Object Segmentation Using Background Modelling and Cascaded Change Detection

Luís F. Teixeira, Jaime S. Cardoso, Luís Corte-Real
INESC Porto, Faculdade de Engenharia, Universidade do Porto
Rua Dr. Roberto Frias, 378; 4200 - 465 Porto, Portugal
Email: {luis.f.teixeira, jaime.cardoso, lreal}@inescporto.pt

Abstract— **The automatic extraction and analysis of visual information is becoming generalised. The first step in this processing chain is usually separating or segmenting the captured visual scene in individual objects. Obtaining a perceptually correct segmentation is however a cumbersome task. Moreover, typical applications relying on object segmentation, such as visual surveillance, introduce two additional requirements: (1) it should represent only a small fraction of the total amount of processing time and (2) real-time overall processing. We propose a technique that tackles these problems using a cascade of change detection tests, including noise-induced, illumination variation and structural changes. An objective comparison of common pixel-wise modelling methods is first done. A cost-based partition-distance between segmentation masks is introduced and used to evaluate the methods. Both the mixture of Gaussians and the kernel density estimation are used as a base to detect structural changes in the proposed algorithm. Experimental results show that the cascade technique consistently outperforms the base methods, without additional post-processing and without additional processing overheads.**

Index Terms— **foreground segmentation, object detection, background modelling and subtraction, objective comparison of segmentations, cost-based partition-distance, cascaded change detection, visual surveillance**

I. Introduction

Automatic processing of large amounts of information has been a recurrent research topic over the last years, and important results have been achieved. Visual information processing is not an exception and due to its much improved capabilities, applications like search engines based in visual terms are becoming widespread. A very important initial step toward efficient processing is the separation of a complex scene in its composing objects or visual elements. Each object can then be separately further analysed, identified, classified, etc. This separation or segmentation is a very complex problem since, to attain results comparable to human performed segmentations, semantic or high-level *a priori* knowledge is required. For typical real-time applications oriented to the analysis of visual scenes in order to identify events and actions – such as intelligent surveillance systems and human-machine interface systems – simplifications are needed. For these, motion is a key factor aiding the segmentation process. Objects that are moving or performing some action, often called *foreground objects*, need to be separated from the other elements in the scene, the *background*.

Probably due to its simplicity, the most common approach for discriminating a moving object from the background is *background subtraction*. The rationale is the subtraction of the current image from a reference image, which is somehow acquired in a step prior to subtraction. Non-changing segments of the image are considered as being part of the background, whereas the foreground consists of the changing segments – including moving and new objects. However, if the reference is not modelled or updated adequately this technique can be highly susceptible to environment conditions like illumination changes. For example, a straightforward way of acquiring a reference image would be using the previous "history" by obtaining a *background model* based on the statistical representation of the previous N frames, for example a pixel-wise average image. After estimating the reference image, the segmentation can be obtained from an efficient thresholded subtraction operation. This simple approach, although efficient, may not perform well in real-world, non-controlled environments. Changes in illumination conditions and dynamic behaviour in the background may cause unacceptable rates of false positives. To achieve robust background modelling, techniques that can better adapt to dynamic behaviour are needed. Ideally, the performance should not depend on the camera placement, nor should it be sensible to what happens in its visual field or to lighting effects. It should also be capable of dealing with movement through cluttered areas, objects overlapping in the visual field, shadows, lighting changes, effects of moving objects in the scene, slow-moving objects and objects being introduced or removed from the scene. However, it is also important to stress that this operation is often required to perform as fast as possible, since it is usually the first step in the processing chain. Overly complex modelling schemes may reveal themselves unfeasible despite performing at low error rates.

A proposed classification of existing algorithms for background modelling divides them in predictive and non-predictive methods. Predictive methods model the scene as a time series and develop a dynamic model to recover the current input based on past observations. Usually

This paper is based on "Cascaded change detection for foreground segmentation," by Luís F. Teixeira and Luís Corte-Real, which appeared in the Proceedings of the IEEE Workshop on Motion and Video Computing (WMVC 2007), Austin, Texas, USA, February 2007. © 2007 IEEE.

Kalman filters [1] [2] are employed to update slow and gradual changes in the background. In general, these methods are mainly applicable to backgrounds consisting of stationary objects. On the other hand, non-predictive methods for background modelling do not consider the order of input observations and build a probabilistic representation (p.d.f.) of the observations at a particular pixel. These are by far the most common methods and many different proposals and adaptations can be found in the literature. In [3] a unimodal distribution was proposed – a Gaussian distribution is used to model the background. If each pixel resulted from a particular surface under particular lighting, a single Gaussian would be sufficient to model the pixel value accounting for acquisition noise. Moreover, if only lighting changed over time, a single, adaptive Gaussian model per pixel would be sufficient. In practice this does not happen and for that purpose a model based in the mixture of a fixed number of Gaussians distributions has been frequently used [4] [5]. In [6], a non-parametric model is proposed, where a kernel-based function is used to represent each pixel's colour distribution. The kernel-based distribution is a generalization of the mixture of Gaussians which does not require parameter estimation. In [7], a similar approach is followed, where the distribution of temporal variations in colour at each pixel is used to model the background. Motion-based adaptive kernel density estimation is also proposed in [8]. More recent approaches to background modelling include the principal features approximation [9] that considers only the more relevant features to create a model and the mean-shift method [10] [11], which was also previously applied to image segmentation.

All of the previous methods are based in pixel-wise background modelling. Other techniques combine temporal and spatial modelling. In [12], a mixture model (Gaussians or Laplacians) is used to represent the distributions of background differences for static background points. A Markov random field (MRF) model incorporates the spatial coherence for robust foreground segmentation. Another approach to detect moving objects is to extract groups of motion, either by accumulating consistent flows in terms of direction over successive frames [13] or by using layered approaches to fit a collection of motion models to the image data [14] [15]. Recently, an approach that defines foreground objects as clusters of pixels salient with respect to both motion and colour was presented [16].

Additionally other authors propose a different approach that employs information about the object's structure to segment them. In [17], the contour of moving objects is estimated by fusing motion with colour segmentation and edge detection. Active contours were employed in [18] and, besides edge information, prior models on the image intensity values inside and outside the contour were also proposed [19]. These methods estimate the object's contour by minimizing a global cost function and, although robust, the estimates are achieved at a high computational cost. The main drawbacks are the complexity and the need to generally assume some prior knowledge in order to avoid ill-posed problems. However, this knowledge is not always available. Moreover, the priors assumed by the techniques can be unsuitable to the problem in hands. More recent work in this line of thought includes the use of layered models [20] and rigid objects modelling [21].

This paper is organized as follows. In Section II an overview of some background modelling and subtraction algorithms is presented as well as an objective comparison between them. The comparison between segmentations produced by each of the algorithms is done using a metric introduced also in this section. The metric is a generalization of the distance between partitions. Section III describes the cascaded change detection algorithm and adapts it to two commonly used methods for background modelling: mixture of Gaussians and kernel density estimation. Experimental results are presented in Section IV. Finally, conclusions are drawn in Section V.

II. Background modelling and subtraction overview and comparison

In this section we present a small overview of some of background and modelling algorithms mentioned in the previous section. A representative set of state of the art techniques were implemented and tested. A comparison was done and the conclusions are presented. The Running Average background modelling algorithm is used as a base performance index. To compare the segmentation results of each technique, an objective metric is also proposed.

A. Overview

1) Running Average (RAvg) – The background can be modelled as the average of the previous frames but, in order to avoid expensive memory requirements, this average is approximated by an adaptive filter with a learning rate α. Each background pixel value at position (i, j) and time instant t is given by:

$$B_{(i,j)}(t) = \alpha I_{(i,j)}(t) + (1 - \alpha) B_{(i,j)}(t-1)$$

Foreground is then estimated using a thresholded subtraction of the current frame and the estimated background. This technique is probably the most naive but has a very simple and very fast implementation. The results are therefore far from good in particular with complex backgrounds. Since we are considering only a static representation of the background to perform the subtraction, whenever some kind of dynamic behaviour in the background happens, it will be incorrectly classified as foreground. Nevertheless, the running average should represent the base performance for these types of algorithms. All tests were performed with a threshold T_{RAvg} of 15.

2) Mixture of Gaussians (MoG) – Instead of estimating the background representation directly, another and more effective approach is to estimate a background model that can predict the behaviour in each pixel, using the pixel's "history". By estimating the background probability density function (p.d.f.) we are able to do just that. Assuming that any structural changes affecting the value of the pixel are caused by several processes, each modelled by a Gaussian, we can therefore define the probability of observing its value as:

$$P(\upsilon_t) = \sum_{k=1}^{K} P(G_k)P(\upsilon_t|G_k) = \sum_{k=1}^{K} \omega_k \cdot \eta(\upsilon_t, \mu_k, \sigma_k) \tag{1}$$

where G_k is the k-th Gaussian of K distributions, ω_k, μ_k and σ_k are, respectively, an estimate of the weight, the mean value and the variance of the k-th Gaussian in the mixture; η is the normal density function. Moreover, it can be easily shown [5] that, given the current colour vector υ_t in a pixel, the probability that the pixel belongs to the background is:

$$P(B|\upsilon_t) = \frac{\sum_{k=1}^{K} P(\upsilon_t|G_k)P(G_k)P(B|G_k)}{\sum_{k=1}^{K} P(\upsilon_t|G_k)P(G_k)} \tag{2}$$

If $P(B|\upsilon_t) > T_{MoG}$ the pixel is estimated as being part of the background. However, two density estimation problems are left to resolve: firstly, estimating the distribution of all observations, within a period of time, at each pixel location using the Gaussian mixture in equation (1), which provides estimates of both $P(G_k)$ and $P(\upsilon_t|G_k)$; and secondly, evaluating how likely each Gaussian in the mixture represents the background, i.e., $P(B|G_k)$. To accomplish the first estimation, in [4] an online K-means approximation is proposed in order to model pixel variation over time by a mixture of Gaussians, as given by equation (2). It uses a fixed learning rate to update each Gaussian's parameters over time and a Gaussian substitution algorithm whenever no match is possible. However, using a fixed learning rate can often result in slow convergence. Following the same rationale, and in order to improve the convergence speed, in [5] it is proposed an adaptive learning rate schedule for each Gaussian defined by a parameter α; we will be using this last approach. The estimation of $P(B|G_k)$ is based on application-specific heuristics [5].

The background image representation can be defined as the expected value of the background process. Thus, the background pixel at (i,j) and time t is defined by $E[\upsilon_{i,j,t}|B]$ which is evaluated by a weighted average of the Gaussian means.

$$E[\upsilon_{i,j,t}|B] = \frac{\sum_{k=1}^{K} \mu_k P(B|G_k)P(G_k)}{\sum_{k=1}^{K} P(B|G_k)P(G_k)} \tag{3}$$

The tests with MoG were done with the following parameters: $K = 3$, $\alpha = 0.005$ and $T_{MoG} = 0.05$.

3) Kernel Density Estimation (KDE) – It is possible to approximate each background's pixel p.d.f. by the histogram of the most recent values classified as background. This approach has however some problems namely, being the histogram a step function, the p.d.f. modelling can reveal itself erroneous. A non-parametric model based on Kernel Density Estimation (KDE) is proposed in [6]. KDE guarantees a smoothed, continuous representation of the histogram. The background p.d.f. is given as a sum of Gaussian kernels centred in the most recent N background values:

$$P(\upsilon_t) = \frac{1}{N}\sum_{k=1}^{N} \eta(\upsilon_t - \upsilon_k, \Sigma_k) \tag{4}$$

Even if background values are not known, unclassified sample data can be used instead. This inaccuracy will be recovered along model updates. Given 4, the pixel with the colour vector υ_t is classified as foreground if $P(\upsilon_t) < T_{KDE}$, where T is a global threshold. An important issue in KDE is the estimation of Σ_k – the kernel bandwidth. In [6], it is considered a diagonal matrix for simplicity and each variance is estimated in the time domain by analysing the set of differences between two consecutive values

Model update consists in selectively updating the vector of the previous N background values. The model proposed in [6] also considers the use of two concurrent similar models, one for long-term and the other for short-term memory. In addition, spatial correlation is taken into consideration by the model. However, we will not be considering both these modifications since we are comparing pixel-wise modelling techniques. These types of considerations are transversal to all algorithms we are evaluating. The tests executed with this algorithm used the following parameters: $N = 50$ and $T_{KDE} = 10^{-6}$.

4) Principal Features (PF) – More recently, other approaches were proposed to estimate the background p.d.f.. In [9], the background is represented at each pixel by the most frequent features, or *principal features*.

The classification is done using a Bayesian framework, and it is shown that a pixel represented by $\boldsymbol{\upsilon}$ is classified as belonging to the background if:

$$2P(\boldsymbol{\upsilon}|B)P(B) > P(\boldsymbol{\upsilon}) \tag{5}$$

Otherwise, it is classified as belonging to the foreground. We need however to know *a priori* or estimate the probabilities $P(\boldsymbol{\upsilon}|B)$, $P(B)$ and $P(\boldsymbol{\upsilon})$. As stated previously, one way to estimate these probabilities is to use a histogram of features. The important contribution of [9] is that it proposes that these probabilities can be estimated using solely the most representative features in the histogram, given that these can represent the background effectively. Therefore, for a proper selection of features, there would be a small value N of features (the principal features) that can approximate well the background by $\sum_{k=1}^{N} P(\upsilon_k|B)$.

The learning and update process is done using a table of statistics for the possible principal features of the background. The update of estimated probabilities through time is done using a simple adaptive filter according to the type of change that occurred (gradual or "once-off").

Note also that the algorithm proposed in [9] uses several types of features, namely: spectral, spatial and temporal features. For this comparison we will be using only the spectral features, i.e. colour information. Otherwise, the results for this algorithm would be biased. The tests executed with principal features used the following parameters: $\alpha = \beta = 0.04$ (rate for probability and background learning, respectively), $M = 50$, $N = 20$ and $M1 = 0.75$ (for "once-off" detection).

5) Mean Shift (MS) – Finally, yet another way of estimating the background p.d.f. is to use the mean-shift [10] [11] which is an iterative gradient-ascent method to detect modes of a multimodal distribution and their covariance matrix. The only parameter needed is the bandwidth range that is application-specific. The mean shift algorithm states that, for a given set of points $x_i, i = 1, \dots, n$, the mean shift vector in the one-dimensional case can be expressed as:

$$m(x) = \frac{\sum_{i=1}^{n} x_i g\left(\frac{x-x_i}{h}\right)^2}{\sum_{i=1}^{n} g\left(\frac{x-x_i}{h}\right)^2} - x$$

where x is an arbitrary point in the data space, h is a positive value called the analysis bandwidth and $g(u)$ is a bounded support function, first derivative of another bounded support function, $k(u)$, or kernel profile. It can be proven that, for a kernel with a convex and monotonically decreasing profile, the iterative procedure $x^{l+1} = m(x^l) + x^l$ converges. In [11] some optimizations are introduced in order to reduce processing time, namely histogram-base mean shift computation – the mean shift vector is calculated with respect to the number N of histogram bins. This was the method implemented and tested.

All points $x_i, i = 1, \dots, t_u$ belonging to a mode will converge to the same point, the mode centre, or mean μ_u. Moreover, if we assume Gaussian modes, for each feature – in this case the components of the colour vector $\boldsymbol{v}$ – the p.d.f consists of a weight sum of the U modes modelled by a Gaussian distribution. A threshold test can simply be applied to the estimated p.d.f:

$$\sum_{u=1}^{U} \prod_{f=1}^{F} \omega_{(u,f)} \eta(x_f, \mu_{(u,f)}, \sigma^2_{(u,f)}) < T_{MS}$$

Note that we are assuming that the features $f, f = 1, \dots, F$ are independent. The weights $\omega_{(u,f)}$ also need to be estimated, and are generally defined by heuristics [11]. If the probability estimated for a given pixel value $\boldsymbol{v}$ is smaller than the threshold T, the pixel is classified as foreground. The mean shift algorithm was tested with the parameters: $N = 50$, $h = 3$ and $T_{MS} = 10^{-20}$.

B. Metric to compare foreground segmentations

When working with object-based spatial segmentation of video, the major objective is to design an algorithm that produces appropriate segmentation results for the particular goals of the application being addressed. The demonstration of the usefulness of a particular algorithm entails necessarily a comparative analysis of the algorithm against similar algorithms. And the fair assessment requires suitable metrics providing a meaningful and objective measure of performance.

A unifying model for the comparison of image segmentations was proposed recently in the literature [22], [23]. Here, we recover the general framework and instantiate a new metric, particularly adapted for the application at hand. At the core of the framework is the intersection-

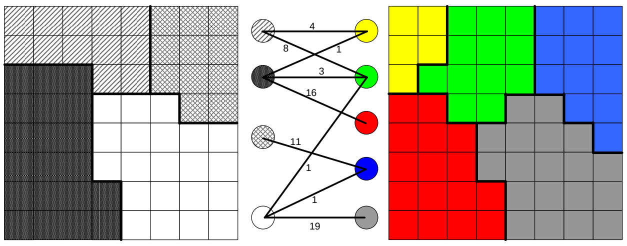

Figure 1. Intersection-graph for two segmentations. The weights shown correspond to the number of pixels in the intersection.

graph between two segmentations, defined as the bipartite graph with one node for each region of the segmentations. Two nodes are connected by an undirected, weighted edge if and only if those two regions intersect each other. Figure 1 exemplifies such setting. The intersection-graph associated with two image segmentations can now be used as a factory of indices of similarity between partitions. The partition-distance [22] has been defined as the problem of finding a maximum weighted matching in the intersection-graph. The sum of the weights of the unmatched edges on this matching process provides the distance between both segmentations. And the pixels corresponding to these unmatched edges constitute a informative error mask.

The weight assigned to an edge should express the importance of the corresponding intersection. The simplest way is to assign the area of the intersection to the weight of each edge. But this formulation is not necessarily the one better capturing the perceived quality of a segmentation. In particular, all pixels in the same intersection of two regions are being equally weighted. However, to accommodate human perception, the different error contributions should be weighted according to their visual relevance. Therefore, it is, arguably, more sensible that the cost of erring a pixel increases with pixel distance to the object border.

The foregoing argument motivates the introduction of the cost-based partition-distance, d^c_{sym}, as a generalization of the partition-distance. Start by computing the distance of each pixel to the object border in both segmentations, d_1 and d_2. Define a monotonically increasing cost function on these two distances, $C(d_1, d_2)$. Different laws can be considered, such as linear, exponential or logarithmic. A suitable strategy is to set $C(d_1, d_2) =$

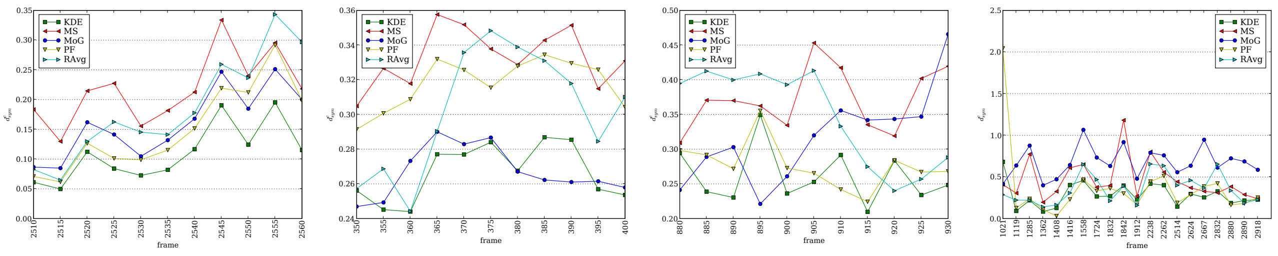

Figure 2. Evolution of d^c_{sym} for the sequences SW, SH, OD and BR (from left to right) using different methods for background modelling – kernel density estimation, mean-shift, mixture of gaussians, principal features and running average.

$\max(d_1, d_2)$ or $C(d_1, d_2) = 2^{\max(d_1,d_2)}$. Finally, the weight of an edge will be the mere sum of the individual costs of the pixels in the intersection. Note that setting $C(d_1, d_2) = 1$ results in edges weighted by the area of the intersection. The cost-based partition-distance will be the sum of the weights of the unmatched edges on the matching process that follows. The cost-based partition-distance will penalize thick discrepancies between two segmentations, favouring thin, along the borders, differences. This metric can also be seen as generalization of the perceptual spatial measure proposed in [24].

It is possible to confirm that the cost-based partition-distance still enjoys of most of the useful properties of the original partition-distance [22]. Most notably, the cost-based partition-distance is still non-negative (being zero iff the two partitions coincide), symmetric, and transitive. It is, therefore, a metric. The transitive property is especially significant in the context of comparing more than two algorithms. It conveys the desirable behaviour that if segmentation A is similar to segmentation B and segmentation B is similar to segmentation C, then segmentation A is similar to segmentation C.

C. Comparative test

A comparative study of background modelling techniques was previously presented in [25], however this study consists of a theoretical comparison of several algorithms and no qualitative tests are presented. In order to get a better understanding of the algorithms, we tested them in several sequences. The results for some of the test set sequences are presented next – refer to section IV for more details on the test sequences. All tests were executed using only colour vectors as features; the YUV colour space was used.

Table I summarizes the results obtained for each algorithm in each sequence; for each algorithm-sequence combination the average distance to the “ground-truth” and the throughput in frames per second (fps) are presented. No post-processing was employed on each algorithm's output segmentations. Figure 2 shows the evolution of the distance over time in the SW, SH, OD and BR sequences. All algorithms were implemented by the authors and the tests were performed in a Pentium 4 3.4GHz with 1GB of RAM.

Results show that KDE perform better than the other methods. The principal features and the mixture of Gaussians modelling methods follow closely. Note that the results obtained with MoG in the BR sequence are severely degraded by the initial state which includes foreground objects. Since the analysed time window is not sufficient for the model to recover, the results are continuously affected by it. Also, note that the mean shift approach processing time is extremely high, invalidating its use for real-time applications. On the other hand the MoG method has a considerably better processing frame rate than the other methods and is therefore better suited for real-time applications.

TABLE I.
AVERAGE d^c_{sym} AND FRAMES PER SECOND FOR EACH METHOD OVER THE EVALUATED FRAMES OF EACH SEQUENCE.

		SW	SH	OD	BR	FT
RAvg	d^c_{sym}	0.185	0.302	0.347	0.357	0.336
	fps	134	152	156	571	526
MoG	d^c_{sym}	0.160	0.267	0.317	0.678	0.307
	fps	6.7	6.1	6.7	31.5	29.2
KDE	d^c_{sym}	0.109	0.267	0.261	0.285	0.093
	fps	1.3	1.1	1.4	5.6	7.5
PF	d^c_{sym}	0.150	0.318	0.276	0.362	0.126
	fps	2.9	2.2	2.9	13.7	14.7
MS	d^c_{sym}	0.218	0.333	0.372	0.460	0.157
	fps	0.04	0.04	0.05	0.2	0.2

In summary, despite having the less average distance to the “ground-truth”, KDE's throughput makes it less interesting than MoG, which has the best relation performance/complexity. Moreover, the segmentations produced by the MoG method are prone to be improved even further since they are less fragmented than the ones produced by the other methods – this will become more evident in section IV.

III. CASCADED CHANGE DETECTION

Consider a typical visual surveillance scene being capture by a static camera; most of the background pixel values change slowly in time and can be caused by phenomena of different nature. If it is possible to model these independently, better results can be achieved. Before detailing the proposed algorithm let us first consider the following: when no structural change occurs, the difference between a pixel value, represented by a colour vector $\boldsymbol{v}$, in the current frame $\mathcal{F}_c$ and a reference frame $\mathcal{F}_r$ can essentially result from two factors – illumination variation or noise. Illumination variation can be accounted by a positive multiplicative factor k which modulates the signal, while noise can be accounted by a superimposed vector $\boldsymbol{v}_N$, modelled by a Gaussian or Laplacian distribution – Figure 3 represents this in a two-dimensional space.

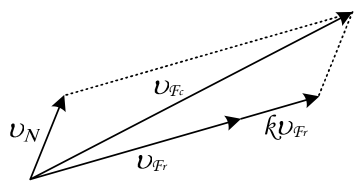

Figure 3. Effects of illumination variation and noise over a reference colour vector – the latter is modelled by a positive multiplicative factor which modulates the signal, while the former is modelled by a superimposed vector, modelled by a Gaussian or Laplacian distribution

Considering that the sophomore cause can be successfully eliminated, we are left with the first. Hence, when a pixel change results solely from illumination variation, its colour vector is necessarily collinear with the reference colour vector and a simple test can be used to identify illumination variation-induced changes. Therefore, we will first address how we can effectively remove typical noise introduced by the capture process in order to guarantee colour vector collinearity.

A. Identification of noise-induced changes

Assuming that we know the reference frame $\mathcal{F}_r$, we will first address how we can effectively remove typical noise introduced by the capture process. For that purpose we use a method proposed in [26] and previously used for simple background subtraction [27]. It states that it is possible to assess what is the probability that a value at a given position, in a given image, is due to noise instead of other causes when compared to another image. It is assumed that the additive noise affecting each image results from a Gaussian process with mean μ_N and standard deviation σ_N. Also, noise affecting successive images in the sequence is considered as uncorrelated. The standard deviation σ_N can be obtained by computing the statistics of the difference $d_{(i,j)}$ for each pixel (i,j) between the reference image and the current image. Now, consider a window W^n containing n pixels around the pixel under evaluation with $\Delta^2_{(i,j)} = \sum_{(k,l)\in W^n_{i,j}} d^2_{(k,l)}$. It can be shown that the corresponding random variable Δ^2 follows a χ^2 distribution. Given the hypothesis H_0 that $\Delta^2_{(i,j)}$ results from noise and not from other factor, the probability that hypothesis H_0 is satisfied is given by:

$$P(\Delta^2 > \Delta^2_{(i,j)}|H_0) = \frac{\Gamma(\frac{n}{2}, \frac{\Delta^2_{(i,j)}}{2\sigma_c^2})}{\Gamma(\frac{n}{2})} \tag{6}$$

with $\sigma_c^2 = 2\sigma_N^2$ and where $\Gamma(n/2)$ is the Gamma function. The choice for the window size n must take into consideration the trade-off between noise sensitivity and foreground edge definition. Nevertheless, all experiments were performed using a window size of $n = 25$. When the estimated probability in equation (6) is smaller than a threshold T_N we consider that H_0 is not satisfied at the pixel position (i,j).

Whereas for pixels that validate the hypothesis H_0 we guarantee that changes were originated solely by camera noise, for others we can safely assume that the any effect of noise is negligible when compared to any other change. In other words, if a pixel's colour vector is being modified by illumination variation and no structural change, we have $\boldsymbol{v}_{\mathcal{F}_c} \simeq k\boldsymbol{v}_{\mathcal{F}_r}$.

This test defines a first set of pixels that can potentially be part of the foreground because all pixels that satisfy H_0 are necessarily part of the background and are marked as such for the current frame; all others need further analysis.

B. Identification of illumination variation-induced changes

After discarding noise-induced changes, a simple collinearity test is performed. As previously stated, with this test any modification introduced by illumination variation is discarded. The test consists in evaluating the angle between the current pixel colour vector $\boldsymbol{v}^c$ and the reference colour vector $\boldsymbol{v}^r$.

$$\cos\theta = \frac{\boldsymbol{v}^c \cdot \boldsymbol{v}^r}{\|\boldsymbol{v}^c\|\,\|\boldsymbol{v}^r\|} \tag{7}$$

If $\cos\theta$ is smaller than a threshold T_I very close to 1, the vectors are not considered to be collinear and the test is not validated. In practice, this test can become unstable and to overcome this problem we consider a second threshold in the noise-identification test; the second threshold usually is much smaller than T_N. The identification of illumination variation-induced changes is therefore only applied only to pixels that fall between both these thresholds.

The set of potential foreground pixels is refined by marking as background the pixels that validate this second test. Pixels that have a probability less than the second threshold in the previous test are maintained as foreground.

C. Identification of dynamic background behaviour

The final estimation of whether a pixel is part of the background can be performed only in the pixels that do not validate the statistical (6) nor the collinearity (7) tests, resulting in a considerable reduction in execution time of the algorithm. This is especially true for typical surveillance streams where the relevant moving objects occupy a small fraction of the entire field of view. Another positive effect of the proposed modification is the drastic reduction of small artefacts which often need to be removed in typical modelling by applying morphological filtering (e.g. connected operators).

To model and identify the dynamic background behaviour we will be using pixel-wise background estimation approaches, namely the ones presented in section II. The head-to-head comparison revealed that the Mixture of

Gaussians (MoG) modelling approach had the best performance/efficiency relation. We will therefore be using mainly MoG to identify dynamic background behaviour. In this case, if the probability defined by equation (2) for a given pixel is larger than a threshold T_S, the pixel is considered to be part of the background. Also, background representation defined by the MoG model and described by (3) is used as the reference image $\mathcal{F}_r$. Figure 4(a) summarizes the algorithm in pseudo-code, which until the end of the article will be called as cascaded mixture of Gaussians (CMoG).

```
Input: $\alpha$, K, $T_N$, $T_I$ and $T_S$
begin
   for t = 0, ...
      get current frame Fc
      foreach pixel in Fc
         // classification
         if (6) > $T_N$
            classify pixel as background
         else if (6) > $10^{-10}T_N$
            if (7) > $T_I$
               classify pixel as background
         else if (2) > $T_S$
               classify pixel as background
         else
               classify pixel as foreground
         // update background
         update $\omega_k$, $\mu_k$ and $\sigma_K$ in (2)
         update reference using (3)
      end foreach
   t = t + 1
   end for
end
```

(a) CMoG

```
Input: N, $T_N$, $T_I$ and $T_S$
begin
   for t = 0, ...
      get current frame Fc
      foreach pixel in Fc
         // classification
         if (6) > $T_N$
            classify pixel as background
         else if (6) > $10^{-10}T_N$
            if (7) > $T_I$
               classify pixel as background
         else if (4) > $T_S$
               classify pixel as background
         else
               classify pixel as foreground
         // update background
         update reference using running average
         update previous background sample
      end foreach
   t = t + 1
   end for
end
```

(b) CKDE

Figure 4. Proposed algorithm in pseudo-code. Both are very similar with the exception of the way dynamic background behaviour is modelled.

Alongside MoG, cascaded kernel density estimation (KDE) was also tested. For KDE, if the probability defined by equation (4) for a given pixel is larger than a threshold T_S, that pixel is classified as background. The background representation can be defined as the weighted average of the N most recent background values. The weights are defined, for each previous background and for each pixel, by the probability in (4) when a sample is tested against the other $N-1$ previous backgrounds. For efficiency reasons a simple running average of the incoming frames was used to estimate a background without significant errors in the final foreground estimation. In accordance with CMoG, the KDE-based cascaded algorithm will be named CKDE hereafter. Figure 4(b) summarizes the CKDE algorithm.

D. Summary

The proposed algorithm can be seen as a series of different techniques resulting in the refinement of the segmentation provided by the previous as shown in Figure 5. First we use the statistical test (6) to determine the set $\mathcal{S}_a$ of pixels that are identified as being changed by any phenomenon other than noise – in our model a structural change or illumination variation. Then, on that set of pixels a simple collinearity test (7) is performed in order to assert that the modification was not due to some modification in illumination conditions, removing any pixel that are, resulting in a new set $\mathcal{S}_b$ of candidate pixels. Finally, the set is further refined eliminating any structural change that resulted from repetitive dynamic behaviour of the background using pixel-wise background estimation. The result is the final set of pixels $\mathcal{S}_c$ which are labelled as belonging to the foreground, i.e., any relevant moving object. Although the different classification tests happen in a cascade configuration, some interaction happens between them. Namely, if the change results from camera noise and not from other factor, the model is updated with the current background value instead of the new frame value. This way we effectively reduce model error by only introducing modifications to the model when any other phenomenon than noise changes the pixel value.

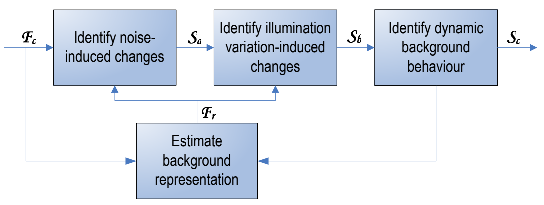
$\mathcal{F}_c$
Identify noise-induced changes
$\mathcal{S}_a$
Identify illumination variation-induced changes
$\mathcal{S}_b$
Identify dynamic background behaviour
$\mathcal{S}_c$
$\mathcal{F}_r$
Estimate background representation

Figure 5. Block representation of the proposed cascaded algorithm. The modules are connected in cascade configuration with the exception of the background estimation feedback.

E. Implementation

The proposed algorithm as well as the other background modelling and subtraction algorithms described in section II were implemented in the MarsyasX framework[1]. MarsyasX is an open-source cross-modal analysis framework. It follows a dataflow architecture where complex network of processing objects can be assembled to form systems that can handle multiple and different types of multimedia flows with expressiveness and efficiency. Is is based in Marsyas [28] which originally was dedicated to audio analysis. The modularity and flexibility of MarsyasX was an important factor to create the background modelling and subtraction testing framework[2].

IV. Results

The data set used to evaluate the proposed method performance consists of 12 sequences which were manually segmented in selected frames[3]. The first sequence, called shopping (SH) shows a view of a shopping corridor and is one of the test case scenarios made publicly available by the EC Funded CAVIAR project/IST 2001 37540. The scene consists of people walking, browsing the stores' displays or waiting for others. It has stable illumination conditions, except for a small portion in the right side of the field of view. However, hard shadows and reflections in the floor and in the display's glass are present. The second sequence, labelled outdoor (OD) shows an outdoor scene with several people passing along the camera field of view and is available in the MPEG-7 test set (results

[1]Information about MarsyasX can be found in `http://telecom.inescporto.pt/~lfpt/MarsyasX`

[2]Implementation and details are available in the BGMS page at `http://telecom.inescporto.pt/~lfpt/BGMS`

[3]Test set is available at the BGMS page.

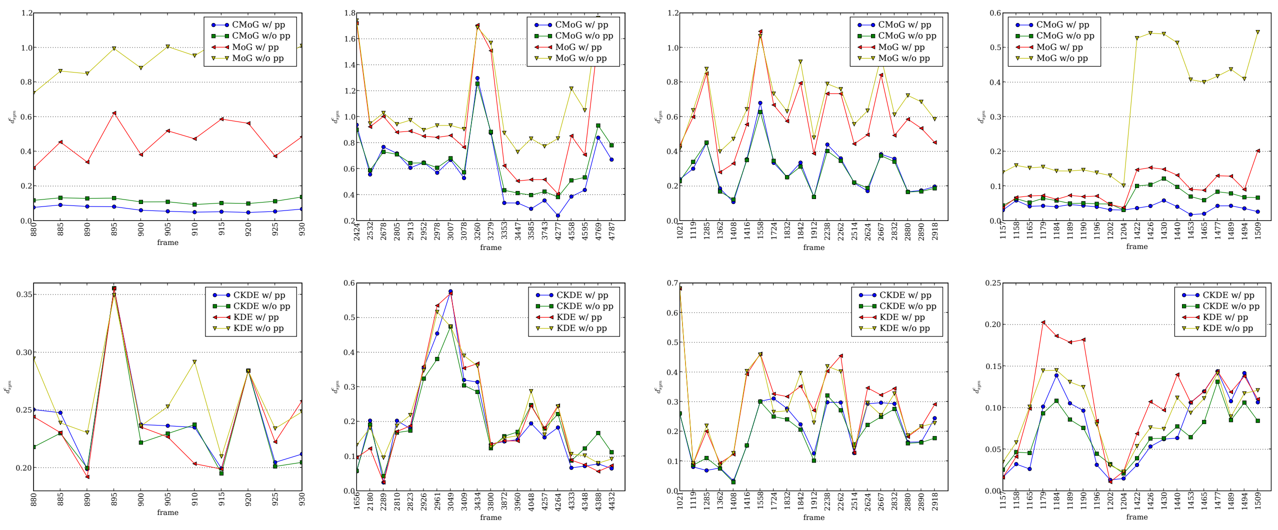

Figure 6. Evolution of d^c_{sym} for the sequences OD, AP, BR and FT (from left to right). In the first row the results refer to MoG and CMoG implementations, with and without post-processing; the second row KDE and CKDE are compared the same way.

are presented for stream A). The sequence has some noise and although the illumination conditions are fairly stable, the background presents significant vegetation swing. The speedway (SW) sequence was captured from a bridge over a speedway and is also available in the MPEG-7 test set. It shows different sorts of vehicles moving in both directions. Overall, it is the most stable stream regarding background changes but some relevant shadows are present. The following frames were manually segmented by visual inspection: 350, 355,... and 400 in the SH sequence; 880, 885,... and 930 in the OD sequence; and 2510, 2515,... and 2560 in the SW sequence. Besides these three sequences, the test set also includes nine openly available segmented sequences, namely: meeting room with moving curtain (MR), campus with wavering tree branches (CAM), lobby in an office building with switching on/off lights (LB), shopping centre (SC), hall of an airport (AP), restaurant (BR), subway station (SS), water surface (WS), and fountain (FT). All sequences from this data set present challenging scenes with considerable background activity – mainly in MR, CAM, WS and FT – and sudden changes of illumination conditions are also present – with special difficulty in LB. More details about these sequences can be found in [9].

Results are presented next and a comparison is first drawn between CMoG and MoG as well as CKDE and KDE. An overall comparison between all is done last. Also, note that all experiments used the YUV colour space as input features.

A. CMoG

The CMoG configuration consisted of a MoG model composed by a mixture of 3 Gaussians, a learning rate α of 0.005 and a threshold T_S of 0.05. For the statistical test it was used a significance threshold T_N of 10^{-4}. Finally, for the collinearity test it was used a threshold T_I of 0.995. All thresholds were found through empirical testing to be fairly stable and can be used without modification for typical real-world scenes. The MoG algorithm was tested with the same common parameters. Moreover, to reduce the implementation complexity, for MoG implementation, it was considered a diagonal covariance matrix.

Figure 6 shows in the first row the evolution of the cost-based partition distance (d^c_{sym}) for four sequences of the test set, namely OD, AP, BR and FT. It is clear that the proposed method outperforms the mixture of Gaussians modelling method in all test sequences. Even without post-processing the CMoG method performs significantly better than the regular MoG with post-processing. In fact, for the test set sequences, CMoG's results with and without post-processing do not differ much. In Table II the overall results are presented for every sequence in the test set. Note that for noisy and highly dynamic scenes, like OD, the regular method without post-processing has the worst results. Also note that for the AP and FT sequences a noticeable difference between the segmentations and the "ground-truth" occurs at some point in time. This is due to sudden changes of global illumination, that are not properly handled by pixel-wise estimation of the background – in this case higher level input would be needed in order to quickly adapt to the changes. Nevertheless, CMoG easily returns to the normal classification performance. Additionally, the proposed method is faster than the original MoG: for 176×144 sequences, MoG performs at 26fps, while CMoG performs at 32fps; for 352×288 sequences, MoG performs at 7fps, while CMoG performs at 9fps.

B. CKDE

CKDE was tested with the same significance threshold T_N and collinearity threshold T_I as CMoG and a sample size N of 50 and a threshold T_S of 10^{-6}. The KDE algorithm was tested with the same common parameters.

Figure 6 shows in the second row the evolution of d^c_{sym} for the same four sequences as CMoG. CKDE performs better than KDE but in general the gains are not as significant as compared to CMoG. Also, the difference between the results obtained with and without post-processing is small and, in some sequences, the results are worse with post-processing. This is due to the fragmented masks typically produced by KDE. Some isolated fragments, if small enough, will be incorrectly removed by post-processing. Speed-wise KDE is very slow compared to MoG as already noted in Table I. Also, CKDE is faster than KDE: for 176×144 sequences, KDE performs at 5fps, while CKDE performs at 6fps; for 352×288 sequences, KDE performs at 1fps, while CKDE performs at 2fps.

C. Overall

Figure 7 shows, from left to right, the segmentation results for frames 365 and 415 of the SH sequence, frame 600 of the OD sequence and frame 2550 of the SW sequence. The top row shows the original frame, the second row shows the results from the MoG implementation of [5] and the third and fourth rows show the results obtained with CMoG and CKDE, respectively.

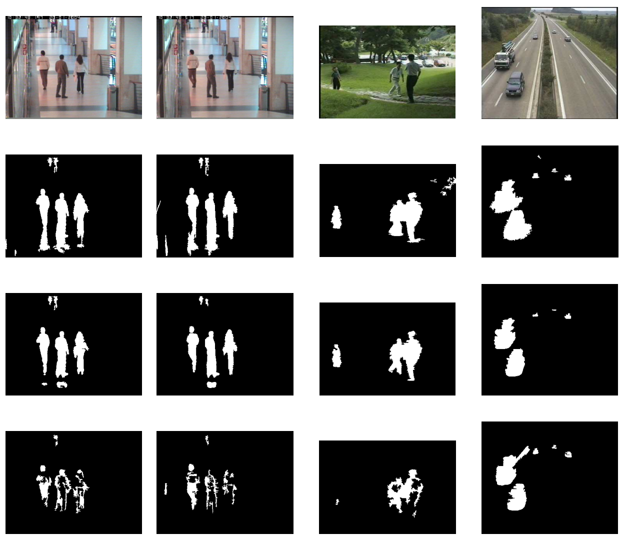
Figure 7. Segmentation results. The top row shows the original frame. The second row shows the results from the MoG implementation of [5]. The third row shows the results obtained with CMoG. Finally, the fourth row shows the results with CKDE.

Table II shows the average distance for all the sequences in the data set and Figure 8 depicts the evolution of d^c_{sym} in each sequence. Comparing CMoG and CKDE as well as their respective base algorithms, it can observed that the CMoG has the best performance with an overall average distance of 0.162 compared to 0.331 for MoG, 0.182 for CKDE and 0.199 for KDE. Again, it is clear that adapting KDE with a cascaded change detection configuration results is not as advantageous as adapting MoG. For the former an approximate gain of 8.5% achieved while for the latter the average distance between outputted segmentations and the "ground-truth" is more than halved.

V. Conclusion

The extraction moving objects, or foreground object segmentation, from a visual scene is an important first step of processing chains usually assembled to acquire higher level semantic knowledge. The objective of foreground segmentation is to dissociate the regions containing the relevant objects from regions composing the background. Due to typical dynamical behaviour of the background and variations in illumination conditions it is not always a straightforward task.

An efficient method of extracting the foreground objects consists of modelling each background pixel evolution, by estimating a probability density function (p.d.f.). Several methods to accomplish this have been proposed in the past but no objective comparison of these methods has been done before. This paper contributes with an objective comparison of background modelling and subtraction methods. For that purpose, a metric based on the partition-distance between segmentations was introduced. It accommodates human perception by weighting pixels according to their visual relevance – the cost of an incorrect classification increases with pixel distance to the object border. The mixture of Gaussians, kernel density estimation, principal features statistical modelling and mean shift methods were tested and compared using this metric. Kernel density estimation (KDE) had the best results for the test set but mixture of Gaussians (MoG) modelling proved to have the best performance/complexity relation. The main advantage of MoG is its better throughput compared to the other methods.

Taking in consideration that background changes are caused by phenomena of different nature, a method that models these independently was proposed. The method consists of a cascaded evaluation of typical dynamic elements that, although changing in time, are part of the background. These elements include acquisition noise, illumination variation and slow or repetitive structural changes. The latter type of changes is classified using the methods that estimate a p.d.f. to model the background. Also, a statistical test and a simple collinearity test are used to classify changes originated by noise and illumination changes, respectively. Two approaches were tested: one based in MoG and named cascaded MoG (CMoG), and another based in KDE named cascaded KDE (CKDE). The proposed methods CMoG and CKDE perform consistently better than the base methods. While CMoG reduces the average distance to the "ground-truth" by 51%, CKDE has an 8.5% reduction. Overall, the best results are achieved by CMoG which, even without post-processing, has similar or better performance than the base method with post-processing. Moreover, typical illumination variation changes, like shadows, are successfully eliminated without the additional use of posterior complex shadow detection and suppression techniques.

TABLE II.
AVERAGE d^c_{sym} OVER THE TEST SET SEQUENCES. CMOG HAS THE BEST PERFORMANCE WITH AN OVERALL AVERAGE DISTANCE OF 0.162 COMPARED TO 0.331 FOR MOG, 0.182 FOR CKDE AND 0.199 FOR KDE.

		SW	SH	OD	MR	CAM	LB	SC	AP	BR	SS	WS	FT
MoG	w/o pp	0.160	0.267	0.938	0.808	0.782	0.453	0.355	0.691	0.678	1.117	0.223	0.307
[5]	w/ pp	0.105	0.213	0.463	0.391	0.272	0.164	0.203	0.409	0.594	0.962	0.100	0.096
CMoG	w/o pp	0.090	0.119	0.115	0.118	0.480	0.190	0.111	0.166	0.286	0.651	0.102	0.068
	w/ pp	0.079	0.115	0.065	0.086	0.214	0.110	0.115	0.137	0.292	0.603	0.089	0.038
KDE	w/o pp	0.109	0.267	0.261	0.325	0.080	0.064	0.133	0.221	0.285	0.226	0.322	0.093
[6]	w/ pp	0.092	0.289	0.241	0.351	0.034	0.073	0.140	0.209	0.299	0.168	0.356	0.109
CKDE	w/o pp	0.112	0.235	0.234	0.325	0.151	0.063	0.108	0.199	0.190	0.253	0.329	0.069
	w/ pp	0.100	0.238	0.242	0.365	0.031	0.074	0.114	0.196	0.204	0.175	0.365	0.076

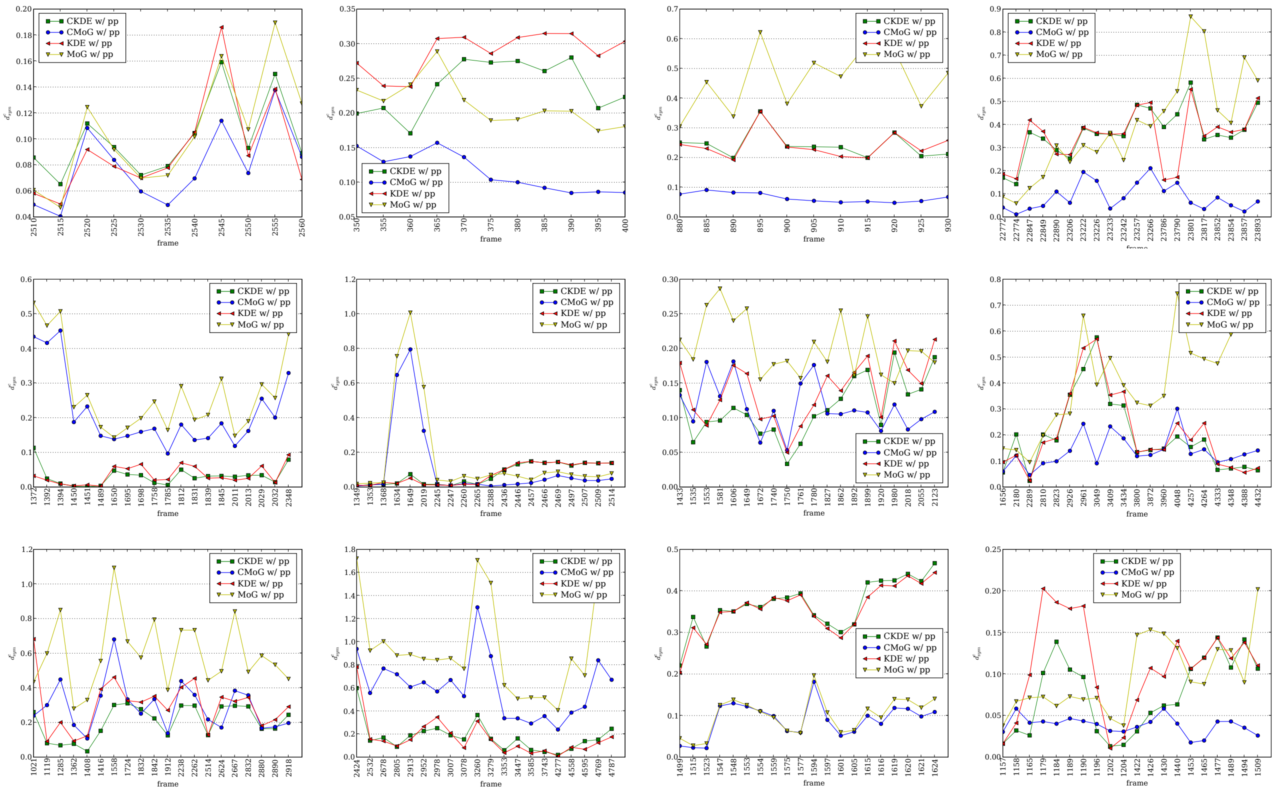

Figure 8. Evolution of d^c_{sym} for each sequence of the test set. Results are presented only with post-processing to avoid excessive graph cluttering. From left to right and from top to bottom, the graphs are from the sequences SW, SH, OD, MR, CAM, LB, SC, AP, BR, SS, WS and FT.

ACKNOWLEDGMENTS

This work has been partially supported by Fundação para a Ciência e a Tecnologia (Portuguese Science and Technology Foundation) and VISNET II, a Network of Excellence funded by the sixth Framework Programme of the European Commission.

REFERENCES

[1] D. Koller, J. Weber, and J. Malik, “Robust multiple car tracking with occlusion reasoning,” in *Proceedings of the European Conference on Computer Vision*, 1994, pp. 189–196.

[2] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: Principles and practice of background maintenance,” in *Proceedings of IEEE International Conference on Computer Vision*, vol. 1, 1999, pp. 255–261.

[3] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, “Pfinder: Real-time tracking of the human body,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 780–785, July 1997.

[4] C. Stauffer and W. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 19, 1999, pp. 246–252.

[5] D.-S. Lee, “Effective Gaussian mixture learning for video background subtraction,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 827–832, May 2005.

[6] A. Elgammal, D. Hardwood, and L. Davis, “Non-parametric model for background subtraction,” in *Proceedings of European Conference on Computer Vision*, vol. 2, 2000, pp. 751–767.

[7] I. Haritaoglu, D. Harwood, and L. S. Davis, “W4: Real-time surveillance of people and their activities,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809–830, August 2000.

[8] A. Mittal and N. Paragios, “Motion-based background subtraction using adaptive kernel density estimation,” in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 2, 2004, pp. II–302 – II–309.

[9] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian, "Statistical modeling of complex backgrounds for foreground object detection," *IEEE Transactions on Image Processing*, vol. 13, no. 11, pp. 1459–1472, November 2004.

[10] B. Han, D. Comaniciu, and L. Davis, "Sequential kernel density approximation through mode propagation: Applications to background modeling," in *Proceedings of Asian Conference on Computer Vision*, 2004.

[11] M. Piccardi and T. Jan, "Mean-shift background image modelling," in *Proceedings of International Conference on Image Processing*, 2004, pp. 3399–3402.

[12] N. Paragios and V. Ramesh, "A MRF-based approach for real-time subway monitoring," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. I–1034 – I–1040.

[13] L. Wixson, "Detecting salient motion by accumulating directionally-consistent flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 774–780, August 2000.

[14] J. Y. A. Wang and E. H. Adelson, "Representing moving images with layers," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 625–638, May 1994.

[15] S. Pundlik and S. Birchfield, "Motion segmentation at any speed," in *Proceedings of British Machine Vision Conference*, September 2006.

[16] A. Bugeau and P. Pérez, "Detection and segmentation of moving objects in highly dynamic scenes," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, June 2007.

[17] M.-P. Dubuisson and A. K. Jain, "Contour extraction of moving objects in complex outdoor scenes," *International Journal of Computer Vision*, vol. 14, no. 1, pp. 83–105, January 1995.

[18] J. A. S. Ravikanth Malladi and B. C. Vemuri, "Shape modelling with front propagation: a level set approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 158–175, February 1995.

[19] A. Chakraborty and J. S. Duncan, "Game-theoretic integration for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 1, pp. 12–30, January 1999.

[20] N. Jojic and B. J. Frey, "Learning flexible sprites in video layers," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. 199–206.

[21] P. M. Aguiar and J. M. Moura, "Figure-ground segmentation from occlusion," *IEEE Transactions on Image Processing*, vol. 14, no. 8, pp. 1109–1124, August 2005.

[22] J. S. Cardoso and L. Corte-Real, "Toward a generic evaluation of image segmentation," *IEEE Transactions on Image Processing*, vol. 14, pp. 1773–1782, November 2005.

[23] ——, "A measure for mutual refinements of image segmentations," *IEEE Transactions on Image Processing*, vol. 15, pp. 2358–2363, August 2006.

[24] A. Cavallaro, E. D. Gelasca, and T. Ebrahimi, "Objective evaluation of segmentation quality using spatio-temporal context," in *Proceedings of IEEE International Conference on Image Processing*, September 2002.

[25] M. Piccardi, "Background subtraction techniques: a review," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, October 2004.

[26] T. Aach, A. Kaup, and R. Mester, "Statistical model-based change detection in moving video," *Signal Processing*, vol. 31, no. 2, pp. 165–180, March 1993.

[27] A. Cavallaro and T. Ebrahimi, "Video object extraction based on adaptive background and statistical change detection," in *Proceedings of SPIE Visual Communications and Image Processing*, 2001, pp. 465–475.

[28] G. Tzanetakis and P. Cook, "Marsyas: a framework for audio analysis," *Organized Sound*, vol. 3, no. 4, 2000.

Luís F. Teixeira was born in Porto, Portugal, in 1979. He received the degree in Electrical and Computers Engineering in 2001, as well as the M.Sc. degree in Computer Networks and Communication Services in 2004, from the Faculdade de Engenharia, Universidade do Porto, Portugal. He is currently a Ph.D. student in the Faculdade de Engenharia, Universidade do Porto, Portugal. Since 2001 he is a researcher at INESC Porto, an R&D institute affiliated to Universidade do Porto. His research interests include visual processing, multimodal analysis and distributed multimedia processing.

Jaime S. Cardoso was born in Argivai, Póvoa de Varzim, Portugal, in 1976. He graduated in Electrical and Computers Engineering from the Faculdade de Engenharia, Universidade do Porto, Portugal, in 1999. He received the M.Sc. degree in Mathematical Engineering in 2005 from Faculdade de Cincias, Universidade do Porto, Portugal and the Ph.D. degree from the Faculdade de Engenharia, Universidade do Porto, in 2006. He is currently invited assistant professor at the Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto. Since 1999, he is a researcher at INESC Porto, an R&D institute affiliated to Universidade do Porto. His research interests include machine learning and image/video processing.

Luís Corte-Real was born in Vila do Conde, Portugal, in 1958. He received the degree in electrical engineering from the Faculdade de Engenharia, Universidade do Porto, Porto, Portugal, in 1981, the M.Sc. degree in electrical and computer engineering from the Instituto Superior Tcnico, Universidade Técnica de Lisboa, Lisbon, Portugal, in 1986, and the Ph.D. degree from the Faculdade de Engenharia, Universidade do Porto, in 1994. In 1984, he joined the Universidade do Porto as a Lecturer of telecommunications. He is currently an Associate Professor with the Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto. Since 1985, he has been a Researcher at INESC Porto, an R&D institute affiliated with Universidade do Porto. His research interests include image/video coding and processing and content-based image/video retrieval.