A Novelty Detection Approach for Foreground Region Detection in Videos with Quasi-stationary Backgrounds

Alireza Tavakkoli[1], Mircea Nicolescu[1], and George Bebis[1]

Computer Vision Lab.
Department of Computer Science and Engineering
University of Nevada, Reno, USA
{tavakkol, mircea, bebis}@cse.unr.edu

Abstract. Detecting regions of interest in video sequences is one of the most important tasks in many high level video processing applications. In this paper a novel approach based on support vector data description is presented, which detects foreground regions in videos with quasi-stationary backgrounds. The main contribution of this paper is the novelty detection approach which automatically segments video frames into background/foreground regions. By using support vector data description for each pixel, the decision boundary for the background class is modeled without the need to statistically model its probability density function. The proposed method is able to achieve very accurate foreground region detection rates even in very low contrast video sequences, and in the presence of quasi-stationary backgrounds. As opposed to many statistical background modeling approaches, the only critical parameter that needs to be adjusted in our method is the number of background training frames.

1 Introduction

In most visual surveillance systems, stationary cameras are typically used. However, because of inherent changes in the background itself, such as fluctuations in monitors and fluorescent lights, waving flags and trees, water surfaces, etc. the background of the video may not be completely stationary. In these types of backgrounds, referred to as quasi-stationary, a single background frame is not useful to detect moving regions. Pless *et al.* [1] evaluated different models for dynamic backgrounds. Typically, background models are defined independently on each pixel and depending on the complexity of the problem, use the expected pixel features (i.e. colors) [2] or consistent motion [3]. Also they may use pixel-wise information [4] or regional models of the features [5].

In [4] a single 3-D Gaussian model for each pixel in the scene is built, where the mean and covariance of the model were learned in each frame. Kalman Filtering [6] is also used to update the model. These background models were unable to represent multi-modal situations. A mixture of Gaussians modeling technique was proposed in [7] and [8] to address the multi-modality of the underlying background. There are several shortcomings for the mixture learning methods. First, the number of Gaussians needs to be specified. Second, this method does not explicitly handle spatial dependencies. Also, even with the use of incremental-EM, the parameter estimation and its convergence is noticeably slow where the Gaussians adapt to a new cluster. The convergence speed can be improved by sacrificing memory as proposed in [9], limiting its applications where mixture modeling is pixel-based and over long temporal windows. A recursive filter formulation is proposed by Lee in [10]. However, the problem of specifying the number of Gaussians as well as the adaptation in later stages still exists. Also this model does not account for the situations where the number of Gaussians changes due to occlusion or uncovered parts of the background.

In [2], El Gammal *et al.* proposed a non-parametric kernel density estimation method (KDE) for pixel-wise background modeling without making any assumption on its probability distribution. Therefore, this method can easily deal with multi-modality in background pixel distributions without determining the number of modes in the background. However, there are several issues to be addressed using non-parametric kernel density estimation. First, these methods are memory and time consuming. For each pixel in each frame the system has to compute the average of all the kernels centered at each training feature vector. Second, the size of the temporal window used as the background buffer needs to be specified. Too small a window increases the estimation speed, while it does not incorporate enough history for the pixel, resulting in a less accurate model. Also the adaptation will be problematic by using small window sizes. Increasing the window size improves the accuracy of the model but with the cost of more memory requirements and slower convergence. Finally, the non-parametric KDE methods are pixel-wise techniques and do not use the spatial correlation of the pixel features. In order to adapt the model a sliding window is used in [11]. However the model convergence is problematic in situations where the illumination suddenly changes.

In order to update the background for scene changes such as moved objects, parked vehicles or opened/closed doors, Kim *et al.* in [12] proposed a layered modeling technique. This technique needs an additional model called *cache* and assumes that the background modeling is performed over a long period. It should be used as a post-processing stage after the background is modeled.

In methods that explicitly model the background density estimation, foreground detection is performed by comparing the estimated probabilities of each pixel with a global threshold [2], or local thresholds [13]. Also there are several parameters that need to be estimated from the data to achieve an accurate density estimation for background. In [11] a binary classification technique is used to detect foreground regions by a maximum likelihood method. Since in these techniques the probability density function of the background is estimated, the model accuracy is bounded to the accuracy of the estimated probability.

In this paper a novel approach is proposed to label pixels in video sequences into foreground and background classes using support vector data description

[14]. As opposed to parametric and non-parametric density estimation techniques, in this method the model is not the probability function of the background or foreground. It can be considered as analytical description of the decision boundary between background and foreground classes. This modeling technique addresses several issues in the traditional density estimation approaches.

First, the model accuracy is not bounded to the accuracy of the estimated probability density functions. Second, the memory requirements of the proposed technique are less than those of non-parametric techniques. In non-parametric density estimation methods, pixel feature vectors for all background training frames need to be stored to regenerate the probability of pixels in new frames. It can be problematic for large frames sizes and temporal windows. In our technique, in order to classify new pixels, they are compared only with the support vectors, which in practice are much fewer than the actual number of frames in the temporal window. Third, because support vector data description explicitly models the decision boundary of the known class, it can be used for novelty detection and single class-classification without a need to threshold any values. This results in less parameter tuning and automatic classification. Finally, the performance of the classifier in terms of false positive and false negatives can be controlled from within the framework. The proposed method is a novel approach that explicitly addresses the one-class classification problem, since in foreground region detection we do not have samples of foreground regions in the training steps of the system. This issue, has not been addressed in any of the traditional techniques.

The rest of this paper is organized as follows. In Section 2 a brief review of the support vector data description is presented. In Section 3, the proposed method for foreground region detection is discussed. Section 4 shows experimental results of our method on synthetic and real-world data, and the performance of classifier is compared with the existing techniques. Finally, conclusions of the proposed method are drawn in Section 5 and future extensions to this work are discussed.

2 Support vector data description

Data domain description concerns the characteristics of a data set [14]. The boundary of the dataset can be used to detect novel data or outliers. A normal data description gives a closed boundary around the data. The simplest boundary can be represented by a hyper-sphere. The volume of this hyper-sphere with center a and radius R should be minimized while containing all the training samples x_i. To allow the possibility of outliers in the training set, slack variables $\epsilon_i \geq 0$ are introduced. The error function to be minimized is defined as:

$$F(R, a) = R^2 + C \sum_i \epsilon_i \tag{1}$$

subject to the constraints:

$$\|x_i - a\|^2 \leq R^2 + \epsilon_i \quad \forall i \tag{2}$$

In equation (1), C is a trade-off between simplicity of the system and its error. We call this parameter confidence parameter. After incorporating the constraints (2) into the error function (1) by Lagrange multipliers we have:

$$L\left(R, a, \alpha_i, \gamma_i, \epsilon_i\right) = R^2 + C \sum_i \epsilon_i - \sum_i \alpha_i \left[R^2 + \epsilon_i - \left(\|x_i - a\|^2\right)\right] - \sum_i \gamma_i \epsilon_i \quad (3)$$

L should be maximized with respect to Lagrange multipliers $\alpha_i \geq 0$ and $\gamma_i \geq 0$ and minimized with respect to R, a and ϵ_i. Lagrange multipliers γ_i can be removed if the constraint $0 \leq \alpha_i \leq C$ is imposed. After solving the optimization problem we have:

$$L = \sum_i \alpha_i (x_i \cdot x_i) - \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j) \quad \forall \alpha_i : 0 \leq \alpha_i \leq C \qquad (4)$$

When a new sample satisfies the inequality in (2), then its corresponding Lagrange multipliers are $\alpha_i \geq 0$, otherwise they are zero. Therefore we have:

$$\begin{aligned} &\|x_i - a\|^2 < R^2 \rightarrow \alpha_i = 0, \gamma_i = 0 \\ &\|x_i - a\|^2 = R^2 \rightarrow 0 < \alpha_i < C, \gamma_i = 0 \\ &\|x_i - a\|^2 > R^2 \rightarrow \alpha_i = C, \gamma_i > 0 \end{aligned} \qquad (5)$$

From the above, we can see that only samples with non-zero α_i are needed in the description of the data set, therefore they are called *support vectors* of the description. To test a new sample y, its distance to the center of the hyper-sphere is calculated and tested against R.

In order to have a flexible data description as opposed to the simple hyper-sphere discussed above a kernel function $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ is introduced. This maps the data into a higher dimensional space, where it is described by the simple hyper-sphere boundary.

3 The proposed method

The methodology described in section 2 is used in our technique to build a descriptive boundary for each pixel in the background training frames to generate its model for the background. Then these boundaries are used to classify their corresponding pixels in new frames as background and novel (foreground) pixels. There are several advantages in using the Support Vector Data Description (SVDD) method in detecting foreground regions:

- The proposed method, as opposed to existing statistical modeling methods, explicitly addresses the single-class classification problem. Existing statistical approaches try to estimate the probability of a pixel being background, and then use a threshold for the probability to classify it into background or foreground regions. The disadvantage of these approaches is in the fact that it is impossible to have an estimate of the foreground probabilities, since there are no foreground samples in the training frames.

```
1. Initialization
    1.1. Confidence parameter: $C$
    1.2. Number of training samples: Trn_No
    1.3. Kernel bandwidth $\sigma$
2. For each training frame
    2.1. For each pixel $\mathtt{x}_{ij}$
        2.1.1. OC(i,j)$\leftarrow$ Train($\mathtt{x}_{ij}$[1 :Trn_No])
3. For new frames
    3.1. For each pixel $\mathtt{x}_{ij}$
        3.1.1. Desc(i,j)$\leftarrow$ Test($\mathtt{x}_{ij}$[Current Frame],OC(i,j))
        3.1.2. Label pixel as foreground or background
               based on Desc(i,j).
```

Fig. 1. The proposed algorithm.

– The proposed method has less memory requirements compared to non-parametric density estimation techniques, where all the training samples for the background need to be stored in order to estimate the probability of each pixel in new frames. The proposed technique only requires a very small portion of the training samples, *support vectors*, to classify new pixels.

– The accuracy of our method is not limited to the accuracy of the estimated probability density functions for each pixel. Also the fact that there is no need to assume any parametric form for the underlying probability density of pixels gives the proposed method superiority over the parametric density estimation techniques, i.e. mixture of Gaussians.

– The efficiency of our method can be explicitly measured in terms of false reject rates. The proposed method considers a goal for false positive rates, and generates the description of data by fixing the false positive tolerance of the system. This helps in building a robust and accurate background model.

Figure 1 shows the proposed algorithm in pseudo-code format[1]. The only critical parameter is the number of training frames (Trn_No) that needs to be initialized. The support vector data description confidence parameter C is the target false reject rate of the system. This is not a critical parameter and accounts for the system tolerance. Finally the Gaussian kernel bandwidth, σ does not have a particular effect on the detection rate as long as it is not set to be less than one, since features used in our method are normalized pixel chrominance values. For all of our experiments we set $C = 0.1$ and $\sigma = 5$.

In the first step of the algorithm, for each pixel in the scene a single class classifier is trained by using its values in the background training frames. This classifier consists of the description boundary and support vectors, as well as a threshold used to describe the data. In the next step, each pixel in the new frames is classified as background or foreground using its value and its corresponding

[1] The proposed method is implemented in MATLAB 6.5, using Data Description toolbox [15].

classifier from the training stage. Details of training of each classifier and testing for new data samples are provided in section 2.

Feature vectors x_{ij} used in the current implementation are $x_{ij} = [cr, cg]$, where cr and cg are the red and green chrominance values for pixel (i, j). Since there is no assumption on the dependency between features, any feature value such as vertical and horizontal motion vectors, pixel intensity, etc. can be used.

4 Experimental results

In this section, the experimental results of the proposed method are presented on both synthetic and real data. The experiments are conducted to compare the results of support vector data description in classification problems with those of traditional methods, such as mixture of Gaussians (MoG), Kernel Density Estimation (KDE) and k-nearest neighbors (KNN).

4.1 Synthetic data sets

In this section we use a synthetic data set, which represents randomly distributed training samples with an unknown distribution function (*banana* data set). Figure 2 shows a comparison between different classifiers. This experiment is performed on 150 training samples using support vector data description (SVDD), mixture of Gaussians (MoG), kernel density estimation (KDE) and k-nearest neighbors (KNN).

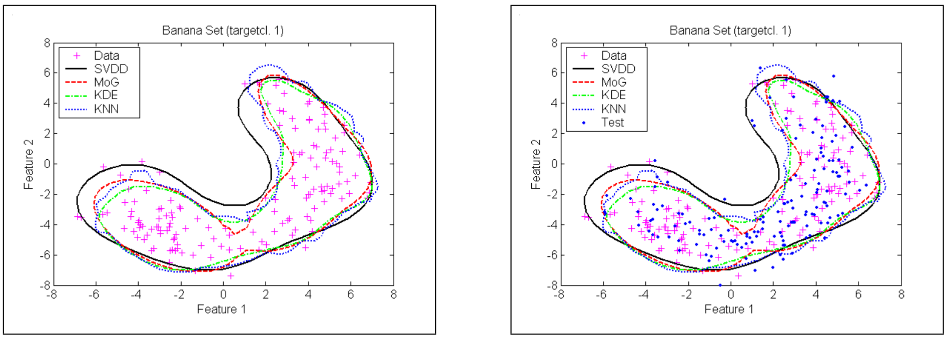
Parameters of these classifiers are manually determined to give a good performance. For all classifiers the confidence parameter is set to be 0.1. In MoG, we used 3 Gaussians. Gaussian kernel bandwidth in the KDE classifier is considered $\sigma = 1$, for the KNN we used 5 nearest neighbors, and for the SVDD classifier the Gaussian kernel bandwidth is chosen to be 5.

Figure 2(a) shows the decision boundaries of different classifiers on 150 training samples from *banana* dataset. As it can be seen from Figure 2(b), SVDD generalizes better than the other three classifiers and classifies the test data more accurately. In this Figure the test data is composed of 150 samples drawn from the same probability distribution function as the training data, thus should be classified as the known class.

Table 1. Comparison of False Reject Rate and Recall Rate for different classifiers.

Method	SVDD	MoG	KDE	KNN
FRR	0.1067	0.1400	0.1667	0.1333
RR	0.8933	0.8600	0.8333	0.8667

Here we need to define the False Reject Rate (FRR) and Recall Rate (RR) for a quantitative evaluation. By definition, FRR is the percentage of missed



(a) (b)

Fig. 2. Comparison between different classifiers on a synthetic data set: (a)- Decision boundaries of different classifiers after training. (b)- Data points (blue dots) outside decision boundaries are false rejects.

targets, and RR is the percentage of correct prediction (True Positive rate). These quantities are given by:

$$\mathrm{FRR} = \frac{\#\text{Missed targets}}{\#\text{Samples}} \qquad \mathrm{RR} = \frac{\#\text{Correct predictions}}{\#\text{Samples}} \tag{6}$$

Table 1 shows a quantitative comparison between different classifiers. In this table, FRR and RR of classifiers are compared after training them on 150 data points drawn from an arbitrary probability function and tested on the same number of samples drawn from the same distribution. As it can be seen from the above example, the FRR for SVDD classifier is less than that of the other three, while its RR is higher. This proves the superiority of this classifier in case of single class classification over the other three techniques.

Table 2. Need for manual optimization and number of parameters.

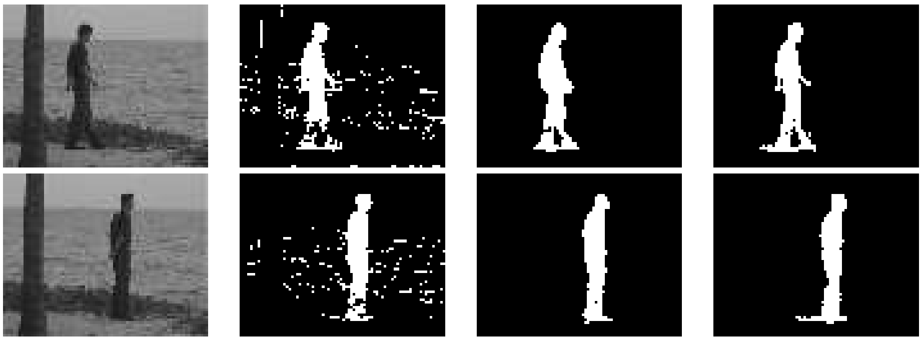
Method	SVDD	MoG	KDE	KNN
No. of parameters	1	4	2	2
Need manual selection	No	Yes	Yes	Yes

Table 2 compares the number of parameters that each classifier has and the need for manually selecting these parameters for a single class classification problem. As it can be seen, the only method that automatically determines data description is the SVDD technique. In all of the other classification techniques there is at least one parameter that needs to be manually chosen to give a good performance.

Table 3 shows memory requirements for each classifier. As it can be seen from the table, SVDD requires much less memory than the KNN and KDE methods, since in SVDD we do not need to store all the training data. Only the MoG

Table 3. Comparison of memory requirements for different classifiers.

Method	SVDD	MoG	KDE	KNN
Memory needs (bytes)	1064	384	4824	4840



(a) Water surface (b) MoG (c) KDE (d) SVDD

Fig. 3. Water surface video: comparison of methods in presence of irregular motion.

method needs less memory than the SVDD technique, but in MoG methods we need to manually determine the number of Gaussians to be used which is not practical when we are training one classifier per pixel in real video sequences.

4.2 Real videos

In this section, foreground detection results of our method on real video sequences are shown and compared with the traditional statistical modeling techniques.

Comparison in the presence of irregular motion. By using the *water surface* video sequence, we compare the results of foreground region detection using our proposed method with a typical KDE [13] and MoG [7]. For this comparison the sliding window of size L=150 is used in the KDE method. The results of MoG are shown in Figure 3(b), the KDE method results are shown in Figure 3(c) and the foreground masks detected by the proposed technique are shown in Figure 3(d). As it can be seen, the proposed method gives better detection since it generates a more accurate descriptive boundary on the training data, and does not need a threshold to classify pixels as background or foreground.

Comparison in case of low contrast videos. Figure 4 shows the result of foreground detection using the proposed method in the *hand shake* video sequence and compares this result with that of the KDE method. As it can be seen from Figure 4(b) and 4(c), the proposed method achieves better detection rates compared to the KDE technique. Notice that in the KDE technique presented in the figure, the system tries to find the best parameters for the classifier in order to achieve its best performance.

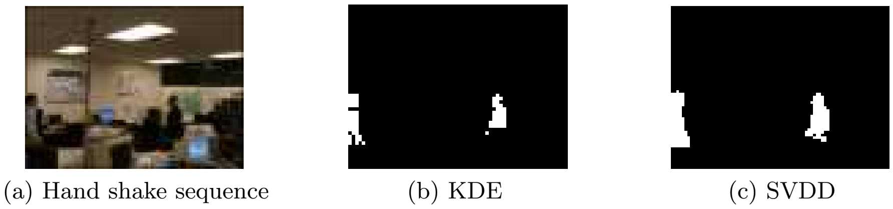

(a) Hand shake sequence (b) KDE (c) SVDD

Fig. 4. Hand shake video: comparison of methods in case of low contrast videos.

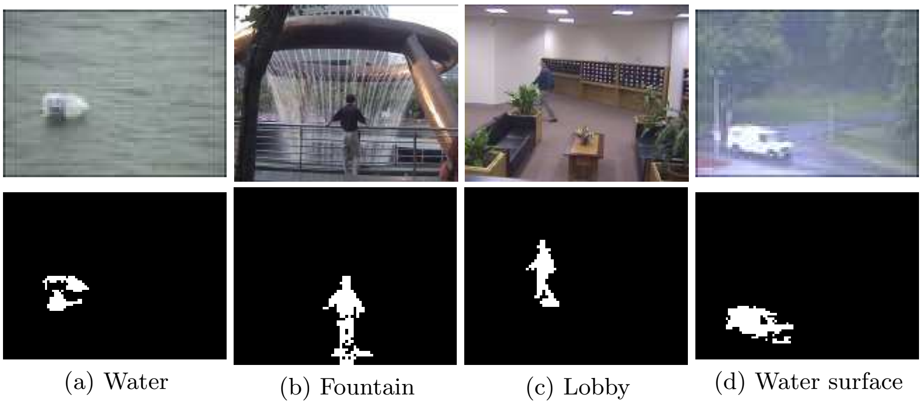

(a) Water (b) Fountain (c) Lobby (d) Water surface

Fig. 5. Foreground detection results.

Detection results in difficult scenarios. Figure 5 shows results of the proposed foreground detection algorithm in very difficult situations. In Figure 5(a) and 5(b) the irregular motion of water in the background make it difficult to detect true foreground regions. In Figure 5(c) there are flickers in the lighting. Our system accurately detects the foreground regions in all of these situations. Also the car in Figure 5(d) is detected accurately by our method despite the presence of waving tree branches and the rain in the background.

5 Conclusions and future work

In this paper a novel approach is proposed to label pixels in video sequences into foreground and background classes using support vector data description. The contributions of our method can be described along the following directions:

- The model accuracy is not bounded to the accuracy of the estimated probability density functions.
- The memory requirement of the proposed technique is less than that of non-parametric techniques.
- Because support vector data description explicitly models the decision boundary of the known class, it is suitable for novelty detection without the need to use thresholds. This results in less parameter tuning.

– The classifier performance in terms of false positive is controlled explicitly.

One direction of future investigation is to use this work in non-parametric tracking approaches. Also we are investigating the effect of adaptive kernel bandwidth parameters on the performance of the system.

6 Acknowledgement

This work was supported by the NSF-EPSCoR Ring True III award EPS0447416 and by the Office of Naval Research award N00014-06-1-0611.

References

1. Pless, R., Larson, J., Siebers, S., Westover, B.: Evaluation of local models of synamic backgrounds. Proceedings of the CVPR **2** (2003) 73–78.
2. Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.: Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proceedings of the IEEE **90** (2002) 1151–1163.
3. Pless, R., Brodsky, T., Aloimonos, Y.: Detecting independent motion: The statistics of temporal continuity. IEEE Transactions on PAMI **22** (2000) 68–73.
4. Wern, C., Azarbayejani, A., Darrel, T., Petland, A.: Pfinder: real-time tracking of human body. IEEE Transactions on PAMI **19** (1997) 780–785.
5. Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: principels and practive of background maintenance. Proceedings of ICCV **1** (1999) 255–261.
6. Koller, D., adn T. Huang, J.W., Malik, J., Ogasawara, G., Rao, B., Russel, S.: Towards robust automatic traffic scene analysis in real-time. ICPR **1** (1994) 126–131.
7. Stauffer, C., Grimson, W.: Learning patterns of activity using real-time tracking. IEEE Transactions on PAMI **22** (2000) 747–757.
8. Friedman, N., Russell, S.: Image segmentation in video sequences: A probabilistic approach. Annual Conference on Uncertainty in Artificial Intelligence (1997) 175–181.
9. McKenna, S., Raja, Y., Gong, S.: Object tracking using adaptive color mixture models. Proceedings of Asian Conferenc on Computer Vision **1** (1998) 615–622.
10. Lee, D.S.: Effective gaussian mixture learning for video background subtraction. IEEE Transactions on PAMI **27** (2005) 827–832.
11. Mittal, A., Paragios, N.: Motion-based background subtraction using adaptive kernel density estimation. Proceedings of CVPR **2** (2004) 302–309.
12. Kim, K., Harwood, D., Davis, L.S.: Background updating for visual surveillance. Proceedings of the International Symposium on Visual Computing **1** (2005) 337–346.
13. Tavakkoli, A., Nicolescu, M., Bebis, G.: Automatic robust background modeling using multivariate non-parametric kernel density estimation for visual surveillance. Proceedings of the International Symposium on Visual Computing **LNSC 3804** (2005) 363–370.
14. Tax, D.M.J., Duin, R.P.: Support vector data description. Machine Learning **54** (2004) 45–66.
15. Tax, D.: Ddtools, the data description toolbox for matlab (2005) version 1.11.