A NEW FEATURE CLUSTERING METHOD FOR OBJECT DETECTION WITH AN ACTIVE CAMERA

C. Micheloni, G.L. Foresti, F. Alberti

Department of Computer Science, University of Udine, ITALY

ABSTRACT

Feature based methods for ego-motion estimation are widely used in computer vision but they must deal with errors in feature tracking. In this paper, we propose a robust real-time method for ego-motion estimation by assuming an affine motion of the background from the previous to the current frame. A new clustering technique is applied on image's sub areas to select in a fast and reliable way three features for the affine transform computation. The previous frame after being warped according to the computed affine transform is processed with the current frame by a change detection method in order to detect mobile objects. Results are presented in the context of a visual-based surveillance system for monitoring outdoor environments.

1. INTRODUCTION

The motion detection of an object from image sequences is one of the most studied problems within computer vision. When the image stream is acquired by a moving camera, motion detection is generally considered a difficult task [1], [2], [3]. In fact, when comparing two consecutive frames of a sequence, differences in pixel intensities occur in the whole image, since ego-motion of the camera causes an apparent motion of the static background. Two main techniques could be adopted: a) frame background [4] and b) frame-by-frame [2],[3]. Using a PTZ camera and maintaining an high frame rate, the background compensation could be relaxed to the estimation of a vector $\mathbf{d}$ such that the following expression holds [5]:

$$I(\mathbf{x}+\mathbf{d}, i+1) = I(\mathbf{x}, i) \quad (1)$$

where $I(\mathbf{x}+\mathbf{d}, i+1)$ represents the actual frame compensated by the displacement vector $\mathbf{d}$ and $I(\mathbf{x}, i)$ the previous frame.

Murray and Basu [3] proposed a technique based on the estimation of the background motion by reading the camera pan and tilt angles from the device. To overcome the related rotation problems, Irani and Anandan [2] address the problem by estimating a "dominant" 8-parameters transformation. In particular, the Sum of Squared Differences (SSD) measure is minimized to compute an affine transform for registering two consecutive frames and therefore to cancel the rotational component of the camera motion. This approach must deal with the problem of outliers and with the high dimension of the linear system needed to compute the affine transform. Araki et al. [1] track some feature points extracted from the background and estimate the parameters of an affine transformation from previous frame to actual frame. From the set of tracked features the Least Median of Squares (LMedS) method is adopted to select iteratively a subset of three features used for the computation of the best transform. Even though this technique is really powerful, it needs a DSP architecture to be executed in real-time since also bad tracked features can be considered in the estimation process.

To reject bad tracked feature the thresholding technique proposed by Shi and Tomasi[6] can be adopted. In order to apply an adaptive thresholding, Tommasini et al. [7] propose the statistical X84 rule that, based on the computation of the MAD, identify and reject the outliers. Unfortunately, to reduce motion estimation error, these methods require an high number of features (about 200) by limiting the performance of the tracking system.

In this paper, we propose a new system able to estimate the affine transform in an accurate and real-time way. This is achieved by introducing a clustering method which creates different clusters characterized by features with the same local displacement. Such clusters are also used to reject badly tracked features which imply to adopt a method to introduce new good features to track. Extracting features in the whole image implies a large computational effort compromising the real-time properties of the system. Adopting a technique based on a map of the features [8], the system is able to reduce the computational time by extracting features only in the new areas introduced by the camera movement and by maintaining information about the position of the others.

2. METHOD DESCRIPTION

As shown in Fig. 1, the proposed method is included in a frame by frame motion detection system. Let $I(\mathbf{x}, i)$ be the

0-7803-8554-3/04/$20.00 ©2004 IEEE.

i-th frame of the image sequence, let $Map_i(\mathbf{x})$ be the map [8], computed at the time instant i, that contains positions information about well trackable features for the i-th frame. At each iteration the current Map_i is used to build the set of

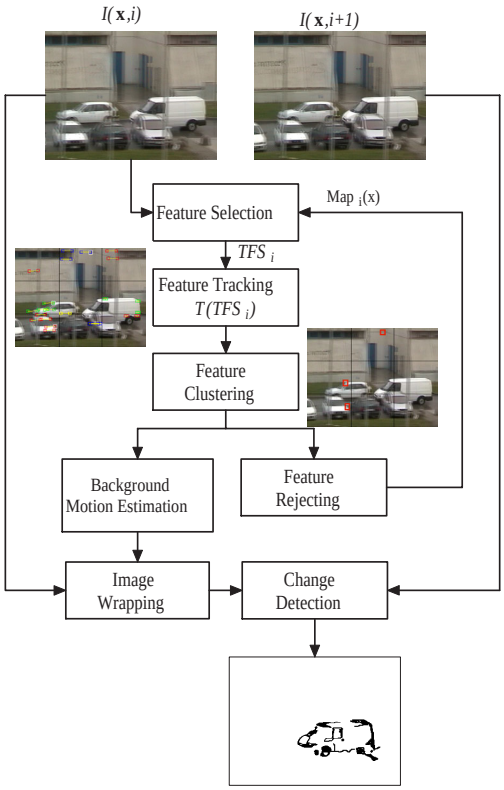

Fig. 1. General architecture of the proposed system

well trackable features TFS_i. The adopted feature tracking algorithm [9] is then applied on TFS_i to obtain the local displacement $\mathbf{d}_j$ of each feature f_j in TFS_i. The feature clustering step takes as input the list of local displacements $D_{i+1} = \{\mathbf{d}_j | f_j \in T(TFS_i)\}$ and uses a new clustering property to estimate the affine transform for the background alignment. In particular, the clustering technique is applied on N windows of the image to determine for each one the displacement vector $\mathbf{d_j}$ characterizing the background motion in the window j.

Therefore, the background compensation operation wraps the previous frame according to the affine transform. Hence, a change detection operation [10], is applied between two consecutive frames. The output is a binary image whose white pixels correspond to points belonging to moving objects while black pixels represent background points in the scene. Affine transform is also used to perform the updating of the feature map. This process involves three different steps: (a) rejection of bad tracked features, (b) updating feature position and (c) introduction of features belonging to new image regions. Feature rejection is a mandatory task since some errors occurs during the feature tracking phase due to distortion introduced by camera motion. All the features belonging to a window j whose displacement differs from the vector $\mathbf{d_j}$, estimated by the proposed method, are considered bad tracked features and eliminated from TFS_i. A problem can occur after this operation: the number of features could become too low for a robust tracking. The proposed solution consists on the repopulation of the TFS by introducing new well trackable features.

3. FEATURE CLUSTERING

Once the TFS is built, the feature tracking algorithm T is applied on it [9]. The output, $T(TFS_i)$ is a correspondence relation between the selected features in the current frame $I(\mathbf{x}, i+1)$ and in the previous frame $I(\mathbf{x}, i)$. Unfortunately, often it occurs that some features of TFS are not well tracked due to noise, occlusions etc. In order to face this problem it is necessary to distinguish features well tracked from the others. Analyzing the behavior of tracked features we have derived the following feature clustering property:

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

This property tells us that the features belonging to objects in the background, if correctly tracked, have the same local displacement. Unfortunately, this property is true only if we allow a translational movement of the camera which therefore would allow to adopt the model proposed by Kanatani [5].

Since the proposed solution has been studied to solve the alignment problem for every camera motion, the strategy adopted requires to divide the image area in N windows. This technique allows to adopt the clustering property within each window since therein the motion can be relaxed to the computation of a displacement vector.

Let $\mathbf{d}_i$ be the displacement of feature f_i, for each window w_l $l = 1 \ldots N$ we define:

$$C_{w_l}(\mathbf{d}_c) = \{f_i \in TFS \cap w_l | \mathbf{d}_i = \mathbf{d}_c\} \qquad (2)$$

the cluster of all features of window w_l having a displacement equal to the vector $\mathbf{d}_c$. Let $C_{w_i} = \{C_{w_l}(\mathbf{d}_c)\}$ be the set of all clusters generated by the tracking algorithm for each window w_l; for each of them it is defined a reliability factor RF as indicator of the closeness of its displacement to the background one. Let RF be defined as follows:

$$RF(C_{w_l}(\mathbf{d}_c)) = \frac{\sum_{f_i \in C_{w_l}(\mathbf{d}_c)} E_{f_i}}{|C_{w_l}(\mathbf{d}_c)|^2} \qquad (3)$$

where E_{f_i} is the residual of the feature between its position in the previous and current frame [6]. Accordingly to the

definition of the RF factor and to the definition of the clustering property, the displacement vector $\mathbf{d_l}$ of a window w_l is computed as follows:

$$\mathbf{d_l} = \mathbf{d}_i |\ RF(C_{w_l}(\mathbf{d}_i)) = \min\{RF(C_{w_l}(\mathbf{d}_j))|\ C_{w_l}(\mathbf{d}_j) \in C_{w_l}\} \quad (4)$$

where $C_{w_l}(\mathbf{d}_i)$ is the cluster selected to represent the background displacement of the window l.
To compute the affine transform $\hat{A}$, we have studied an algorithm based on the following steps:

- Order the selected clusters $C_{w_l}(d_i)$ on the basis of the RF factor;
- Select three clusters having minor RF factor;
- For each cluster, select the feature having minimum ratio $\frac{E_{f_i}}{\lambda}$ where λ is the feature's eigenvalue computed as in [6];
- Use such selected features to compute the affine transform $\hat{A}$;

After having estimated the affine transform for the image alignment, the previous image is warped according to it and the TFS is updated. This process is performed in two steps. At the first step all features not belonging to the selected cluster are labelled as not well tracked, rejected and eliminated from the TFS. At the second step, the TFS is repopulate by the features selection module. See [8] for more details on this step.

4. EXPERIMENTAL RESULTS

The sequences used for experiments are acquired by a Cohu 3812 CCD camera mounted on a Pan-Tilt Unit and are characterized by images of 384x288 pixels. The adopted processor is an Athlon 1.2 GHz, the maximum number of features in the TFS is 50 and the number of windows N is 9.
A compensation error **CE** has been considered to represent a quality measure of the alignment method. It gives a measure of the number of pixel miss-aligned and their weight to the change detection operation. It is defined as follows:

$$CE = \frac{\sum_i g_i log p_i}{\sum_i log p_i} \quad (5)$$

where g_i are the gray values of the image of differenced and p_i the number of pixels with intensity value g_i. Robustness of the system needs two more parameters: the number of good features rejected (GFR) as the percentage of features rejected that would be considered well trackable and (b) the number of bad features maintained (BFM) as percentage of features not rejected that would be considered not well trackable:

$$GFR = \frac{|\{\mathbf{x}|\mathbf{x}\in TFS_i \wedge \mathbf{x}\notin TFS_{i+1} \wedge G(I(\mathbf{x},i+1))=true\}|}{|\{\mathbf{x}|\mathbf{x} \in TFS_i \wedge \mathbf{x} \notin TFS_{i+1}\}|} \quad (6)$$

$$BFM = \frac{|\{\mathbf{x}|\mathbf{x}\in TFS_{i+1} \wedge G(I(\mathbf{x},i+1))=false\}|}{|TFS_{i+1}|} \quad (7)$$

where $|S|$ indicates the cardinality of the set S.

	Scenarios			
	First		Second	
	Mean μ	Max	Mean μ	Max
CE	12.34	18.61	13.08	19.44
GFR %	4.11	33.3	5.32	33.3
BFM %	1.01	12.5	1.12	22.2

Table 1. Scenarios Evaluation

	Compensation Error		Performance
	Mean μ	Max	Frame/sec
Proposed	12.54	19.44	23
Araki	15.49	17.32	2
Tommasini	25.77	49.45	18
Kanade	28.89	63.02	20

Table 2. Comparison results

A first scenario has been considered to evaluate the system over sequences without any moving object. The second scenario tests the system on sequences containing one or more moving objects. Both scenarios include sequences acquired in a variety of whether conditions: sunny, cloudy and rainy day. In Table 1, the values of the considered parameters for the sequences belonging to the first scenario are shown. The CE parameter shows a good estimation of the background motion. The mean time required to update the map and select the new features is $0.024s$, allowing the system to operate at the frame rate of about 23 frame/s.
In Table 1 the values of the evaluation parameters for the sequences belonging to the second scenario are shown. It demonstrates a great tolerance to the presence of moving objects in the scene. In these conditions the frame rate is still equal to 23 frame/s.

To give a better evaluation of the system, the proposed alignment technique has been compared with the methods proposed by Araki [1], Tomasi and Kanade [9] and by Tommasini et al. [7]. As shown in Table 2 two types of comparisons have been taken into account: alignment accuracy and computational effort. Regarding the alignment accuracy it is worth noting how the proposed solution acts well as the Araki's algorithm and really better then those proposed by Kanade *et al.* and Tommasini *et al.*. Looking at the computational effort we can say how the proposed method is faster than all the others. Then, the proposed method is accurate as the Araki's one but really faster. In Figure 2, some results of a test sequences are shown.

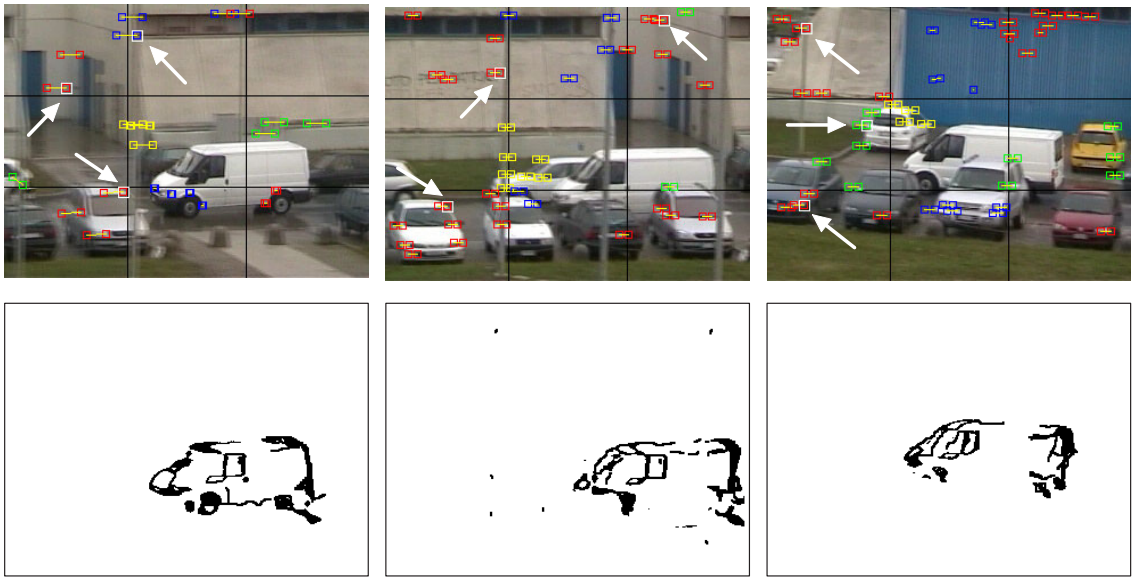

Fig. 2. Results of the feature clustering technique adopted to reduce the ego-motion effects. Top row shows the result of the tracking algorithm with highlighted, by the arrows, the features selected for the affine transform computation. Bottom row shows the results of the alignment process.

5. CONCLUSIONS

In this paper, we have proposed a system able to compensate background changes due to the camera motion and to detect mobile objects in outdoor scenes. The innovative parts cover two main problems: (a) estimating in a robust way the background motion occurring between consecutive frames and (b) speeding up the task by adopting a feature clustering technique over different windows. Experimental results have been performed on different sequences, each one acquired with different parameters (zoom, tilt and pan camera speed) and increasing complexity. Over 10^4 frames computed, the proposed method obtains a good accuracy in the alignment process by executing it in real-time. The proposed approach has been compared with other feature tracking methods [1, 7, 9] and the obtained results show a significant improvement.

Acknowledgments

This work was partially supported by the Italian Ministry of University and Scientific Research within the framework of the project "Distributed systems for multisensor recognition with augmented perception for ambient security and customization " (2002-2004).

6. REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] K. Kanatani, "Camera rotation invariance of image characteristics," in *IEEE Comput. Vision. and Pat. Rec.*, Miami Beach (FL), June 1986.

[6] J. Shi and C. Tomasi, "Good features to track," in *IEEE International Conference on Computer Vision and Pattern Recognition*, Seattle WA, June20-24 1994, pp. 593–600.

[7] T. Tommasini, A. Fusiello, E. Trucco, and V.Roberto, "Making good features track better," in *IEEE Comput. Vision. and Pat. Rec.*, Santa Barbara CA, June23-25 1998, pp. 178–183.

[8] C. Micheloni and G.L. Foresti, "Fast good features selection for wide area monitoring," in *IEEE International Conference on Advanced Video and Signal Based Surveillance*, Miami, FL, July21-22 2003, pp. 271–276.

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

[10] P. Rosin, "Thresholding for change detection," in *IEEE Int. Conf. on Comput. Vision*, Bombay India, 1998, pp. 274–279.