MULTISENSORY ARCHITECTURE FOR INTELLIGENT SURVEILLANCE SYSTEMS

Integration of Segmentation, Tracking and Activity Analysis

Francisco Alfonso Cano, José Carlos Castillo, Juan Serrano-Cuerda
Instituto de Investigación en Informática, Universidad de Castilla-La Mancha, 02071 Albacete, Spain

Antonio Fernández-Caballero
Instituto de Investigación en Informática (I3A), Universidad de Castilla-La Mancha, 02071 Albacete, Spain
Departamento de Sistemas Informáticos, Escuela de Ingenieros Industriales de Albacete, 02071 Albacete, Spain

Keywords: Intelligent surveillance systems, Monitoring architecture, Segmentation, Tracking, Activity analysis.

Abstract: Intelligent surveillance systems deal with all aspects of threat detection in a given scene; these range from segmentation to activity interpretation. The proposed architecture is a step towards solving the detection and tracking of suspicious objects as well as the analysis of the activities in the scene. It is important to include different kinds of sensors for the detection process. Indeed, their mutual advantages enhance the performance provided by each sensor on its own. The results of the multisensory architecture offered in the paper, obtained from testing the proposal on CAVIAR project data sets, are very promising within the three proposed levels, that is, segmentation based on accumulative computation, tracking based on distance calculation and activity analysis based on finite state automaton.

1 INTRODUCTION

Nowadays, obtaining surveillance systems able to cover all levels of traditional images processing stages is still a very challenging issue. Intelligent surveillance systems must be able to predict potential suspicious behaviors, to work semi-automatically, and to warn the human operator if necessary. Moreover, for a human operator, the task of looking at monitor during hours is very monotonous. That is why a system able to detect suspicious situations is essential for an efficient surveillance (Gascueña and Fernández-Caballero, 2011).

It is known that color is a very important feature for object recognition. Several approaches can be found in the bibliography devoted to color image segmentation (e.g. (Lézoray and Charrier, 2009)). In (Maldonado-Bascón et al., 2007) *RGB* and *HSI* color spaces are used for the detection of traffic signals. There are other proposals that use image features combined with color to solve the segmentation problem (e.g. (Moreno-Noguer et al., 2008)). Concerning infrared cameras, many proposals focus on the assumption that objects of interest – mostly pedestrians – possess a temperature higher than their surroundings (Yilmaz et al., 2003). Thermal infrared video cameras detect relative differences in the amount of thermal energy emitted/reflected from objects in the scene. As long as the thermal properties of a foreground object are slightly different (higher or lower) from the background radiation, the corresponding region in a thermal image appears at a contrast from the environment. A technique based on background subtraction for the detection of objects under different environmental conditions has been proposed (Davis and Sharma, 2007).

Tracking objects of interest in a scene is another key step prior to video surveillance events detection (Regazzoni and Marcenaro, 2000). Tracking approaches can be classified into four main categories, namely, *tracking based on the moving object region*, where the bounding box surrounding an object is tracked in the 2D space (Masoud and Papanikolopoulos, 2001), *tracking based on the moving object contour*, where a contour is defined by curves delimiting the moving objects and dynamically updated (Is-

Alfonso Cano F., Carlos Castillo J., Serrano-Cuerda J. and Fernández-Caballero A..
MULTISENSORY ARCHITECTURE FOR INTELLIGENT SURVEILLANCE SYSTEMS - Integration of Segmentation, Tracking and Activity Analysis.
DOI: 10.5220/0003477101570162
In *Proceedings of the 13th International Conference on Enterprise Information Systems* (ICEIS-2011), pages 157-162
ISBN: 978-989-8425-54-6

ard and Blake, 1998), *tracking based on a model of the moving object*, where the 3D geometry of a moving object is defined (Koller et al., 1993), and, *tracking based on the features of the moving object*, where some features of the objects are extracted and monitored (e.g. vertices in vehicle tracking (McCane et al., 2002)).

In (Lavee et al., 2009) some recent approaches for video event understanding are presented. The importance of the two main components of the event understanding process – abstraction and event modeling – is pointed out. Abstraction corresponds to the process of molding the data into informative units to be used as input to the event model (Natarajan and Nevatia, 2008), while event modeling is devoted to formally describing events of interest and enabling recognition of these events as they occur in the video sequence (Ulusoy and Bishop, 2005). In close relation to finite state machines theory, in (Ayers and Shah, 2001) the authors describe a system which automatically recognizes human actions in a room from video sequences. The system recognizes the actions by using prior knowledge about the layout of the room. Indeed, action recognition is modeled by a state machine, which consists of 'states' and 'transitions' between states.

2 THE PROPOSED ARCHITECTURE

This section describes in detail the different phases proposed for the intelligent multisensor surveillance architecture. Processing starts after capturing color images as well as infrared images. The use of two different spectral images allows the detection of objects of interest independently of lighting or environmental conditions (day, night, fog, smoke, etc.). The segmentation algorithm detects motion of the scene objects. As a result, the segmentation algorithm generates a list of blobs detected in the scene. The blobs are used as the inputs to the tracking phase.

Following the segmentation process, a simple but robust tracking approach is proposed. An identifier is assigned to each object until it leaves the scene or the segmentation algorithm does not detect it for a determined period of time (application dependent). Finally, the identified blobs are passed to the activity analysis stage, where some predefined activities are modeled by means of state machines. At this stage the different behaviors and their associated alarm levels are obtained.To configure the parameters needed for segmentation and tracking, as well as the static objects definition on the scene a modeling stage has been included to the proposal.

2.1 Segmentation based on Accumulative Computation

The proposed segmentation method has been tested on visual and non visual spectrum (infrared) image sequences with promising results in detecting moving objects (e.g. (Fernández-Caballero et al., 2011; Fernández-Caballero et al., 2010)). The accumulative computation method consist of the five phases described next.

Preprocessing. This phase performs the preprocessing of the input images. Filters such as the mean or the median are applied in order to enhance the image contrast and to smooth the image noise.

Grey Level Bands Segmentation. This phase is in charge of splitting the input image into k grey level bands. That is, each image is binarized in ranges according to the k bands.

Accumulative Computation. This phase obtains one sub-layer per each band defined in the previous phase. Each band stores the pixels' accumulative computation values. It is assumed that motion takes place at a pixel when that value of the pixel falls in a new band. A complete description may be found in (Delgado et al., 2010).

Fusion. This phase fuses the information coming from the k accumulative computation layers. Each pixel is assigned the maximum value of the different sub-bands. Next, a thresholding is performed to discard regions with low motion. Closing and opening morphological operations eliminate isolated pixels and unite close regions wrongly split by the thresholding operation.

Object Segmentation. This phase obtains the areas containing moving regions. As an input a blobs list L_B is obtained and used to higher layers of the architecture.

2.2 Tracking based on Distance Computation

The second level of the proposed architecture consists in tracking of segmented objects.

Object Labeling. The tracking approach uses the result of the previous segmentation stage, that is, the segmented spots, L_B, though the tracking algorithm has its own list of blobs, L_T, updated over time with the tracking results. Firstly, each blob contained in

L_B, L_{Bi}, where $i \in \{0,1,\ldots,N\}$ and N is the number of elements in L_B, is compared to all blobs contained in L_T. The aim is to calculate the distance between the centers of the boxes associated to the blobs.
Blobs $L_{T\,j}$ with a distance between centers below a prefixed threshold are selected as candidates to be the previous position of blob L_B at time instant $(t-1)$. The blob with the minimum distance to L_{Bi} is selected as the previous position of the current blob.

Blob Updating. The box size is smoothed to avoid abrupt variations. If a foreseen blob is not detected during the segmentation process a prediction about its possible trajectory is performed through a mean distance increment and a displacement angle calculated between consecutive frames. If the permanence memory value reaches its minimum, the blob is discarded as it is considered to leave the scene.

Size Adjustment. The detected blobs might include some noise that modifies the size of their containing boxes. In order to alleviate the effects of noise, the obtained blobs are softened according to a weighted mean of their height and width. Also, depending on the motion direction of a blob its position may be modified.

This phase works with the blobs not updated in the previous segmentation phase. For each of them, $L_{T_k'}$, its permanence memory value is reduced under the assumption that the object is probably not present in the scene. Trajectories are predicted for those blobs with permanence above the aforesaid thresholds. This involves the previous calculation of the mean distance between frames. The blobs and their associated identifiers are used to define the activities carried out in the scene.

2.3 Activity Analysis based on Finite State Automaton

The purpose of activity description is to reasonably choose a group of motion words or shout expressions to report activities of moving objects or humans in natural scenes.

Objects of Interest. From the ETISEO classification proposal, four categories are established for dynamic objects and two for static objects. Dynamic objects are, for instance, a *person*, a *group of people* (made up of two or more people), a *portable object* (such as a briefcase) and other dynamic objects (able to move on their own), classified as *moving object*. Static objects may be *areas* and *pieces of equipment*. The latter can be labeled as a portable object if a dynamic object, people or group, interacts with it and it starts moving.

Table 1: Local activities.

Action	Origin vertex
Object information	Object speed
	Object trajectory
Environment interaction	Direction
	Position
Object interaction	Proximity
	Orientation
	Grouping

Description of Local Activities. In order to generalize the detection process we start with simple functionalities that detect simple actions of the active objects in the scene. Using these functions, more complex behavior patterns are built. Simple actions are defined in Table 1.

Description of Global Activities. Interpreting a visual scene is a task that, in general, resorts to a large body of prior knowledge and experience of the viewer (Neumann and Möller, 2008). Through the actions or queries described in the previous section, basic patterns (e.g. the object speed or direction) and more complex patterns (e.g. the theft of an object) can be found. It is essential to define the desired behavioral pattern in each situation, by using the basic actions or queries. For each specific scene, a state diagram and a set of rules are designed to indicate the patterns. Thus, the proposed video surveillance system is able to detect simple actions or queries and to adapt to a great deal of situations. It is also configured to detect the behavioral patterns necessary in each case and to associate an alarm level to each one.

Image Preprocessing. Input image segmentation is not enough to detect the activities in the scene. Thus, the system takes the initial segmentation data and infers the new necessary parameters (such as the objects speed or direction). Thus, the preprocessing techniques described in Table 2 are necessary.

Specification of Simple Behaviors. The system responds to a series of queries intended to find out behavioral patterns of scene objects (see Table 3). These queries are defined as functions and return a logical value, which is true if they are fulfilled for a specific object. They are represented as follows:

$$query\ (parameter_1, parameter_2, ..., parameter_n)$$

Specification of Complex Behaviors. Concerning the complex behaviors, two categories are differentiated:

Local Complex Behaviors. Objects in the scene are associated to a state machine that indicates the state they are in (what they are doing at a time instant). This state machine can be seen as a directed graph

Table 2: Preprocessing techniques.

Preprocessing	Details
Speed Hypothesis	The average speed for each object is calculated by dividing the displacement (Δx) by the time elapsed (Δt) in each frame.
Direction and Moving Direction Hypothesis	To find out the direction of objects, the angle of the straight line that passes through the positions of the previous and current instants are calculated.
Image Rectification	Perspective distortion occurs because the distance between the furthest points from the camera is less than the distance between the closest points. The real position is measured through the weighted distance measure of the four manually placed points closest to the position to be interpolated.
Data Smoothing	The data taken at two time instants is separated with enough time to avoid small distortions; but this distance is small enough to enable accurate results. The distance between both consecutive time instants is called the analysis interval. At each analysis interval, the value of the hypotheses is updated, but the old value is not automatically substituted. To calculate the value at that instant, the mean values are calculated.

Table 3: Simple queries.

Type	Queries	Description
Movement	*hasSpeedBetween* (min, max)	True if an object moves with a speed within range $[min, max]$.
	hasSpeedGreaterThan ($speed$)	True if an object moves with a speed greater than $speed$.
Direction	*hasDirection* ($staticObject$)	True if an object goes toward $staticObject$, being $staticObject$ a static object of the scene.
	isFollowing ()	True if a dynamic object is following another one. The displacement angle is used.
Position	*isInsideZone* ($staticObject$)	True if a dynamic object is in area $staticObject$.
	isCloseTo ($distance$, $staticObject$)	True if the distance to a $staticObject$ is less than $distance$.
	enterInScene ()	True if an object appears for the first time in the scene.

where the vertices are the possible states of the object and the edges are the basic functions or queries previously discussed. An edge has at least one associated outcome of the assessment (true or false) of a query, indicating an action of object, query q_i. Therefore, an edge could have more than one query associated to it. For an edge with several actions to be fulfilled, all the associated queries have to be fulfilled. If a more complex rule is needed, where disjunctions also appear so that an object changes states, the rule must be divided into two edges.

Global Complex Behaviors. To detect global behavior patterns, more than just the local state machine is needed since the states of global state machines are composed by the local ones. These patterns are represented through state machines whose vertices represent a possible state in the scene. Just like in the local state machine, the edges are made up of a series of queries that must be fulfilled at a certain time for the scene to change states.

3 IMPLEMENTATION

The prototype implementation based on the proposed surveillance architecture must fulfill the following objectives: (1) Obtain a detector of strange behaviors and intrusions in a monitored environment, (2) provide a web interface to allow a view independent of the platform, and (3) Provide a scalable prototype.

The proposal captures information from multiple sources, from traditional surveillance sensors (such as IR barriers or motion detectors) to different spectrum cameras (color, thermal and so on). Also, they are allocated on a two-dimensional map to see the current system state.

3.1 Class View

The architecture design is defined by the class view. The class view shows the classes and interfaces that integrates the system, as well as their relations. Thus, not only the static view of the system is established, but also the interactions among the different components.

The control process class is responsible for controlling the acquisition, segmentation, tracking and behaviors detection (activities). The proposal operates in two modes, desktop mode and web mode. The desktop mode is designed to allow a higher interaction with the user, providing interactive views of the different cameras, selected from the map, and warning the user about the alarms through a pop-ups mechanism. On the other hand, the web mode show read-only information about the state of the sensors located in the map and the triggered alarms. By means of a thresholding mechanism, the application triggers two different alarm types: *Pre-Alarm*, for values below the threshold, and *Alarm*, for values above the threshold. As behavior patterns are detected through a state machine mechanism, an alarm value is associated to each state which will be compared to the threshold in order to differentiate between ”Alarm” and ”pre-Alarm”.

3.2 Implementation View

The implementation view define the components that hold the classes defined in the class view. These components define the architecture of the system. A DLL module is implemented for each stage of the proposed architecture, namely **Image Capture** to capture images of cameras, **Segmentation** based on accumulative computation, **Tracking** based on distance computation and **Activities** based on finite state automaton. Moreover, a control class is added to organize the surveillance architecture and an user interface to visualize results and manage configurations.

3.3 Interface

As aforementioned, the system can operate in two different modes: Regarding the desktop mode (see Fig. 1), the interface provides a map to allocate the different sensors (IR barrier, opening sensor, camera). A color code is utilized to indicate the sensor state (grey means "irresponsible", green "ok", yellow "Pre-Alarm" and red "Alarm"). IR barriers and motion detectors are set in red in the map whether the sensor is activated. The cameras can be set in yellow in case of Pre-Alarm, or red in case of Alarm. Besides, the interface provides different features: turning on or off the system operation, showing sensor coverage and its id. Moreover, selecting a camera on the map, the result of its operation is shown. The system offers a view up to eight cameras simultaneously. The view of the camera process is scaled according to the number of selected cameras. In the lower left side of the interface, the alarm detection log is shown. Each time the system triggers an alarm, a new line is added to the log. The information appeared in the log is based on the detected behavior. Thus, the alarms information are summarized through four columns: alarm type, hour of alarm, behavior, and alarm state(accepted or canceled by the user). Under an alarm condition the user interaction is required to confirm or cancel the alarm. Once the user has confirmed the alarm, the sensor involved returns to normal state (green color).

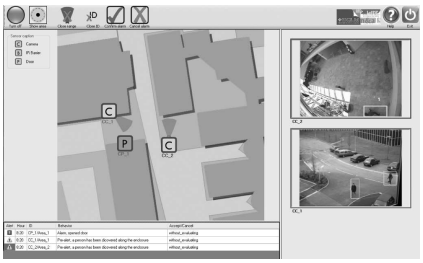

Figure 1: Desktop prototype interface.

Table 4: Segmentation algorithm results with CAVIAR datasets.

Dataset	Accuracy	Sensitivity	F-Score
Browse2	0,885	0,982	0,935
Browse3	0,992	0,855	0,919
Bww	0,995	0,964	0,979
Walk1	0,996	0,915	0,954
Rest1	0,993	0,917	0,953
Mean	0,972	0,927	0,948

On the other hand, the web mode is designed to show concise information about the system state. As seen in Fig. 2, the interface shows a map where the sensors are placed whilst keeping the color code utilized in the desktop mode, as well as the alarms log structure.

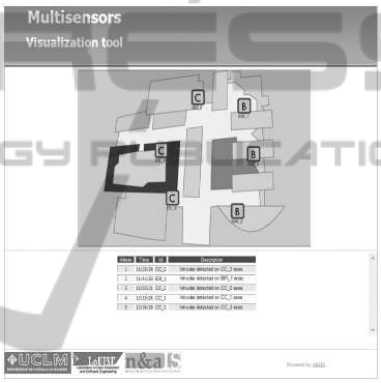

Figure 2: Web Interface.

4 TEST AND RESULT

The test were carried out using the cases that CAVIAR project makes available for researchers. For the tests, videos recorded with a wide angle camera in the lobby of INRIA Labs in Grenoble, France, were used. In these scenes, there are different people interacting with the environment and with each other. The utilized datasets were *walk1*, *browse2*, *browse3*, *rest1* and *browse while waiting*(Bww).

Since segmentation forms the first phase in the proposed architecture, its results must be as accurate as possible because the remaining blocks are built over it. Its results can be shown in table 4, presenting a F-score close to 95%, a sensitivity of 92% and an accuracy of 97%, with most of values close to 100%.

For the activities phase, the test were focused on a complex behavior: position and direction analysis. It must be pointed out that activities detection takes only one from each six frames. These provides robustness against the detection and tracking noise, while pre-

Table 5: Position and direction detection.

Frame	Alert	Object	Estate	Objective
437	Pre-Alarm	1	Initial	Cashpoint
443	Alarm	1	Goings towards the cashpoint	Cashpoint
563	Alarm	1	Close to the cashpoint	Cashpoint

serving accuracy enough to perform inferences about objects trajectories. Global behavior patterns are represented through state machines which vertices represent a possible state in the scene.

To finish with our tests, the table 5 shows an extract of the results for position and direction analysis. The sequence used was “Browse2”.

5 CONCLUSIONS

This article has introduced an intelligent surveillance system by integration of segmentation, tracking and activities detection algorithms. The system is able to detect behaviors and report information to the user thanks to attractive and functional interface. As a future work it is planned to add new sensors types for surveillance and works with a distributive architecture.

ACKNOWLEDGEMENTS

This work was partially supported by Spanish Ministerio de Ciencia e Innovación TIN2010-20845-C03-01 grant, and by Junta de Comunidades de Castilla-La Mancha PII2I09-0069-0994 and PEII09-0054-9581 grants.

REFERENCES

Ayers, D. and Shah, M. (2001). Monitoring human behavior from video taken in an office environment. *Image and Vision Computing*, 19(12):833–846.

Davis, J. and Sharma, V. (2007). Background-subtraction in thermal imagery using contour saliency. *International Journal of Computer Vision*, 71:161–181.

Delgado, A., López, M., and Fernández-Caballero, A. (2010). Real-time motion detection by lateral inhibition in accumulative computation. *Engineering Applications of Artificial Intelligence*, 23:129–139.

Fernández-Caballero, A., Castillo, J., Martínez-Cantos, J., and Martínez-Tomás, R. (2010). Optical flow or image subtraction in human detection from infrared camera on mobile robot. *Robotics and Autonomous Systems*, 58:1273–1281.

Fernández-Caballero, A., Castillo, J., Serrano-Cuerda, J., and Maldonado-Bascón, S. (2011). Real-time human segmentation in infrared videos. *Expert Systems with Applications*, 38:2577–2584.

Gascueña, J. and Fernández-Caballero, A. (2011). Agent-oriented modeling and development of a person-following mobile robot. *Expert Systems with Applications*, 38(4):4280–4290.

Isard, M. and Blake, A. (1998). Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28.

Koller, D., Danilidis, K., and Nagel, H.-H. (1993). Model-based object tracking in monocular image sequences of road traffic scenes. *International Journal of Computer Vision*, 10:257–281.

Lavee, G., Rivlin, E., and Rudzsky, M. (2009). Understanding video events: a survey of methods for automatic interpretation of semantic occurrences in video. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(5):489–504.

Lézoray, O. and Charrier, C. (2009). Color image segmentation using morphological clustering and fusion with automatic scale selection. *Pattern Recognition Letters*, 30:397–406.

Maldonado-Bascón, S., Lafuente-Arroyo, S., Gil-Jiménez, P., Gómez-Moreno, H., and López-Ferreras, F. (2007). Road-sign detection and recognition based on support vector machines. *IEEE Transactions on Intelligent Transportation Systems*, 8(2):264–278.

Masoud, O. and Papanikolopoulos, N. (2001). A novel method for tracking and counting pedestrians in real-time using a single camera. *IEEE Transactions on Vehicular Technology*, 50(5):1267–1278.

McCane, B., Galvin, B., and Novins, K. (2002). Algorithmic fusion for more robust feature tracking. *International Journal of Computer Vision*, 49:79–89.

Moreno-Noguer, F., Sanfeliu, A., and Samaras, D. (2008). Dependent multiple cue integration for robust tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:670–685.

Natarajan, P. and Nevatia, R. (2008). View and scale invariant action recognition using multiview shape-flow models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Neumann, B. and Möller, R. (2008). On scene interpretation with description logics. *Image and Vision Computing*, 26:82–101.

Regazzoni, C. and Marcenaro, L. (2000). *Object detection and tracking in distributed surveillance systems using multiple cameras*. Kluwer Academic Publishers.

Ulusoy, I. and Bishop, C. (2005). Generative versus discriminative methods for object recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 258–265.

Yilmaz, A., Shafique, K., and Shah, M. (2003). Target tracking in airborne forward looking infrared imagery. *Image and Vision Computing*, 21(7):623–635.