A SIMPLE AND ROBUST METHOD FOR MOVING TARGET TRACKING

Gaetano Baldini*, Paola Campadelli, Dario Cozzi, Raffaella Lanzarotti

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano
Via Comelico, 39/41 20135 Milano, Italy
{campadelli,lanzarotti}@dsi.unimi.it

* CESI – Via Rubattino,24
Segrate, Milano – Italy
gaetano.baldini@cesi.it

ABSTRACT

Good motion detection and tracking algorithms are the basic elements of any video-surveillance system. In this paper we present a simple and robust method for tracking unclassified moving targets. Classification of moving objects into dangerous intrusions or not is made only after having tracked them for some frames. Experimental results showing the feasibility of the method are given.

KEY WORDS
motion-detection, tracking, video-surveillance

1. INTRODUCTION

The objective of this paper[1] is to present a simple and robust procedure for a video-surveillance system which has to monitor a wide outdoor site where the presence of people and vehicles should be considered a potentially dangerous event. When such an event occurs, the images which document it are sent to a control room far away from the monitored site. Since the video-surveillance system is designed for outdoor scenes, it has to deal reliably with lighting changes, different weather conditions and very low light level situations (night-time).

There are some interesting projets designed for outdoor surveillance tasks **[1][2][3]**. The goal of two of them is more complex than ours, the other is quite similar. In particular in **[1]** the attention is devoted to distinguishing people from all the other moving objects in the scene, and tracking individual people moving isolated or in a small group. People detection is done through a shape analysis of the foreground regions, tracking is done with the help of a motion model and an appearance model. In **[2]** the goal is to develop a system which not only detects and classifies objects but tries to learn common activity patterns from observations during a long period. The tracking of foreground regions is done with a set of Kalman filters; tracked objects are represented by their position speed, direction and size and they are given as inputs to a hierarchical unsupervised classifier. In **[3]** a system for detecting moving objects and classifying them into predefined categories (human, vehicle, other) is described. The foreground regions, detected by a temporal difference technique, are classified on the base of some simple shape parameters (area and dispersedness) and used as a training template for the tracker. Our experiments **[4]** based on simple motion detection algorithms, like temporal difference or extraction of moving edges **[5]**, fail in producing good results in some critical situations. For this reason we have looked for a more robust motion detection algorithm although more computational expensive.

Most of the recently proposed techniques for motion detection are based on probabilistic methods for background subtraction **[1][2][6][7]**; we have chosen the one proposed in **[6]** because it is rather sensitive. We propose a tracking method which is simple and effective and allows us to eliminate detections of background movements (snow, hail, swaying, leaves…). Each foreground region is tracked matching the data found by the motion detection algorithm with those found by a block-matching technique. Only after having tracked a moving object for some frames (at least three), we can reasonably be sure that it is not a false target; therefore the image can be sent to a control room. We are studying a way to classify objects in order to send to the control room only the images of dangerous ones.

The system proposed works on gray level images and consists of a motion detection algorithm, a tracking algorithm and a neural classifier. In this paper the motion detection algorithm is quickly reported and the tracking

[1] This work was done under contract *P1323* between *CESI* and the *University of Milan*; the work is part of the activity, said *System Research*, assigned to *CESI* by Italian Ministry of Industry, Commerce and Handicraft.

algorithm is described in detail. The algorithm has been experimented on many image sequences taken during the day in very different weather conditions and some sequences taken during the night. It never happened that a non-moving object has been tracked.

2. THE MOTION DETECTION ALGORITHM

Since we were looking for a motion detection algorithm with high sensitivity, we have chosen the background subtraction process presented in **[6].** The background is modelled considering each pixel along an image sequence as an independent statistical process whose intensity distribution might change quickly. The density function of the distribution at any moment is estimated by a non-parametric technique. Given a recent sample of intensity values $x_1,\ldots,x_N$, the value of the density at a point x_t is estimated as

$$\Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N} K(x_t - x_i)$$

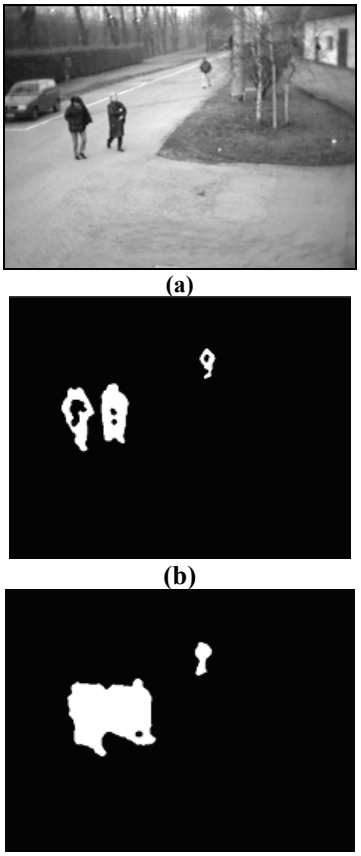
where K is a kernel function. The authors of the above mentioned paper suggest adopting a gaussian for each colour channel. Since we want the system to work also during the night or in other low light level conditions, we apply the method to gray level image. So doing, we obtain:

$$\Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x_t - x_i)^2}{2\sigma^2}}$$

A pixel is considered a *foreground pixel* if $\Pr(x_t)$ is lower than a predefined threshold *T*.

In order to estimate the parameter σ, the median of the distribution built with the absolute differences of successive samples ($|x_i - x_{i-1}|$, $1 \le i \le N$) is calculated.

We set the sample frequency to 30 frames/sec since lower values made the moving objects not well distinguished from the background. **Figure 1** clarifies the point: **Figure 1.a** shows the last image of a sample sequence; **Figure 1.b** and **Figure 1.c** show the results obtained taking the samples at a frequency of 30 frames/sec and 10 frames/sec respectively. It can be seen that in **Figure 1.c** the bigger motion region includes undesirable background pixels, those where the moving objects have "just been". With the frequency of 30 frames/sec we have experimentally determined the sample length *N* and the threshold *T* and set them to 15 and 0.015 respectively. Lower values of *N* do not allow to detect motion properly, whereas higher values do not improve the result significantly.



(a)

(b)

(c)

Figure 1: Effect of the sample frequency on the estimate of foreground pixels. (a): original image; (b) and (c): motion regions found with a sample frequency of 30 frames/sec and 10 frames/sec respectively.

With the given choice of the parameters the estimate of the foreground pixels is very good. Very small regions, probably due to noise, are eliminated with a median filter of size 3×3; the other foreground pixels are segmented into *motion regions* by a connected component algorithm. It may happen that two moving elements, which are very close to each other, are detected as a single motion region (see **Figure 2.a** and **2.b**) or that the same moving object is divided into two or more parts. This last problem is dealt with in the next paragraph.

(a)

(b)

Figure 2: Two moving elements detected as a single motion region. (a): original image; (b) motion regions.

Each motion region is enclosed in a bounding rectangle, a *motion window*. To establish the correspondence of motion windows between frames, a tracking procedure is designed. It allows to eliminate false detections, due both to noise and to small movements in the scene background, and to follow the trajectories of moving objects. Once a motion window has been tracked for a certain number of frames, a recognition algorithm classifies it as representing a dangerous object or not.
In the following section we describe the tracking algorithm.

3. THE TRACKING ALGORITHM

Given *n* motion windows at frame *t,* the corresponding motion windows at frame *t+1* have to be found. Of course, it may happen that some objects disappear from the field of view and others enter into it.
The search of *corresponding windows* is done in two steps:

1. For each *motion window* at time t, the window with the greatest correlation is searched in frame *t+1*.
2. Each window with the highest correlation, *matching window*, found in frame *t+1* has to be validated as a region corresponding to a moving object in the same frame.

Given a window, we look for its closest translate in frame *t+1*, assuming that no transformation except translation can occur between two successive images.
More precisely, let *R(c)* be a window centred in *c* in frame *t*, let *d* be a displacement vector whose components vary in the range *[-W, +W]*, and let *R′(c+d)* be a window of the same size of *R* in frame *t+1*. We choose, among all the possible *d*, the *R′(c+d)* that maximize the function

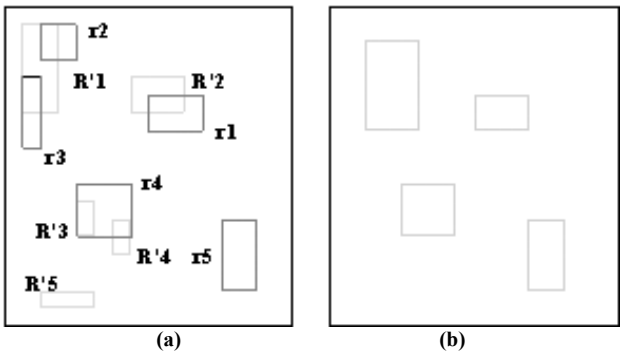
$$Sim(R(c), R'(c+d)) = F(R, R')$$

where

$$F(R, R') = -\sum_{p} (R(p) - R'(p))^2$$

with *p* spanning the image windows.

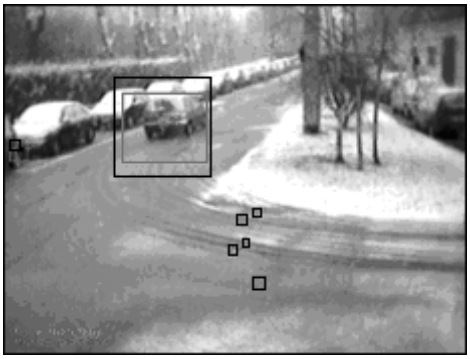
We apply this *block matching* algorithm to each motion window. Since the block matching finds a match even if a moving object disappears, it is necessary to establish which of the *matching windows* really correspond to moving objects. To this end we compare the windows found by the motion detection algorithm with those found by the matching algorithm on the same frame. We say that *two windows correspond* if the centre of one of the two is contained into the other. In other word, we consider that windows projected ahead by block matching and those found by the motion detection represent the same moving object if they superimpose significantly in the same image.
The disappearing or entering of moving objects into the scene are easily recognizable because the correspondence criterion fails. It may also happen that the motion detection algorithm divides an object into different moving regions, or fuses two or more moving objects very close to each other in a unique region (see **Figure 2.b**). While the correspondence criterion allows to solve the first situation, it is helpless in correcting the second one.
The different possibilities which can be encountered when looking for corresponding windows are sketched in **Figure 3.a.** Let us suppose that the rectangles $r_1, r_2, \ldots, r_5$ (dark gray) are motion windows found at frame *t* and that the rectangles R′1,R′2,…,R′5 (light gray) are the matching windows found by the block matching algorithm in the same frame. After verifying the correspondence criterion described above, the windows of **Figure 3.b** are obtained in the following way. A new motion window is maintained as it is (see r5). In case of one-to-one correspondence (R′2 and r1) or when many block windows correspond to one motion window (R′3, R′4 and r4) the motion window is maintained. This is because it is the one that projected ahead may better identify the moving object in the next frame. In case one block window corresponds to many motion ones (R'1 and r2, r3) the new window is obtained taking the minimum rectangle including all the others. Observe that R′5 is not maintained in **Figure 3.b** since it is a window found by block matching but not by the motion detection algorithm.



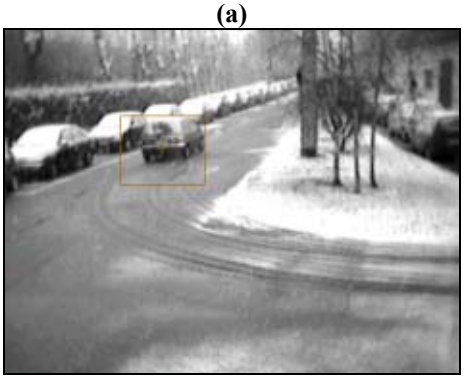
(a) (b)

Figure 3: Sketch of the rule used to build the tracking windows

We call *tracking windows* the windows in **Figure 3.b**, they are the ones to be projected ahead and to be compared with the motion windows in the next frame.
The following example shows how the method described for determining the tracking windows allows to correct situations where the motion detection algorithm finds movements which are not to be considered because they are due to background elements (snow, swaying, leaves…). **Figure 4.a** shows a car and some big snowflakes detected as matching windows (black rectangles on the figure). The tracking algorithm maintains only the window corresponding to the car (**Figure 4.b**) since it is the only one that corresponds to a motion window (gray rectangle on the **Figure 4.a**).



(a)

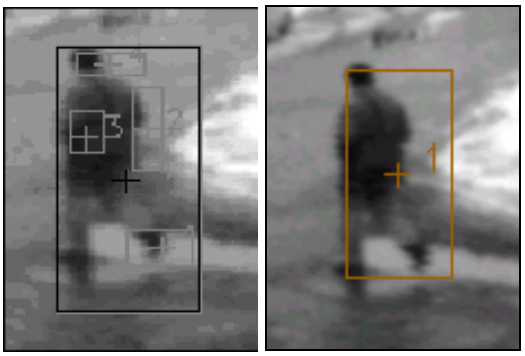


(b)

Figure 4: The snowflakes are not tracked

It may also happen that a moving object is divided into different motion regions in a certain frame of a sequence, and that the tracking algorithm can maintain a window corresponding to the whole object if, at least in a frame, the motion detection algorithm has detected it as a unique region (see **Figure 5.a** and **Figure 5.b**).

The name, size and position of each tracking window is maintained in a list; a window which can not be tracked for at least three frames is cancelled from the list.
The trajectory of each moving object is reconstructed using the centroids of the tracking window as control points of a Bezier curve.

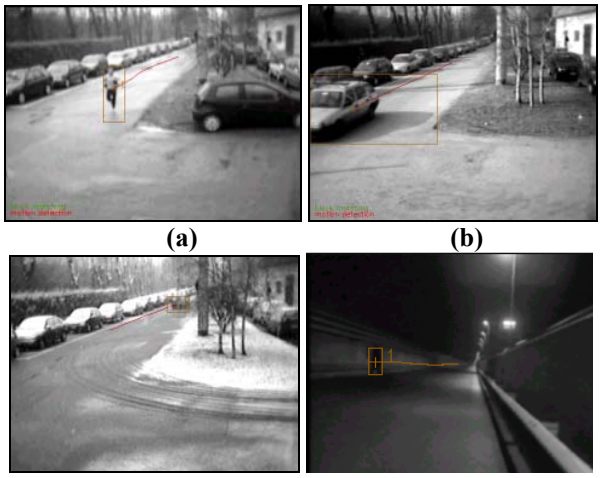


(a) **(b)**

Figure5: A person divided into different motion regions (gray rectangles) is tracked as a unique object since the matching window (black rectangle) corresponds to all the motion regions.

4. EXPERIMENTAL RESULTS AND CONCLUSIONS

The algorithm has been experimented on many image sequences taken during the day in very different weather conditions (sunny, rainy, snowy, windy days) and some sequences taken during the night. It never happened that a non moving-object has been tracked; seldom a group of people has been considered a single object.
Four examples of the trajectories determined in different weather and illumination conditions are shown in **Figure 6.a**, **6.b**, **6.c**, **6.d**.



(a) **(b)**

(c) **(d)**

Figure 6: Tracking in very different weather and illumination conditions

The tracking procedure works even when a target is, or becomes, very small and when it is partially occluded.
We are now studying the problem of classifying tracked windows. Preliminary experiments made with a neural network trained by standard back propagation to classify vehicles and human beings as dangerous objects, and

anything else (see **Figure 7**) as not dangerous, are giving good results.

Figure 7: Ducks

The algorithms are implemented in IDL (Interactive Data Language), an interpreted language. Currently for 320×240 resolution gray level images, the computation time is about three seconds on Pentium III-800MHz, 128MB of RAM and Windows'98 OS.
We are facing the problem of making the system operate in real time, this requires optimizing the code (using C language) and running it on a higher performance platform.

REFERENCES

[1] I. Haritaoglu, D.Harwood, L.S. Davis – "W^4: Real-Time Surveillance of People and Their Activities" *–IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 22, N.8, August 2000*

[2] C. Stauffer, W.Eric, L. Grimson – "Learning Pattern of Activity Using Real-Time Tracking" *–IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol.22, N.8, August 2000*

[3] A.J.Lipton, H. Fujiyoshi, R.S. Patil – " Moving Target Classification and Tracking from Real-time Video" - *Proc. IEEE Workshop Application of Computer Vision, 1998*

[4] D. Cozzi – "Realizzazione di un sistema per il rilevamento di intrusioni in un'area video-sorvegliata"- Tesi di Laurea – Università degli Studi di Milano, Dipartimento di Scienze dell'Informazione A.A. 2001-2002

[5] A. Makarov, J.M. Vesin, M.Kunt – "Intrusion Detection Using Extraction of Moving Edges" – *12^{th} IARP International Conference on Pattern Recognition, Jerusalem, Israel – October 9-13, 1994*

[6] A. Elgammal, D. Harwood, L. Davis – " Non-parametric Model for Background Subtraction" – *Proceedings of the 6^{th} European Conference on Computer Vision, pages 751-767, 2000*

[7] N. Friedman, S. Russell – "Image segmentation in video sequences: a probabilistic approach" – *Proceedings of the 13^{th} Conference on Uncertaintyin Artificial Intelligence, pages 1-3, 1997*