pattern analysis and machine intelligence, 32(9), pp. 1627-1645, 2010.

[16] P. Dollár, R. Appel, S. Belongie, P. Perona. "Fast feature pyramids for object detection". IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(8), pp. 532-1545, 2014

[17] J. C. San Miguel, J. M. Martínez. "Robust unattended and stolen object detection by fusing simple algorithms". In Advanced Video and Signal Based Surveillance, pp. 18-25, 2008.

[18] J. C. Sanmiguel, L. Caro, J.M. Martínez. "Pixel-based colour contrast for abandoned and stolen object discrimination in video surveillance. Electronics letters, 48(2), pp. 86-87, 2012.

[19] Abandoned Object Detection in Long-Term Video-Surveillance, Elena Luna García, (advisor: Juan Carlos San Miguel Avedillo), Trabajo Fin de Máster (Master Thesis), Master en Investigación e Innovación en TIC – Programa Internacional de Múltiple Titulación IPCV (Image Processing and Computer Vision), Univ. Autónoma de Madrid, 2017.

[20] Online Contextual Updating in Multi-Camera Scenarios, Alejandro López Cifuentes, (advisor: Marcos Escudero Viñolo), Trabajo Fin de Máster (Master Thesis), Master en Investigación e Innovación en TIC – Programa Internacional de Múltiple Titulación IPCV (Image Processing and Computer Vision), Univ. Autónoma de Madrid, 2017.

[21] Long-Term Tracking with Target Re-Identification, Erik Velasco Salido, (advisor: José M. Martínez), Trabajo Fin de Máster (Master Thesis), Master en Investigación e Innovación en TIC – Programa Internacional de Múltiple Titulación IPCV (Image Processing and Computer Vision), Univ. Autónoma de Madrid, 2017.

[22] J. Sanjuan, "Seguimiento de objetos en tiempo real," F. Lahoz Seguido, Trabajo fin de grado, Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación, Universidad Autónoma de Madrid, 2014.

[23] A García-Martín, B Alcedo, and J.M. Martínez, "PDbm: people detection benchmark repository", Electronics Letters 51(7), pp. 559-560, 2015.

[24] N. Goyette, P.M. Jodoin, F. Porikli, J. Konrad, P. Ishwar, "Changedetection.net: A new change detection benchmark dataset.", in Computer Vision and Pattern Recognition Workshops, pp. 1-8, 2012.

[25] "Diseño de redes de cámaras inteligentes utilizando Smartphones" F. Lahoz Seguido, Trabajo fin de grado, Grado en Ingeniería Informática, Universidad Autónoma de Madrid, 2017.

[26] "Automatización de funciones en el seguimiento del profesor para la emisión de clases presenciales". Alberto Palero Almazán, Trabajo fin de Máster, Master en Ingeniería de Telecomunicación, Universidad Autónoma de Madrid, 2016.

[27] "Herramientas de apoyo a la emisión de clases presenciales". Álvaro J. Rubio Redondo, Proyecto final de carrera, Ingeniería de Telecomunicación, Universidad Autónoma de Madrid, 2015.

[28] P. Dollár, C. Wojek, B. Schiele, P. Perona, "Pedestrian detection: A benchmark". In Computer Vision and Pattern Recognition, pp. 304-311, 2009.

[29] R. Stiefelhagen, K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, P. Soundararajan, "The CLEAR 2006 evaluation. In International evaluation workshop on classification of events, activities and relationships", pp. 1-44. Springer, Berlin, Heidelberg, 2006.

[30] EPFL Terrace Dataset. https://cvlab.epfl.ch/data/data-pom-index-php/

[31] P. Felzenszwalb, D. McAllester, D. Ramanan. "A discriminatively trained, multiscale, deformable part model". In Computer Vision and Pattern Recognition, pp. 1-8. 2008.

[32] S. Ren, K. He, R. Girshick, J. Sun. "Faster R-CNN: Towards real-time object detection with region proposal networks". In Advances in neural information processing systems pp. 91-99. 2015.