Real-Time Image Processing for Active Monitoring of Wide Areas

Christian Micheloni and Gian Luca Foresti

Department of Mathematics and Computer Science, University of Udine
Via delle Scienze 206, 33100 Udine, ITALY

Abstract

A real-time active system able to monitor wide areas by processing video sequences acquired by a pant tilt and zoom (PTZ) camera is presented. The system is able to compensate background changes due to camera motion, to detect and to maintain the gaze on objects moving in the scene. To fit the real-time constraint two speed-ups, representing the novelty of the proposed work, have been introduced. The first speed-up consists in the adoption of a simple yet efficient transformation model for image alignment computed by using a well known feature tracking method. The robustness of such a task has been improved by developing a new feature clustering method to reject badly tracked features. The second improvement relies on the introduction of a reference map, containing well trackable features, that is used to select features in a fast and reliable way. The map is maintained and updated continuously by introducing new features related to new regions appeared in the current frame. Hence, to detect moving objects, the previous and current frame, after compensation, are processed by a change detection method. Finally, a standard Kalman filter is applied to track objects and to determine the pan and tilt angles that the camera has to perform in order to maintain the gaze on the target.

1 Introduction

Detection and tracking of moving objects are important tasks for computer vision especially for visual-based surveillance systems [1–6]. The applications of video-surveillance have an high range of purposes, from traffic monitoring [7,8] (*invece di [9]*) to human activity [10] and behaviour [11] understanding . Video surveillance applications, mostly, imply to pay attention to wide areas, hence different kinds of cameras are generally used, e.g, fixed cameras [3, 5], omnidirectional cameras [12, 13] or mobile cameras [4, 14–17]; here, PTZ cameras are considered.

Preprint submitted to Elsevier Science

Motion segmentation is generally considered a difficult task if image sequences are acquired by a moving camera [15–17], [18]. As matter of fact, when comparing two consecutive frames of a sequence, differences in pixel intensities occur in the whole image, since the movement of the camera causes an apparent (induced) motion of static objects. In the following, such set of static objects in the scene will be indicated with the term *background*.
The literature proposes several methods for segmenting the motion in active camera sequences. Mainly, these methods are based on temporal image differencing techniques that can be classified in two major groups [19]: a) frame to background [20–22] and b) frame-by-frame [15], [17]. The former consists in building and maintaining a background image of the whole monitored scene. Camera position is estimated and the appropriate subsection of the whole background is used as reference for image differencing. Instead, techniques belonging to the latter group compute the image differences between two time related images. These, can be consecutive frames or frames acquired at particular time instants. Hence, since the compensation of the apparent motion with the alignment of all the pixels belonging to static objects is mandatory, registration techniques have to be considered. A survey of such methods [23] explains how particular transforms (translational, affine, perspective, etc.) have to be computed for registering images.
Murray and Basu [15], in order to correspond pixels of the current image to those in the previous, compute the background motion by reading at each time instant the camera pan and tilt angles. Such readings are then used to compute the transform. This technique just allows rotation of the camera about the lens centre and shows problems in synchronization between image acquisition and angles readings.
Irani and Anandan [17] address the problem of moving object detection in multi-planar scenes by using a direct method which estimates a "dominant" 8-parameters of an affine transformation. The solution is achieved by solving a linear system that sometimes can be very large requiring a high computational cost. Araki *et al.* [18], proposes to estimate the background motion by using a set of features. This is obtained by tracking some feature points on the background and estimating the parameters of an affine transformation between the previous and the current frame. In such a method, since feature tracking is not reliable, principally due to noise, occlusions and also to tracking errors, an iterative process ($LSMedS$) is executed on the entire set of tracked features. The resulting optimal features are therefore used to compute an affine transform. Although effective, the proposed method requires a relevant number of DSPs thus avoiding its employment for real-time application on low-cost platforms. Therefore, to reduce the number of features processed by the $LSMedS$ algorithm, feature rejection or outlier detection rules could be used. In [24], Shi and Tomasi adopt *a priori* defined threshold to reject features whose residual [1] is too high. In [25], Tommasini *et al.* use a statistical

[1] The residual is defined as the absolute difference between the current and the

rule to identify and to reject the outliers (i.e. badly tracked features). Even though more reliable then Shi and Tomasi technique, also Tommasini *et al.* require a large number of feature (about 100) to track. Obviously, such a requirement limits the performance of the system.
Recently, Sugaya and Kanatani proposed a technique to remove outliers [26] to extract moving objects from a moving camera sequence [27]. The performance of such algorithm, based on *subspace separation*, does not allow a real-time outlier detection. Indeed, just considering the outlier detection task the system performance shows an upper bound of $4fps$. Lately, Guo *et al.* [28] proposed a linear combination representation for outlier detection. The proposed scheme works on 4 frames by estimating 4 parameters for each of them. As for the other outliers detectors the number of required features is too big for a real-time application.
In this paper, we propose a motion segmentation method for video sequences acquired by a PTZ camera rotating on the lenses centre. The main characteristics of the proposed solution are the low computational requirements, hence a fast method, and the reliability of both image alignment and detection of moving objects techniques. To achieve these results, a feature based method has been developed to compute the background transformation needed by a frame-by-frame change detection algorithm. Moreover, conjecturing on a real-time processing allowed us to model the induced motion with a translational transform [29]. In addition, if compared to more complex models, the reliability of the image alignment is not affected. Thanks to such scheme a lower number of features can be considered hence reducing the computational effort. Moireover, the coarser alignment model allowed to develop an innovative method that based on a Map of features is capable of a the fast feature selection. The Map is studied to maintain track of good features through the camera motion.
Moreover,a further reduction has been obtained by employing a new feature rejection rule, based on a feature clustering technique [30]. Such a technique allows a reliable execution of the entire method on a very low number of features (in the range $10 \div 20$). The obtained results show how the proposed method is as robust as those proposed by Irani [17] and Araki [18] but faster thanks to lower computational requirements.
The system is completed by an object tracker that based on a Kalman Filter maintains the gaze on the target. The position of the tracked object at next time instant is estimated. Therefore, on the basis of a pin-hole model, the PTZ camera is driven to centre the target in the image centre.
Results will show how, thanks to the map, a fast and reliable feature extraction is performed for feature tracking and how the proposed alignment method can work with a lower number of features than those required by the principal methods developed so far. We will highlight how both techniques allow to speed up the image registration process yielding to a robust real-time object

previous feature

detection.

1.1 Notation

Before proceeding with the description of the proposed method, let us introduce some notation used hereafter. Let $I(i)$ be the image at time instant i, $f^i(\mathbf{x})$ be a $W \times W$ feature (i.e. a corner) in $I(i)$ centred at pixel position $\mathbf{x} = (x, y)$ and $G : I \times X \times Y \rightarrow \{true, false\}$ be the function that classifies the feature centred in $\mathbf{x}$ as good or bad trackable feature. Such a function has been written according to the heuristic introduced by Shi, Tomasi and Kanade [24, 31] which requires the following steps:

- Given the feature $f^i(\mathbf{x})$, compute the matrix

$$\mathbf{M}^i(\mathbf{x}) = \sum_{f^i(\mathbf{x})} \begin{bmatrix} I(i)_x^2 & I(i)_x I(i)_y \\ I(i)_x I(i)_y & I(i)_y^2 \end{bmatrix}$$

  where $I(i)_x = \frac{I(i)}{\partial_x}$ and $I(i)_y = \frac{I(i)}{\partial_y}$ represent the two spatial partial derivatives.
- Compute the two eigenvalues $\lambda_1^i(\mathbf{x})$ and $\lambda_2^i(\mathbf{x})$ of the matrix $\mathbf{M}^i(\mathbf{x})$
- Label $f^i(\mathbf{x})$ as a good trackable feature if, given a threshold λ_{th}, the following holds:

$$\min(\lambda_1^i(\mathbf{x}), \lambda_2^i(\mathbf{x})) > \lambda_{th} \tag{1}$$

Then, G is defined as follows:

$$G(I(i), \mathbf{x}) = \begin{cases} true & \text{if } \min(\lambda_1^i(\mathbf{x}), \lambda_2^i(\mathbf{x})) > \lambda_{th} \\ false & \text{otherwise} \end{cases} \tag{2}$$

Notice how the goodness of the extracted features depends on the value λ_{th}. In fact, if it was too low we would have the extraction of a large number of features, while if it was to high we could prevent the extraction of any feature. Therefore, we first normalize the eigenvalues in the range $[0, 255]$ then an adaptive thresolding technique [32] is applied.
The function G is therefore applied on the whole image to detect good features to track and to build a Trackable Feature Set (TFS_i) given by the best m features, e.g. those with highest eigenvalues.
Let $T : \{f^i(\mathbf{x}_1), f^i(\mathbf{x}_2), \ldots, f^i(\mathbf{x}_m)\} \rightarrow \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_m\}$ be the function that estimates the local displacement of each feature belonging to a set of features. It is defined as follows:

$$\mathbf{d}_j = \left[\mathbf{M}^i\right]^{-1} (\mathbf{x}_j)\xi \tag{3}$$

where $[\mathbf{M}^i]^{-1}(\mathbf{x}_j)$ is the inverse of the 2×2 matrix used to compute the eigenvalues and ξ is the feature residual [31]. Then, the function T corresponds features belonging to the TFS_i from previous to current frame. The new position of a feature $f^{i+1}(\mathbf{x}'_j)$ is given by $\mathbf{x}'_j = \mathbf{x}_j + \mathbf{d}_j$.
Since extracting good feature from scratch for each time instant is very expensive we maintain track of the positions of good trackable features in a map of features. In order to achieve this result, the map $\mathbf{MAP}^i : X \times Y \rightarrow \{true, false\}$ is defined as follows:

$$\mathbf{MAP}^i(\mathbf{x}) = \begin{cases} 1 \; if \; G(I(i), \mathbf{x}) = true \\ 0 \; otherwise \end{cases} \tag{4}$$

1.2 Outline

The structure of the remainder is as follows: In Section 2, we present a brief description of the proposed method by highlighting the computation flow through the developed modules. In Section 3, the new developed method to achieve a fast feature selection is presented in terms of a *Map* that is updated at each time instant. In particular, we describe a method for detecting good trackable features for background motion estimation. In Section 4, we describe the motion segmentation module for detecting moving objects inside the monitored scene. Finally, in Section 5 results are presented in context of a video surveillance application to highlight the efficiency of the proposed method compared with other ones.

2 Method Description

The proposed method has been studied to give a solution to the problem of detecting moving objects inside an area monitored with a PTZ camera. A feature based method has been considered to estimate the background motion for images alignment purposes. Features must be first selected on static objects then tracked over two consecutive frames. Their displacements are finally used to compute a transform that best aligns the considered frames. The main problems of these methods concern on the execution of the feature selection and the transform computation tasks. Therefore, the proposed solution can be analysed by considering the two major tasks involved: Fast Feature Selection (FFS) and Motion Segmentation (Fig. 1).

The FFS task has been studied to select in a fast and reliable way good trackable features belonging to static objects. Here, the main technical issue is

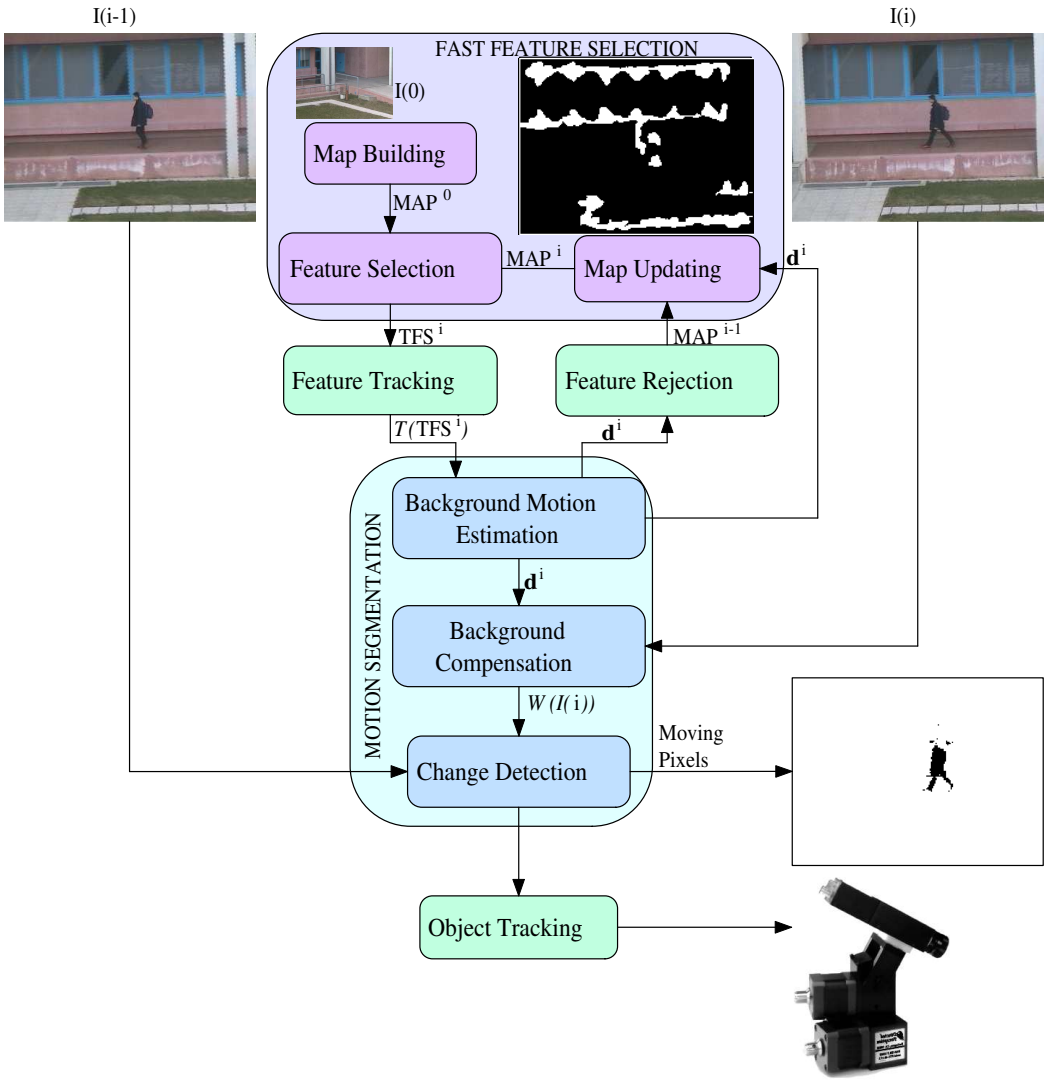


Fig. 1. General logical architecture of the proposed method

represented by the introduction of the concept of *Map of Features*. The MAP is used to maintain track of the position of good trackable features through the camera motion. This allows the selection of features just by checking inside the MAP.

At the initialization step, a first Map ($\mathbf{MAP}^0$) is built by executing the function G for each pixel of image $I(0)$. The feature selection module checks whether at the position $\mathbf{x}$ there is a good trackable feature ($\mathbf{MAP}^i(\mathbf{x}) = 1$) or not ($\mathbf{MAP}^i(\mathbf{x}) = 0$). Good features are finally selected and inserted into the current TFS^i.

Once the background motion estimation has been completed, the position of good features must be update. In addition, possible new features must be considered inside new background sections introduced by the movement of the camera. The MAP updating module takes into account all these issues by updating the good features' position and by extracting good features within

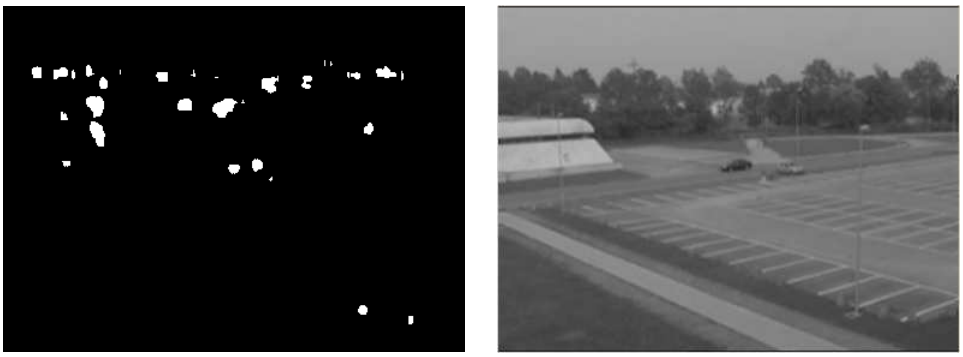
new areas.

The Lucas-Kanade-Tomasi tracker T [31,33] implemented in OpenCV library [34], is applied on the TFS^i generated during the last feature selection phase. The output (a set of the local displacements $\mathbf{d}_j^i$ for each feature $f_j^i \in TFS^i$) is used to segment the real motion in the current frame. Then, the motion segmentation first computes the background motion estimation then warps the current image on the basis of the computed transform.
Background motion estimation, at time instant i, takes as input the set $D^i = T(TFS^i)$ and uses a feature clustering property to estimate a vector $\mathbf{d}^i$. This vector, under the assumptions of a translational model, represents the motion of the background and therefore the transformation for the image alignment. Therefore, the background compensation applies the computed transform on the current image yielding to a translated image $I'(\mathbf{x}, i)$ whose static pixels are aligned with the static pixels in the previous image.
Once having registered the current image, a change detection technique [35] is applied on the $I'(\mathbf{x}, i)$ and $I(\mathbf{x}, i-1)$ in order to detect the *Moving pixels* (i.e. pixels whose intensity substantially changed between the two frames). Such pixels are then used as seeds in a region growing process [36] which determines moving regions and therefore the blobs corresponding to the moving objects. Once a blob is selected for tracking, a technique based on a Kalman Filter [37,38] is applied to maintain the detected object in the centre of the camera frame.

To close the loop for the next iteration, a feature rejection module has been developed to overcome the errors of the feature tracking step. Bad tracked features are removed from both TFS^i and MAP^i. First,those features $f_j^i \in TFS^i$ whose local displacement $\mathbf{d}_j^i$ is different from the estimated background displacement $\mathbf{d}^i$ are deleted. Then, MAP^i is updated in such a way that $MAP^{i+1}(\mathbf{x}_{f_j^i}) = 0$ if f_j^i is a bad tracked feature.

3 Fast Feature Selection

The fast feature selection is composed by two modules for the current MAP computation and by a module for the selection of a set of good trackable features. The current MAP^i can be indeed obtained in two different ways: a) by building it from scratch or b) by updating it by considering important information delivered by the motion segmentation task.
At the initialization of the system, MAP^0 is built from scratch by computing the function G over the entire image. In the proposed version, the function G is based on the quality criterion proposed by Tomasi *et al.* [31], but any other quality criterion for feature extraction (i.e. the one proposed by Schmid [39]) can be here employed. In Fig. 2, the map computed on a real image is shown.



(a) (b)

Fig. 2. Map representation (a) of a real image (b)

The MAP updating process is based on a simple observation: *as the background translates so the good trackable features do*. So, as well as tracked features are used to estimate the displacement vector $\mathbf{d}^i$ between the current and the previous frame, the MAP^i could be translated by the displacement vector $\mathbf{d}^i$. The new $MAP^{i+1}(\mathbf{x})$ will be used in the next feature selection process.
During this operation, features belonging to regions no longer present in the current frame $I(i)$ are rejected from the new map. Nonetheless, as the camera moves, new regions are introduced into the image hence requiring particular attention from the updating process. It executes the extraction function G only over the new areas.
Then, the updating process (see Fig.3) for the computation of the MAP^{i+1} from the current MAP^i can be summarized as follows:

$MAP^{i+1}(\mathbf{x}) = MAP^i(\mathbf{x} + \mathbf{d}^i)$
for each $\mathbf{x} \in$ new regions of $I(i)$ **do**

$$MAP^{i+1}(\mathbf{x}) = G'(I(i), \mathbf{x}) = \begin{cases} 1 \text{ if } G(I(i), \mathbf{x}) == true \\ 0 \text{ otherwise} \end{cases}$$

end for

Since the iteration depends on the displacement vector $\mathbf{d}^i$, we shorten the expression to the form:

$$MAP^{i+1}(\mathbf{x}) = MAP^i(\mathbf{x} + \mathbf{d}^i) + G'^{\,i}(\mathbf{d}^i) \tag{5}$$

where $G'^{\,i}(\mathbf{d}^i)$ represents the set of good features extracted from the regions appeared in the current frame.

This method allows to save a lot of computational time elsewhere spent in feature optimality calculation on the whole frame. Here, only few regions of features, related to the camera motion, are analysed by the function G. Hence, let $O(W^2)$ be the complexity of the extraction function G then the proposed

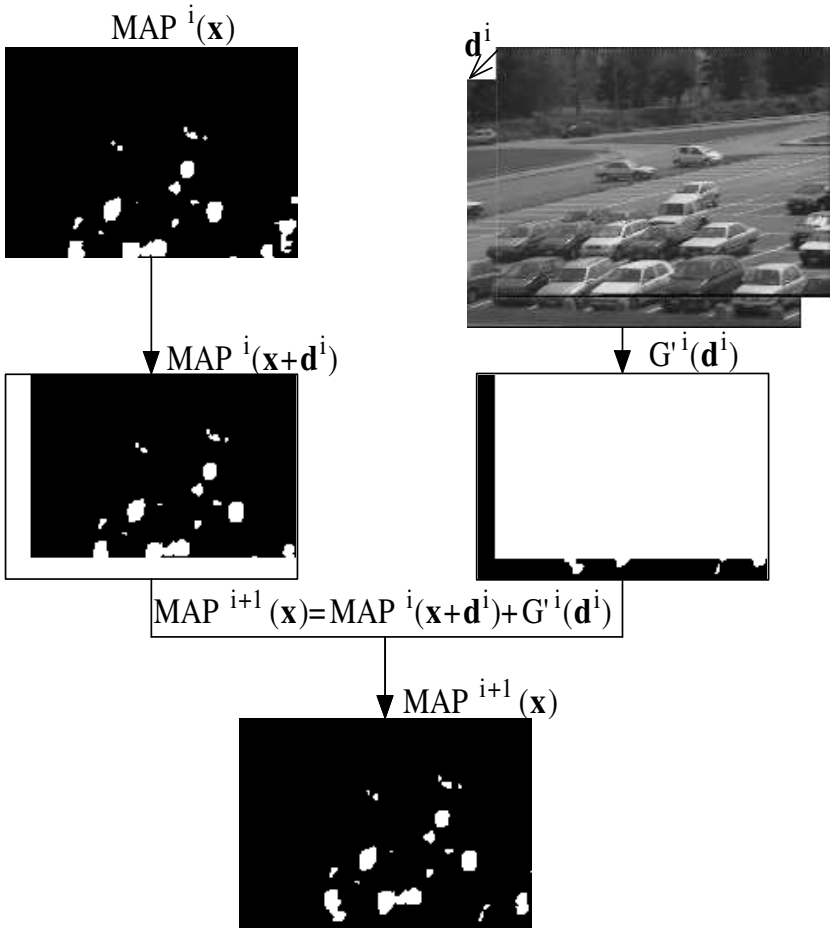


Fig. 3. Map updating process

method has the following complexity:

$$O((d_x^i \times N + d_y^i \times M)W^2 + N \times M) \tag{6}$$

where $\mathbf{d}^i = (d_x^i, d_y^i)$ and the image size is $N \times M$. This represents a great improvement if compared to the extraction of all the features from scratch $O(N \times M \times W^2)$. This improvement is highlighted by the time performance evaluation, where the proposed method works about 100 times faster ($0,01s$ vs. $1s$ on a 1.2GHz PC).

In addition, to satisfy real-time constraints, without using specialized hardware like DSP [18], we have to work with few features. Then, these must be selected carefully for they inclusion into the TFS_i. In particular, it could happen that all features belonging to the TFS_i are located on a small region (i.e., TFS_i consists of few neighbourhood points). In such a case it could be possible that the whole TFS_i is wrongly tracked due to noise or occlusions. To avoid this problem, only appropriate features should be selected from the MAP^i. This selection is performed in two steps. In the first step, a feature f_j is extracted from the MAP^i; in the second one, all neighbourhood features of

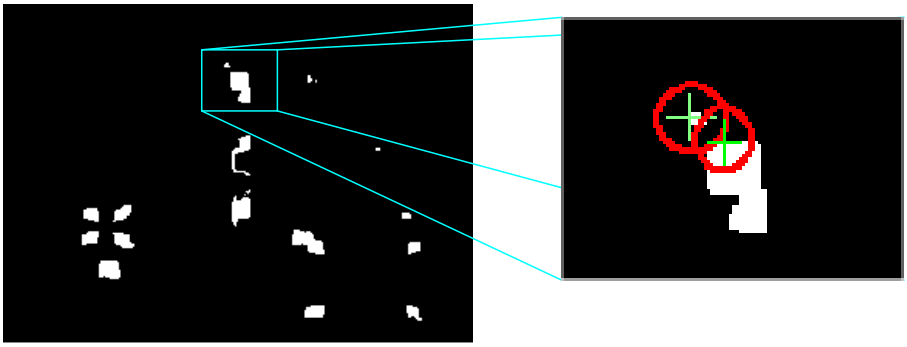


Fig. 4. Feature selection inhibition. The second feature is selected outside of the inhibition circle imposed by the first feature selection.

f_j are inhibited at the next selection. Neighbourhood inhibition is an important task in feature selection and an example of this process is shown in Fig.4. The neighbourhood of a feature f_j consists in a circle with centre on f_j and radius equal to a prefixed threshold R_{th} which depends on the complexity of the scene. Textured background objects allow to extract a large number of well trackable features, so an higher R_{th} value could be considered. On the other hand, a smaller R_{th} value is required for scenes with homogeneous background containing few static objects. Experimental tests have demonstrated that the 20 for the R_{th} parameter is a good trade-off for sequences acquired in outdoor environments. The selection process must be repeated for a number of times enough to select the desired number of features. On each iteration, the R_{th} value is decreased of an amount $\Delta_{R_{th}}$ defined experimentally (i.e. in our application $\Delta_{R_{th}} = 5$). This selection mode allows to track good features spread on different areas of the image.

4 Motion Segmentation

The *Background Motion Estimation* module, thanks to data computed by the feature tracking algorithm, estimates the induced motion occurred on the image plane. Such an estimation will be used by two more modules: a) *Background Compensation* that compensates the motion of static objects occurred between two consecutive frames and b) *Change Detection* which first performs the image differencing then thresholds the result to highlight only the moving objects inside the monitored scene.
Once the TFS^i is built, the feature tracking algorithm T is applied on it. The output, $T(TFS^i)$ is a correspondence relation among selected features in the current frame $I(i)$ and in the previous $I(i-1)$. It often occurs that some features are not well tracked due to noise, occlusions, etc. In order to face this problem it is necessary to distinguish features tracked well from the others (outliers). By analysing the behaviour of tracked features it has been possible

to deduce the following feature clustering proposition [30]:

Features belonging to objects with different velocities, if correctly tracked, group themselves in clusters on the basis of their local displacement.

This proposition allows us to define

$$C(\mathbf{d}_c^i) = \left\{ f_j \in TFS | \mathbf{d}_j^i = \mathbf{d}_c^i \right\} \tag{7}$$

as the cluster of the features whose displacement is equal to the vector $\mathbf{d}_c^i$. Here, the distance between the features or other geometrical aspects are not considered for the cluster association. Let $C = \{C(\mathbf{d}_c^i)\}$ be the set of all clusters generated by the tracking algorithm; for each of them it is defined a reliability factor RF as indicator of the reliability of its displacement with regards to the background motion. Let RF be defined as follows:

$$RF(C(\mathbf{d}_c^i)) = \frac{\sum_{f_j^i \in C(\mathbf{d}_c^i)} E_{f_j^i}}{|C(\mathbf{d}_c^i)|^2} \tag{8}$$

where $E_{f_j^i} = \sqrt{\sum_{\mathbf{x} \in f_j^i} \left[f_j^i(\mathbf{x} + \mathbf{d}_j) - f_j^{i-1}(\mathbf{x}) \right]^2}$ is the residual of the feature between its position in the previous and current frame and the $|\cdot|$ operator returns the cardinality of a set. Then, according to the definition of the RF factor and to the definition of the displacement clustering, the displacement vector $\mathbf{d}^i$ is determined as follows:

$$\mathbf{d}^i = \mathbf{d}_l^i |\ RF(C(\mathbf{d}_l^i)) = \min_{k=1\ldots|C|} \{ RF(C(\mathbf{d}_k^i)) |\ C(\mathbf{d}_k^i) \in C \} \tag{9}$$

where $C(\mathbf{d}_l^i)$ is the cluster selected to represent the background displacement.

After warping the current image by the computed displacement vector and having computed the difference with the reference frame a thresolding phase follows. For such a step, we considered the technique proposed by Rosin [35] which is an adaptive thresholding method computed on the current image. Besides this, the frame by frame technique results in a coarse identification of the moving objects especially if their speed is slow, as shown by Collins *et al.* [40]. To sidestep this problem, we have adopted a strategy that adaptively selects previous frames on the basis of the speed of the object of interest and therefore of the camera motion. Precisely, the rotation speed of the camera is determined as consequence of the object's speed on the image plane. Therefore, if the target appearance, in the current detection, is not meaningful we select an older frame as reference frame. This selection is done iteratively until the detection is enough accurate or the displacement between the two selected images is grater than $2 \times W$. In the latter case, we suppose that the object is became static and it is still in its previous position. Here, it is important to

notice that we do not need to compute again the displacement between the current and the selected reference frames, but we apply iteratively the already computed displacement between all consecutive frames as follows:

$$\mathbf{d}^{rf} = \sum_{j=rf+1}^{i} \mathbf{d}^{j} \tag{10}$$

where rf is the index of the reference frame and $\mathbf{d}^j$ is the computed displacement between time instants j and $j-1$.

5 Experimental Results

The proposed method has been tested on sequences acquired in an outdoor environment represented by a parking area around the university building. Several sequences have been acquired by changing the pan and/or the tilt and the zoom parameters of the camera. The tests have been performed to evaluate first the proposed method for what concerns the feature extraction and selection then to evaluate the motion segmentation for the moving object detection. Different types of sequences grouped into two different scenarios have been acquired.

The first scenario consists on sequences where no moving objects are present hence ideal to evaluate the feature extraction and image alignment modules of the system. Precisely, we applied multiple rotations to the PTZ camera in order to evaluate the correctness of the background displacement estimation performed by the proposed method. This is the simplest scenario since no feature occlusions occur (e.g., caused by moving objects). In addition, sequences belonging to a scenario with one or more moving objects have been considered to test the motion segmentation and evaluate the overall performance of the proposed system. Since no higher level modules were provided, the selection of the target to track was demanded to a human operator.
The sequences adopted in the testing phase have been acquired in 45 days considering different weather conditions (sunny, cloudy and rainy day). In Fig. 5, some frames of the used sequences are shown. The images represent how the proposed method identifies and tracks the features.

5.1 System Setup

The sequences used for experiments are acquired by a Cohu 3812 CCD camera mounted on a Robosoft Pan-Tilt Unit (PTU 46-17.5) and are characterized

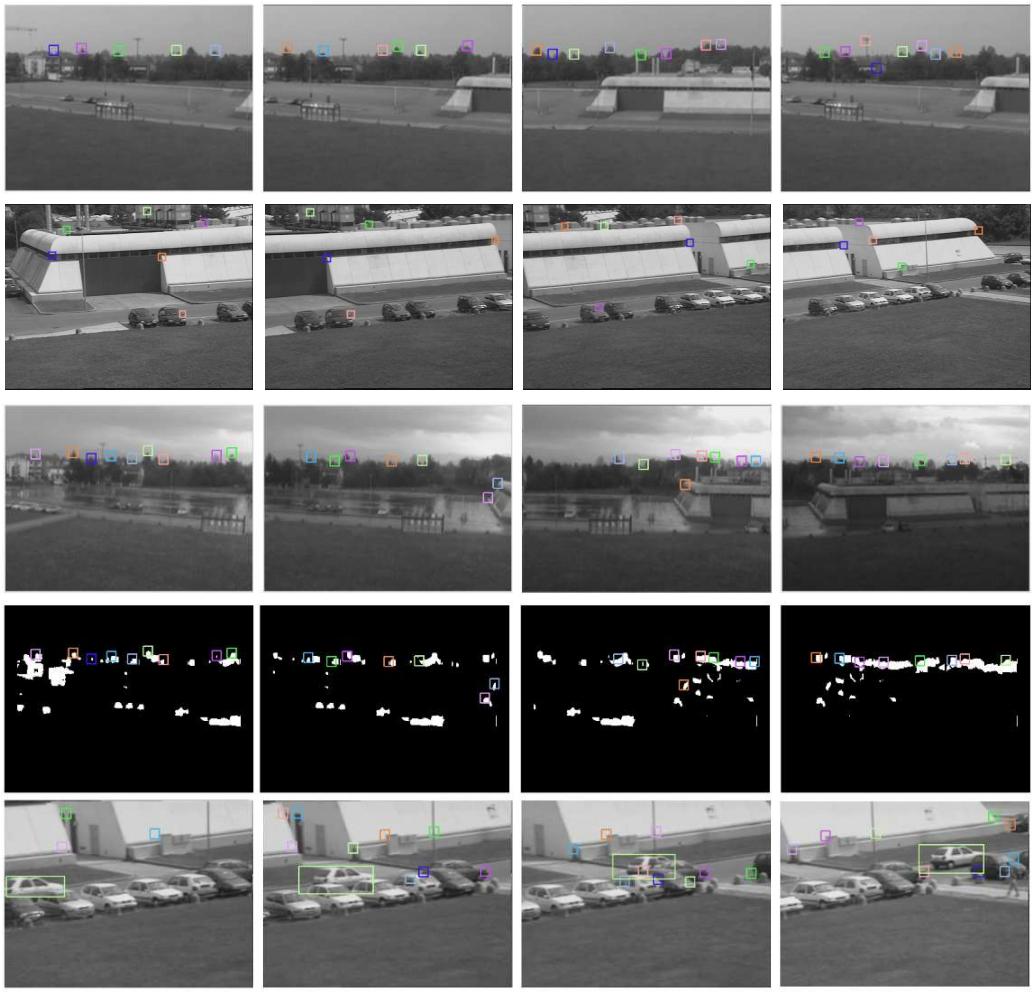


Fig. 5. Three different conditions considered for the first scenario are here presented: a) cloudy day with low 1× zoom level (top row), b) sunny day with 4× zoom level and c) heavy rainy day with 1× zoom level. The forth row shows the feature's map state related to the frames in the third row. Finally, some frames from a typical sequence in which there are moving objects in the scene is presented in the fifth row.

by images of 384x288 pixels. The tests have been performed on a platform composed by an Athlon 1.2GHz processor, 512MByte of RAM and a Matrox Meteor II frame grabber. In addition, for the motion detection subsystem, the method proposed by Tomasi and Kanade [31] has been selected as function G and the method proposed by Shi and Tomasi [24], based on [33], has been selected as function T. Such technique have been executed on feature with dimension $W = 11$.

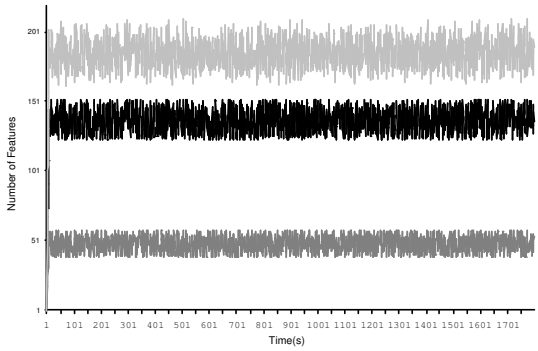


Fig. 6
The graph plots the average number of features miss-classified (light grey) during the 30 minutes of the 10 test sequences. In addition the average number of missed (grey) and false features (black) are plotted as well.

	Bad Features	
$\|TFS\|$	μ	%
10	0.32	3.2
20	0.88	4.4
30	1.38	4.6
40	1.97	4.9
50	2.71	5.4
60	3.42	5.7
70	4.34	6.2
80	5.36	6.7
100	7.28	7.3

Table 1
The average number of bad features selected and inserted in the TFS and the percentage of these features with regards of the cardinality of the TFS are shown.

5.2 Fast Feature Selection

In the first scenario context, we ran the proposed algorithm on 10 sequences of about 30 minutes each (10*30min*60sec/min*25fps=450000 frames) of footage and obtained by rotating back and forth the pan-tilt unit in a range of 120° and at the speed of $8°/sec$. We have therefore evaluated the performance of the system by considering its different aspects. First of all, we verified the reliability then the efficiency of the fast feature selection module.

Reliability
To measure the reliability we have computed for each frame the differences between the Map and the image of the features extracted from scratch on that particular frame. In Fig. 6, a graphical representation shows how the proposed updating method allows to have a small error in the extraction of the features. Precisely, the average number of wrongly classified features (missed and false) is 185 with standard deviation $\sigma = 8.69$ over 3923 of good features, hence representing the 4.71%. Of this amount, 137 (74%) with $\sigma = 10.56$ are bad features (false) in the map and 48 (26%) with $\sigma = 5.81$ are the good trackable feature not available (missed) in the map. It is interesting to notice how these values are not affected by the time which means that the proposed feature extraction method is stable in time.

	Proposed			Tomasi and Kanade		
Levels	1	2	3	1	2	3
Time	$0.011s$	$0.015s$	$0.017s$	$0.891s$	$0.208s$	$0.078s$
Missed	1.22%	1.98%	2.89%	0%	1.08%	2.22%
False	3.49%	2.48%	2.01%	0%	0%	0%

Table 2
Comparison between proposed Map approach and a multi-resolution version of the Tomasi and Kanade method [31]. Data have been computed as average on the sequences for the first scenario.

Moreover, in the Table 1 we can see how the number of bad features selected is affected by the errors introduced by the map updating process. Such value goes from the 3.2% when $|TFS| = 10$ to 7.3% when $|TFS| = 100$. The increasing trend is mostly due to the fact that a good trackable feature is often surronded by other good trackable features forming a sort of connected component. Besides, even though the selection tries to pick out features in the centres of such components, the inhibition policy force the selection on the borders. Those areas are indeed the principal regions subject to the errors committed by the updating process. This effect can be solved by reducing the inhibition circle as consequence of the increasing dimension of the TFS.

Efficiency
To measure the efficiency of the proposed method, we compared it with the feature extraction introduced by Tomasi and Kanade [31]. Precisely, we juxtaposed the Map method to a multi-resolution approach where good features detected on lower resolution levels are propagated to higher resolution levels. As can be seen from Table 2, the proposed method does not gain in speed with the use of multiple levels but even decrease its performance. This is the result of the map updating for the different levels. Instead, for what concerns the performance multi-resolution version of Tomasi and Kanade method, the more the considered levels the better. However, the proposed method ran on a single level is still 7 times faster than the Tomasi and Kanade method ran on three levels ($0.011s$ Vs. $0.078s$). Besides, while looking at the reliability of the two approaches, our method extracts false features and the other does not, if we look at the missed features the proposed works better (1.22% Vs. 2.22%). Concluding the comparison, the proposed approach is faster than the one proposed by Tomasi and Kanade, even if boosted with a multi-resolution approach, and it is as well reliable like shown during the reliability evaluation.

To measure the performance of the motion segmentation module we followed the same strategy adopted in the previous tests. We first evaluated the reliability then the efficiency and the accuracy.

Reliability

For what concerns the reliability of the motion segmentation, multiple metrics have been selected for this purpose. First, the module of the displacement error **MDD** has been considered to give a qualitative measure of the displacement estimation algorithm. It is given by the difference between the estimated vector **d** and the real image displacement **rd** (i.e., the displacement that minimize the compensation error thus minimize the percentage of misclassified pixel, i.e. static pixels classified as moving and vice versa) and it is calculated as follows:

$$MDD = \|\mathbf{d} - \mathbf{rd}\|_2 \tag{11}$$

In addition, since the motion segmentation is based on a feature clustering algorithm, to verify its robustness, we used two more metrics: the number of good features rejected (GFR - missed) as the percentage of features rejected that would be considered well trackable and (b) the number of bad features maintained (BFM - false) as percentage of features not rejected that would be considered not well trackable:

$$GFR = \frac{|\{\mathbf{x}|\mathbf{x} \in TFS_i \wedge \mathbf{x} \notin TFS_{i+1} \wedge G(I(\mathbf{x}, i+1)) = t\}|}{|\{\mathbf{x}|\mathbf{x} \in TFS_i \wedge \mathbf{x} \notin TFS_{i+1}\}|} \tag{12}$$

$$BFM = \frac{|\{\mathbf{x}|\mathbf{x} \in TFS_{i+1} \wedge G(I(\mathbf{x}, i+1)) = f\}|}{|TFS_{i+1}|} \tag{13}$$

For each of these parameters, the average μ and the maximum value Max over sequences have been computed. In particular we first analysed the performance on first scenario sequences then on 4 hours ($4h*60min/h*60ses/min*25frame/s = 360000frames$) of footage where at least one moving object were acquired.

Both scenarios have shown a good behaviour of the system. In Table 3, the values of the considered parameters for the first scenario sequences are shown. The MDD parameter shows a good estimation of the background displacement. This result is due to the ability of the proposed method to maintain a good feature set. This is guaranteed by the low number of bad features maintained in the TFS. Nonetheless, the average time required to update the MAP and to select the new features is $0.024s$, allowing the system to operate at the frame rate of about 25 frame/s.

In Table 4, the values of the considered parameters computed on sequences belonging to the second scenario are shown. The system behaviour is good also in these conditions demonstrating a great tolerance to the presence of moving objects in the scene. Indeed, the obtained values are similar to those

	MDD	GFR%	BFM%
μ	0.181	4.11	1.01
Max	4	33.3	12.5

Table 3
The MDD value shows how reliable our estimation algorithm is by computing a displacement vector that has the average error of only 0.181 pixels. GFR and BFM highlight how the proposed system prefers to rejects good features whether to maintain bad features. Values have been computed as average on 5 hours footage of first scenario

	MDD	GFR%	BFM%
μ	0.25	5.32	1.12
Max	7	33.3	22.2

Table 4
Reported values show how the behaviour of the system is not affected by the presence of moving objects in the scene. Indeed, the values computed as average on the 4 hours of footage for the second scenario are similar to those computed for the first as showed in Table 3.

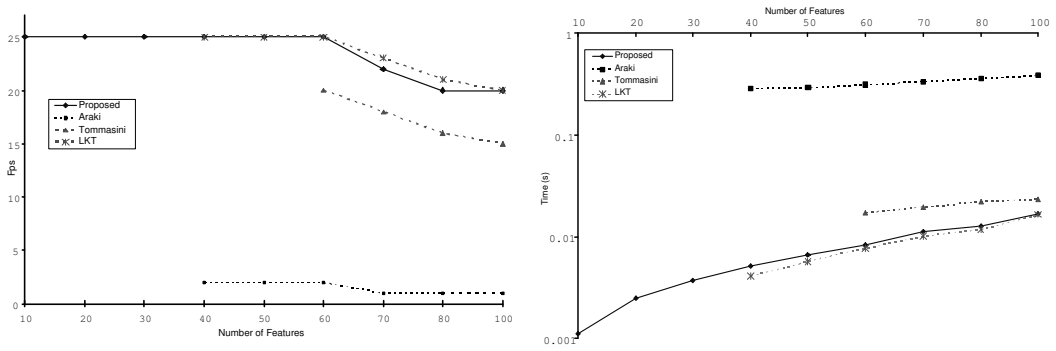
computed for the first scenario. In addition, the frame rate is still equal to 25 frame/s.
The results, obtained from the two scenarios, show how the system performs a good displacement estimation regardless the presence of moving objects.

Efficiency and accuracy
To give a better evaluation of the system, for what concerns its efficiency, the proposed alignment technique has been compared with the methods proposed by Araki [18], Lucas-Kanade-Tomasi (LKT) [31, 33, 34] and by Tommasini *et al.* [25]. In particular, the proposed translational alignment has been compared to the affine alignment proposed by Araki, while the methods proposed by LKT and by Tommasini *et. al* have been used as feature rejecters. Then, the resulting features have been considered for the computation of the image alignment. As shown in Table 5, the average compensation error of the proposed method is minor than the others, while the maximum is really close to the maximum error performed by the Araki's algorithm. Hence, we can conclude that the alignment accuracy of the proposed solution is as good as the Araki's one and really better then those obtained by employing the techniques proposed by Lucas-Kanade-Tomasi and by Tommasini *et al.*.
The quality evaluation of the proposed method has been performed by testing each method off-line. Thus, to every algorithm we gave the necessary time to process the current frame before proceeding to the next. In Fig. 7, for each method, the processed frames per second and the execution time for each frame versus the number of used features are plotted. Looking at the computational effort we can say how the proposed method is faster than all the others. Therefore, we can again conclude that the proposed method is accurate as the Araki's one but really faster.

Overall performance
On the 4 hours of ground-truth footage for the second scenario (excerpts of sample sequences are shown in Fig. 8), we have measured the overall perfor-

	Compensation Error	
	Mean μ	Max
Proposed	12.54	19.44
Araki [18]	15.49	17.32
Tommasini *et al.* [25]	25.77	49.45
Kanade-Lucas-Tomasi [34]	28.89	63.02

Table 5
The compensation error is given as percentage of the number of pixels miss-classified by the algorithms over the number of miss-classified pixel by a null transform. It has been computed as average of 5 hours footage for the first scenario sequences.



(a) (b)

Fig. 7. In figure (a) the performance in frames per second (fps) versus the number of features for each of the methods used in the comparison is shown. It is interesting to notice, even from the computational time required by each method plotted in figure (b), how the proposed method works as fast as the LKT and faster than the other two. In particular, it is worth noting that only the proposed algorithm is able to operate with a number of feature lower than 40.

mance of the proposed system. For such purposes, we considered the following aspects: the accuracy of the detection, its rate as well as the missed detection. In 4 hours of footage, 2352 objects were acquired and 816 were selected to be tracked. The average detection rate, as shown in Table 6, is 97.3%. The acuracy is within 3.2 and 1.4 pixels respecitvely in x and y from the ground-truth centroid. Such values represent the 4% of the width and the height of the objects inside the image. Moreover, we like to state that the detection accuracy is stable regardless the total running time of the system or the age (i.e. time in the scene) of each object.
It is interesting to see, from the comparison presented in Table 6, how only the performance of the proposed does not decrease when executed in real-time. In particular, we have that the method proposed by Araki performs very well off-line while in the real-time mode its performance worsen to an unacceptable level. The same effect can be seen also in the performance of the other

	Off-line		Real- Time	
	Detected	False	Detected	False
Proposed	97.3%	0.8%	97.3%	0.8%
Araki	98.3%	0.7%	1.1%	98.2%
Tommasini	82.4%	3.2%	68.9%	22.3%
Lucas-Kanade-Tomasi	75.8%	3.8%	71.5%	19.7%

Table 6
The table shows the results of the proposed comparison. The values of detection, computed as number of detected objects over number of total objects on all frames, and the false detection, computed as percentage of frames in which there is a false detection, are considered. In particular, such values as been computed by running the algorithm off-line therefore leaving them all the time needed for each frame and simulating a real-time acquisition by dropping all the frames occurred in the time needed for the computation

techniques even if with minor entity. This effect is due to the fact that in real time, dropping frames, the magnitude of the transform for the alignment becomes bigger and the distortions cannot be modelled anymore by the chosen transform.

5.4 Limits of the system

A critical situation is represented by objects moving on an almost uniform background (e.g. a wall). In such a context, the system cannot be applied because no features can be extracted from static objects. This limit is not so restrictive if the proposed method is adopted in a video surveillance system for the monitoring of outdoor environments where, generally, an uniform background is not common.

Regarding the tracking subsystem, it imposes a limit on the objects speed. Tests have highlighted that $60Km/h$ is the upper speed limit to allow the system to obtain a reliable tracking of objects moving at the distance of about $50 - 100$ meters from the camera. Once again, this limit is not to restrictive since adopting the system for monitoring city roads, the law speed limit is lower than the speed limit imposed by the system.
Moreover, limits have been identified on complex scenes presenting more than 6 vehicles performing unusual trajectories. In particular, we tested the system asking to track a car going to stop and occluded by other vehicles making u-turns, changing direction or proceeding backwards. As result we obtained that the system continued to track the wrong vehicle. Basically, such a problem can be solved by exploiting colour and shape information in the tracking process.

Fig. 8. Frames of test sequences with different zoom levels. The last three rows are characterized by a lot of objects that occlude the moving objects. The last row, in particular, shows a representative frame of a sequence where two vehicles cross their trajectories.

6 Conclusions

In this paper, we have proposed an active (visual-based) surveillance system able to compensate background changes due to the camera motion, detect and track mobile objects in real scenes. The innovations are: (a) estimating in a robust way the displacement occurring among two consecutive frames and (b) speeding up the task for the maintenance of a reliable set of features.
Precisely, the proposed method obtains a good displacement estimation as demonstrated by the low percentage of errors achieved. Moreover, the use of a map of features for the maintenance of a reliable feature set, allows the system to operate in real-time thanks to a lower complexity of the extraction process for new features. Finally, the tracking module allows a continuous tracking of an object with a moderate speed, also when briefly occluded by other objects. The results allowed us to adopt the proposed system as part of a video-surveillance system whose principal task is to monitor a wide area. Future works will investigate the possibility to extend the system to automatically change the camera parameters (focus, iris, zoom) and to track mobile objects by maintaining them at the centre of the image with a constant size, optimal

focus and luminosity.

References

[1] C. Regazzoni, V. Ramesh, and G. Foresti, "Special issue on video communications, processing, and understanding for third generation surveillance systems," *Proceeding of the IEEE*, vol. 89, no. 10, Oct. 2001.

[2] Y. Y. L. Davis, R. Chellapa and Q. Zheng, "Visual surveillance and monitoring of human and vehicular activity," in *In Proceedings of DARPA97 Image Understanding Workshop*, 1997, pp. 19–27.

[3] R. Howarthand and H. Buxton, "Visual surveillance monitoring and watching," in *In European Conference on Computer Vision*, Cambridge, UK, Apr.15-18 1996, pp. 321–334.

[4] T. Kanade, R. Collins, A. Lipton, P.Burt, and L.Wixson, "Advances in cooperative multisensor video surveillance," in *In Proceedings of DARPA Image Understanding Workshop*, vol. 1, Nov. 1998, pp. 3–24.

[5] G. Foresti, P. Mahonen, and C. Regazzoni, *Multimedia Video-Based Surveillance Systems: from User Requirements to Research Solutions.* Kluwer Academic Publisher, Sept. 2000.

[6] C. Kim and J. Hwang, "Object-based video abstraction for video surveillance systems," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1128–1138, Dec. 2002.

[7] H. Veeraraghavan, O. Masoud, and N. Papanikolopoulos, "Computer vision algorithms for intersection monitoring," *IEEE Trans. Intell. Transport. Syst.*, vol. 4, no. 2, pp. 78–89, Jun 2003.

[8] P. Kumar, S. Ranganath, H. Weimin, and K. Sengupta, "Framework for real-time behavior interpretation from traffic video," *IEEE Trans. Intell. Transport. Syst.*, vol. 6, no. 1, pp. 43–43, Mar 2005.

[9] D. Koller, K. Daniilidis, and H. H. Nagel, "Model-based object tracking in monocular sequences of road traffic scenes," *International Journal of Computer Vision*, vol. 10, pp. 257–281, 1993.

[10] J. Ben-Arie, W. Zhiqian, P.Pandit, and S. Rajaram, "Human activity recognition using multidimensional indexing," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 8, pp. 1091–1104, Aug. 2002.

[11] S. Dockstader and A. Tekalp, "Multiple camera tracking of interacting and occluded human motion," *Proceedings of the IEEE*, vol. 89, no. 10, pp. 1441–1455, Oct. 2001.

[12] J. Gluckman and S. Nayar, "Ego-motion and omnidirectional camera," in *IEEE International Conference on Computer Vision*, Bombay, India, Jan. 3–5 1998, pp. 999–1005.

[13] T. Gandhi and M. M. Trivedi, "Parametric ego-motion estimation for vehicle surround analysis using an omnidirectional camera." *Machine Vision and Applications*, vol. 16, no. 2, pp. 85–95, 2005.

[14] A. Davison, I. D. Reid, and D. Murray, "The active camera as a projective pointing device," in *Proceedings of* 6^{th} *British Machine Vision Conference*, Birmingham UK, Sept.11-14 1999, pp. 11–14.

[15] D. Murray and A. Basu, "Motion tracking with an active camera," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, no. 5, pp. 449–454, May 1994.

[16] R. Okada, "Object tracking based on optical flow and depth," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Washington D.C., Dec. 1996, pp. 565–571.

[17] M. Irani and P. Anandan, "A unified approach to moving object detection in 2d and 3d scenes," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, no. 6, pp. 577–589, June 1998.

[18] S. Araki, T. Matsuoka, N. Yokoya, and H. Takemura, "Real-time tracking of multiple moving object contours in a moving camera image sequences," *IEICE Transaction on Information and Systems*, vol. E83-D, no. 7, pp. 1583–1591, July 2000.

[19] R. Radke, S. Andra, O. Al-Kofai, and B. Roysam, "Image change detection algorithms: A systematic survey," *IEEE Trans. Image Processing*, vol. 14, no. 3, pp. 294–307, Mar. 2005.

[20] S. Haritaoglu, D. Harwood, and L. Davis, "w^4: Real-time surveillance of people and their activities," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 809–830, August 2000.

[21] T. Matsuyama and N. Ukita, "Real-time multitarget tracking by a cooperative distributed vision system," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1136–1149, 2002.

[22] Y. Yiming, K. Bennet, J. Tsotsos, and E. Harley, "Tracking a person with pre-recorded image database and a pan, tilt and zoom camera," *Machine Vision and Applications*, vol. 12, no. 1, pp. 32–43, Jan. 2000.

[23] B. Zitova and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, pp. 977–1000, 2003.

[24] J. Shi and C. Tomasi, "Good features to track," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Seattle WA, June 20-24 1994, pp. 593–600.

[25] T. Tommasini, A. Fusiello, E. Trucco, and V.Roberto, "Making good features track better," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 23-25 1998, pp. 178–183.

[26] Y. Sugaya and K. Kanatani, "Outlier removal for motion tracking by subspace separation," *IEICE Transactions on Information and Systems*, vol. E86-D, no. 6, pp. 1095–1102, 2003.

[27] Y. Sugaya and K.Kanatani, "Extracting moving objects from a moving camera video sequence," in 10^{th} *Symposium on Sensing and Image Information*, Yokohama, Japan, June 9-11 2004, pp. 279–284.

[28] G. Guo, C. Dyer, and Z. Zang, "Linear combination representation for outlier detection in motion tracking"," in *IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 2, no. 2, San Diego, CA, USA, June 20-25 2005, pp. 274–281.

[29] K. Kanatani, "Camera rotation invariance of image characteristics," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Miami Beach (FL), June 1986.

[30] G. Foresti and C. Micheloni, "A robust feature tracker for active surveillance of outdoor scenes," *Electronic Letters on Computer Vision and Image Analysis*, vol. 1, no. 1, pp. 21–36, 2003.

[31] C. Tomasi and T. Kanade, "Detection and tracking of point features," Carnegie Mellon University, Pittsburgh PA, Tech. Rep. CMU-CS-91-132, 1991.

[32] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram," *Graphical Models and Image Processing*, vol. 29, pp. 273–285, 1985.

[33] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the* 7^{th} *International Joint Conference on Artificial Intelligence*, Vancouver, Aug. 1981, pp. 674–679.

[34] *Open Source Computer Vision Library.* [Online]. Available: http://www.intel.com/research/mrl/research/opencv/

[35] P. Rosin, "Thresholding for change detection," in *Proceedings of IEEE International Conference on Computer Vision*, Bombay India, 1998, pp. 274–279.

[36] J. Fan, D. Yau, A. Elmagarmid, and W. Aref, "Automatic image segmentation by integrating color-edge extraction and seeded region growing," *IEEE Trans. Image Processing*, vol. 10, no. 10, pp. 1454–1466, Oct. 2001.

[37] A. Arsenio and J. Victor, "Active monocular tracking with temporal integration of visual cues," in 9^{th} *Portoguese Conference on Pattern Recognition*, Coimbra, Portugal, Mar. 20-21 1997.

[38] M. Kohler, *3D Image Analisys and Synthesis.* Springler Verlag, 1996, ch. Vision Based Remote Control in Intelligent Home Enviroments, pp. 147–154.

[39] C. Schmid, R. Mohrand, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 151–172, 2000.

[40] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa, "A system for video surveillance and monitoring," Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-00-12, May 2000.