7

Novelty Detection: An Approach to Foreground Detection in Videos

Alireza Tavakkoli
University of Nevada, Reno
USA

1. Introduction

Classification is an important mechanism in many pattern recognition applications. In many of these application, such as object recognition, there are several classes from which the data originates. In such cases many traditional classification methods such as Artificial Neural Networks or Support Vector Machines are used. However, in some applications the training data may belong to only one class. In this case, the classification is performed by finding whether a test sample belongs to the known class or not. The main criteria in single-class classification (also known as novelty detection) is to perform the classification without any information about other classes.

This chapter presents a classic problem in video processing applications and addresses the issues through novelty detection techniques. The problem at hand is to detect foreground objects in a video with quasi-stationary background. The video background is called quasi-stationary if the camera is static but the background itself changes due to waving tree branches, flags, water surfaces, etc. Detection of foreground region in such scenarios requires a pixel-wise background model for each pixel in the scene. Once the pixel models are built, there should be a mechanism to decide whether pixels in new frames belong to their corresponding background model or not. The generation of pixel models from their history and the decision making mechanism is a novelty detection problem.

In order to address the foreground detection problem, two main approaches to novelty detection, namely statistical and analytical, are presented in this chapter. The advantage and disadvantages of these approaches are discussed. Moreover, the suitability of each approach to specific scenarios in video processing applications are evaluated.

2. Foreground Detection

Detecting foreground regions in videos is one of the most important tasks in high-level video processing applications. One of the major issues in detecting foreground regions using background subtraction techniques is that because of inherent changes in the background, such as fluctuations in monitors and lights, waving flags and trees, water surfaces, the background may not be completely stationary. These difficult situations are illustrated in Fig. 1.

In the presence of these types of backgrounds, referred to as quasi-stationary, a single background frame is not enough to accurately detect moving regions. Therefore the background pixels of the video have to be modeled in order to detect foreground regions while allowing

(a) (b) (c)

Fig. 1. Examples of challenges in quasi-stationary backgrounds: (a) Fluctuating monitors. (b) Rain/Snow. (c) Waving tree branches.

for the changes in the background. The scenarios in which the background modeling techniques are used to detect foreground regions are very diverse. Applications vary from indoor scenes to outdoor, from completely stationary to dynamic backgrounds, from high quality videos to low contrast scenes and so on. Therefore, a single system capable of addressing all possible situations while being time and memory efficient is yet to be devised.

(Pless et al., 2003) evaluated different models for dynamic backgrounds. Typically, background models are defined independently on each pixel, and depending on the complexity of the problem employ the expected pixel features (i.e. colors), (Elgammal et al., 2002), or consistent motion, (Pless et al., 2000). They also may employ pixel-wise information, (Wern et al., 1997), or regional models of the features, (Toyama et al., 1999). To improve robustness to spatio-temporal features, (Li et al., 2004), may be used.

In (Wern et al., 1997) a single 3-D Gaussian model for each pixel in the scene is built, where the mean and covariance of the model are learned in each frame. This system tried to model the noise and used a background subtraction technique to detect those pixels whose probabilities are smaller than a threshold. However, the system fails to label a pixel as foreground or background when it has more than one modality due to fluctuations in its values, such as a pixel belonging to a fluctuating monitor.

A mixture of Gaussians modeling technique was proposed in (Stauffer & Grimson, 2000); (Stauffer & Grimson, 1999) to address the multi-modality of the underlying background. In this modeling technique background pixels are modeled by a mixture of a number of Gaussian functions. During the training stage, parameters of each Gaussian are trained and used in the background subtraction, where the probability of each pixel is generated. Each pixel is labeled as foreground or background based on its probability.

There are several shortcomings for mixture learning methods. First, the number of Gaussians needs to be specified. Second, this method does not explicitly handle spatial dependencies. 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 (McKenna et al., 1998), limiting its applications where mixture modeling is pixel-based and over long temporal windows.

In (Elgammal et al., 2002), a non-parametric kernel density estimation method (KDE) for pixel-wise background modeling is proposed without making any assumption on its probability distribution. Therefore, this method can easily deal with multi-modality in background pixel distributions without specifying the number of modes in the background. However, there are several issues to be addressed using non-parametric kernel density estimation. These

methods are memory and time consuming since the system has to compute the average of all kernels centered at each training sample for each pixel in each frame. Also the size of temporal window used as the background model is critical. In order to adapt the model a sliding window is used in (Mittal & Paragios, 2004). However, the model convergence is problematic in situations where the illumination suddenly changes.
In the traditional approaches for foreground detection presented above, the problem is addressed by reformatting a bi-class classification methodology to fit into the novelty detection approach. For example in the Mixture of Gaussian approach, changes in each pixel are modeled by a number of Gaussian functions. For new pixels a probability is calculated using the pixel model. Then a heuristically selected threshold is used to determine whether the pixel belongs to background or foreground based on its probability.
The major drawback of such approaches is the threshold choice. In these statistical approaches such as the mixture of Gaussians or the KDE, the pixel model is its probability distribution function belonging to the background. Since the background is quasi-stationary and natural, pixels in different locations undergo different amount of changes. Since the probability density functions are normalized, the pixels with less changes will have narrow but tall probability density functions while the pixels with more changes are represented by wider but shorter density functions. Therefore, finding a global threshold that works well for the majority of the background pixels and in a diverse range of applications is practically untractable.
In this chapter, two approaches based on novelty detection to address the single class classification, inherent to background modeling, are investigated. The statistical approach is based on a recursive modeling of the background pixels. This technique is called the RM, (Tavakkoli et al., 2006c). As an alternative to this statistical approach an analytical counter part to the RM technique is presented and is based on the Support Vector Data Description, (Tax & Duin, 2004). This technique is called Support Vector Data Description Modeling (SVDDM) and looks at modeling the pixels as an analytical description boundary, (Tavakkoli, Kelley, King, Nicolescu, Nicolescu & Bebis, 2007). An incremental version of the SVDDM technique is presented in (Tavakkoli, Nicolescu & Bebis, 2008).
The rest of this chapter is organized as follows. In Section 3 the theory behind the RM technique is presented. Section 4 gives a detailed algorithm of the support vector data description method in detecting foreground regions in video sequences. Performances of the proposed methods are evaluated in Section 5. Section 6 presents a comparison between the performance of these techniques and other existing methods on real videos as well as synthetic data and a comparison summary is drawn in this section. Finally, Section 7 concludes the chapter and gives future direction for research.

3. The Recursive Modeling

This section describes a technique called Recursive Modeling (RM) for foreground region detection in videos. The theory behind this approach is to generate a histogram of the data samples, with the hope that when a large number of training samples are processed, the histogram estimates the actual probability of the underlying data. System details and its theory are explained in the following, (Tavakkoli et al., 2006a), (Tavakkoli et al., 2006c), and (Tavakkoli, Nicolescu, Bebis & Nicolesu, 2008).

3.1 The theory

Let x_t be the the intensity value of a pixel at time t. The non-parametric estimation of the background model that accurately follows its multi-modal distribution can be reformulated

in terms of recursive filtering, (Tavakkoli, Nicolescu, Bebis & Nicolesu, 2008):

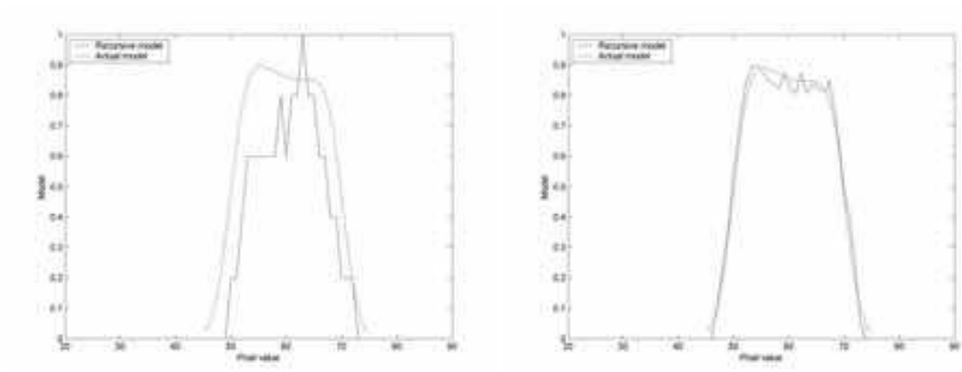
$$\hat{\theta}_t^B(x) = [1-\beta_t]\cdot\theta_{t-1}^B(x) + \alpha_t \cdot H_\Delta(x - x_t) \qquad \forall x \in [0,255] \tag{1}$$

$$\sum_{x=0}^{255} \theta_t^B(x) = 1 \tag{2}$$

where θ_t^B is the background pixel model at time t, normalized according to (2). $\hat{\theta}_t^B$ is updated by the local kernel $H(\cdot)$ with bandwidth Δ centered at x_t. Parameters α_t and β_t are the learning rate and forgetting rate schedules, respectively. The kernel H should satisfy the following:

$$\begin{aligned} \sum_x H_\Delta(x) &= 1 \\ \sum_x x \times H_\Delta(x) &= 0 \end{aligned} \tag{3}$$

These conditions should be satisfied to ensure that the kernel is normalized, symmetric and positive definite in case of multivariate kernels. Note that in this context there is no need to specify the number of modalities of the background representation at each pixel. In our implementation of the RM method we use a Gaussian kernel which satisfies the above conditions.



(a) Model after 10 frames (b) Model after 200 frames

Fig. 2. Recursive modeling convergence to the actual probability density function over time.

Figure 2 shows the updating process using our proposed recursive modeling technique. It can be seen that the trained model (solid line) converges to the actual one (dashed line) as new samples are introduced. The actual model is the probability density function of a sample population and the trained model is generated by using the recursive formula in (1).
In existing non-parametric kernel density estimation methods, the learning rate α is selected to be constant and has small values. This makes the pixel model convergence slow and keeps its history in the recent temporal window of size $L = 1/\alpha$. The window size in non-parametric models is important as the system has to cover all possible fluctuations in the background model. That is, pixel intensity changes may not be periodic or regular and consequently do not fit in a small temporal window. In such cases larger windows are needed, resulting in higher memory and computational requirements to achieve accurate, real-time modeling.

Another issue in non-parametric density estimation techniques is that the window size is fixed and is the same for all pixels in the scene. However, some pixels may have less fluctuations and therefore need smaller windows to be accurately modeled, while others may need a much longer history to cover their fluctuations.

3.1.1 Scheduled learning

In order to speed up the modeling convergence and recovery we use a schedule for learning the background model at each pixel based on its history. This schedule makes the adaptive learning process converge faster, without compromising the stability and memory requirements of the system. The learning rate changes according to the schedule:

$$\alpha_t = \frac{1-\alpha_0}{h(t)} + \alpha_0 \tag{4}$$

where α_t is the learning rate at time t and α_0 is a small target rate which is:

$$\alpha_0 = 1/256 \times \sigma_\theta \tag{5}$$

where σ_θ is the model variance. The function $h(t)$ is a monotonically increasing function:

$$h(t) = t - t_0 + 1 \tag{6}$$

where t_0 is the time at which a sudden global change is detected. At early stages the learning occurs faster ($\alpha_t = 1$), then it monotonically decreases and converges to the target rate ($\alpha_t \rightarrow \alpha_0$). When a global change is detected $h(t)$ resets to 1. The effect of this schedule on improving the convergence and recovery speed are discussed later.

The forgetting rate schedule is used to account for removing those values that have occurred long time ago and no longer exist in the background. In the current implementation we assume that the forgetting rate is a portion of the learning rate $\beta_t = l \cdot \alpha_t$, where $l \leq 1$. In the current implementation $l = 0.5$ is employed in all experiments. This accounts for those foreground objects that cover some parts of the background but after a sufficiently small period move. This keeps the history of the covered background in short-term.

3.1.2 Incorporating color information

The recursive learning scheme in 1-D has been explained in the previous section. The background and foreground models are updated using the intensity value of pixels at each frame. To extend the modeling to higher dimensions and incorporate color information, one may consider each pixel as a 3 dimensional feature vector in $[0,255]^3$. The kernel H in this space is a multivariate kernel H_Σ. In this case, instead of using a diagonal matrix H_Σ a full multivariate kernel can be used. The kernel bandwidth matrix Σ is a symmetric positive definite 3×3 matrix. Given N pixels, $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$, labeled as background, their successive deviation matrix is a matrix Δ_X whose columns are:

$$\left[\mathbf{x}_i - \mathbf{x}_{i-1}\right]^T \quad \text{with} \quad i = 2,3,\cdots,N \tag{7}$$

The bandwidth matrix is defined so that it represents temporal scatter of the training data:

$$\Sigma = \text{cov}(\Delta_X) \tag{8}$$

In order to decrease the memory requirements of the system we assumed that the two chrominance values are independent. Making this assumption results in a significant decrease in memory requirements while the accuracy of the model does not decay drastically. The red/green chrominance values can be quantized into 256 discrete values.

1. Initialization; $\Delta, \alpha_0, \beta, \kappa$ and th
2. For each frame
   For each pixel
   2.1. **Training stage**
   - Update $\alpha_t = \frac{1-\alpha_0}{h(t)} + \alpha_0$ and Δ
   - Update $\theta_t^B = (1-\beta_t)\theta_{t-1}^B + \alpha_t \cdot H_\Delta$
   - If $\theta_t^B \leq th$ then update $\theta_t^F = (1-\beta_t)\theta_{t-1}^F + \alpha_t \cdot H_\Delta$

   2.2. **Classification stage**
   - If $ln\left(\text{med}(\theta_t^F)/\text{med}(\theta_t^B)\right) \geq \kappa$ then label pixel as foreground.

   2.3. **Update stage**
   - Update κ and th

Fig. 3. The RM algorithm.

3.2 The algorithm

The proposed method, in pseudo-code, is shown in Figure 3. There are three major steps in the RM method: training, classification and update stages, respectively. The role and results of each stage along with its details are presented in the following.

3.2.1 The Training Stage

Before new objects appear in the scene, at each pixel all the intensity values have the same probability of being foreground. However, in each new frame the pixel background models are updated according to equation (1), resulting in larger model values (θ^B) at the pixel intensity value x_t. In essence, the value of the background pixel model at each intensity x is:

$$\theta_t^B(x) = P(\text{Bg}|x) \qquad x \in [0,255] \tag{9}$$

In order to achieve better detection accuracy we introduce the foreground model which in the classification stage is compared to the background model to make the decision on whether the pixel belongs to background or foreground. This foreground model represents all other unseen intensity/color values for each pixel that does not follow the background history and is defined by:

$$\hat{\theta}_t^F(x) = [1-\beta_t^F] \cdot \theta_{t-1}^F(x) + \alpha_t^F \cdot H_\Delta(x - x_t) \qquad \forall x \in [0,255] \tag{10}$$

$$\sum_{x=0}^{255} \theta_t^F(x) = 1 \tag{11}$$

Once the background model is updated, it is compared to its corresponding threshold th. This threshold is automatically maintained for each pixel through the update stage which is described in details later. If the pixel probability is less than this threshold the foreground model for that pixel value is updated according to (10) and (11).

3.2.2 The Classification stage

For each pixel at time t we use a function θ_t^B for the background model and θ_t^F for the foreground. The domain of these functions is $[0,255]^N$, where N is the dimensionality of the pixel feature vector. For simplicity assume the one dimensional case again, where θ_t is the background/foreground model whose domain is $[0,255]$. From equation (10), each model ranges between 0 to 1 and its value shows the amount of evidence accumulated in the updating process (i.e., the estimated probability). For each new intensity value x_t we have the evidence

(a) An arbitrary frame (b) The threshold map

Fig. 4. Adaptive threshold map: different pixels need different thresholds.

of each model as $\theta_t^B(x_t)$ and $\theta_t^F(x_t)$. The classification uses a *maximum a posteriori* criterion to label the pixel as foreground:

$$ln\left(\frac{\theta_t^B}{\theta_t^F}\right) \leq \kappa \tag{12}$$

3.2.3 The Update stage

In order for the RM technique to address the single class classification problem at hand there is a need for an adaptive classification criteria. Because not all pixels in the scene follow the same changes, the decision threshold, θ and κ should be adaptive and independent for each pixel and has to be derived from the history of that pixel. Figure 4 explains this issue.
For each pixel its threshold value (th) is selected such that its classifier results in 5% false reject rate. That is, 95% of the time the pixel is correctly classified as belonging to background. Therefore, The Thresholds th for each pixel should adapt to a value where:

$$\sum_{x:\theta_t^B(x)\geq th} \theta_t^B(x) \geq 0.95 \tag{13}$$

This can be seen in Figure 4, where (a) shows an arbitrary frame of a sequence containing a water surface and (b) shows the trained threshold map for this frame. Darker pixels in Figure 4(b) represent smaller threshold values and lighter pixels correspond to larger threshold values. As it can be observed, the thresholds in the areas that tend to change more, such as the water surface, are lower than in those areas with less amount of change, such as the sky. This is because for pixels which change all the time, the certainty about the background probability values is less.
For the other set of thresholds κ, we similarly use a measure of changes in the intensity at each pixel position. Therefore the threshold κ is proportional to the logarithm of the background model variance:

$$\kappa \approx \ln\left\{\sum_{x=0}^{255}\left(\theta_t^B(x) - \text{mean}[\theta^B(x)]\right)\right\} \tag{14}$$

This ensures that for pixels with more changes, higher threshold values are chosen for classification, while for those pixels with fewer changes smaller thresholds are employed. It should be mentioned that in the current implementation of the algorithm, the thresholds are updated every 30 frames (kept as the background buffer and used to perform the adaptation process).

More in depth evaluation of the RM technique for novelty detection and its experimental results on synthetic data and real videos will be presented in the future sections. The RM is also compared intensively with the SVDDM as well as the traditional background modeling approaches.

4. The Support Vector Data Description Modeling

In this section a powerful technique in describing the background pixel intensities, called Support Vector Data Description Modeling is presented, (Tavakkoli, Nicolescu & Bebis, 2007). Single-class classifiers, also known as novelty detectors are investigated in the literature, (Bishop, 1994). Our method trains single class classifiers for each pixel in the scene as their background model. The backbone of the proposed method is based on describing a data set using their support vectors, (Tax & Duin, 2004). In the following, details of the SVDDM and the algorithm which detects foreground regions based on this technique are presented.

4.1 The theory

A normal data description gives a closed boundary around the data which can be represented by a hyper-sphere (i.e. $F(R,a)$) with center a and radius R, whose volume should be minimized. 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:

$$F(R,a) = R^2 + C\sum_i \epsilon_i \|x_i - a\|^2 \leq R^2 + \epsilon_i \tag{15}$$

subject to:

$$\|x_i - a\|^2 \leq R^2 + \epsilon_i \quad \forall i. \tag{16}$$

In order to have a flexible data description kernel functions $K(x_i,x_j) = \Phi(x_i) \cdot \Phi(x_j)$ are used. After applying the kernel and using Lagrange optimization the SVDD function becomes:

$$L = \sum_i \alpha_i K(x_i,x_i) - \sum_{i,j} \alpha_i \alpha_j K(x_i,x_j) \tag{17}$$
$$\forall \alpha_i : 0 \leq \alpha_i \leq C$$

Only data points with non-zero α_i are needed in the description of the data set, therefore they are called *support vectors* of the description. After optimizing (17) the Lagrange multipliers should satisfy the normalization constraint $\sum_i \alpha_i = 1$.
Optimizing equation (17) is a Quadratic Programming (QP) problem. Generally the SVDD is used to describe large data sets. In such applications optimization via standard QP techniques becomes intractable. To address this issue several algorithms have been proposed which employ faster solutions to the above QP problem.

4.2 The algorithm

The methodology described in section 4.1 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:

- Unlike existing statistical modeling techniques, the proposed method explicitly addresses the single-class classification problem.

```
1. Initialization; C , Trn_No, σ
2. For each frame t
   For each pixel x(i,j)
      2.1. Training stage % OC(i,j) = 1- class classifier for pixel (i,j)
           SVD(i,j)← Incrementally train(x_t(i,j)) % SVD: The Description
      2.2. Classification stage % Desc(i,j) = classification values
           Desc(i,j)← Test(x_t(i,j),OC(i,j))
           Label pixel based on Desc(i,j).
      2.3. Update stage
           Re-train classifiers every 30 frames
```

Fig. 5. The SVDDM algorithm.

- The proposed method has less memory requirements compared to non-parametric density estimation techniques, in which 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*.
- The accuracy of this method is not limited to the accuracy of the estimated probability density functions for each pixel.
- 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 the data by fixing the false positive tolerance of the system.

Figure 5 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's 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 $\sigma = 5$. The optimal value for these parameters can be estimated by a cross-validation stage.

4.2.1 The Training Stage

In order to generate the background model for each pixel the SVDDM method uses a number of training frames. The background model in this technique is the description of the data samples (color and/or intensity values). The data description is generated in the training stage of the algorithm. In this stage, for each pixel a SVDD classifier is trained using the training frames, detecting support vectors and the values of Lagrange multipliers.
The support vectors and their corresponding Lagrange multipliers are stored as the classifier information for each pixel. This information is used for the classification step of the algorithm. The training stage can be performed off-line in cases where there are not global changes in the illumination or can be performed in parallel to the classification to achieve efficient results.

4.2.2 The Incremental SVDD Training Algorithm

Our incremental training algorithm is based on the theorem proposed by Osuna et al. in Osuna et al. (1997). According to Osuna a large QP problem can be broken into series of

[1] The proposed method is implemented in MATLAB 6.5, using Data Description toolbox (Tax, 2005).

smaller sub-problems. The optimization converges as long as at least one sample violates the KKT conditions.
In the incremental learning scheme, at each step we add one sample to the training working set consisting of only support vectors. Assume we have a working set which minimizes the current SVDD objective function for the current data set. The KKT conditions do not hold for samples which do not belong to the description. Thus, the SVDD converges only for the set which includes a sample outside the description boundary.
The smallest possible sub-problem consists of only two samples (Platt, 1998b). Since only the new sample violates the KKT conditions at every step, our algorithm chooses one sample from the working set along with the new sample and solves the optimization. Solving the QP problem for two Lagrange multipliers can be done analytically. Because there are only two multipliers at each step, the minimization constraint can be displayed in 2-D. The two Lagrange multipliers should satisfy the inequality in (17) and the linear equality in the normalization constraint.
We first compute the constraints on each of the two multipliers. The two Lagrange multipliers should lie on a diagonal line in 2-D (equality constraint) within a rectangular box (inequality constraint). Without loss of generality we consider that the algorithm starts with finding the upper and lower bounds on α_2 which are $H = min(C, \alpha_1^{old} + \alpha_2^{old})$ and $L = max(0, \alpha_1^{old} + \alpha_2^{old})$, respectively. The new value for α_2^{new} is computed by finding the maximum along the direction given by the linear equality constraint:

$$\alpha_2^{new} = \alpha_2^{old} + \frac{E_1 - E_2}{K(x_2, x_2) + K(x_1, x_1) - 2K(x_2, x_1)} \tag{18}$$

where E_i is the error in evaluation of each multiplier. The denominator in (18) is a step size (second derivative of objective function along the linear equality constraint). If the new value for α_2^{new} exceeds the bounds it will be clipped ($\hat{\alpha}_2^{new}$). Finally, the new value for α_1 is computed using the linear equality constraint:

$$\alpha_1^{new} = \alpha_1^{old} + \alpha_2^{old} - \alpha_2^{new} \tag{19}$$

4.2.3 The Classification Stage

In this stage for each frame, its pixels are used and evaluated by their corresponding classifier to label them as background or foreground. To test each pixel $\mathbf{z}_t$, the distance to the center of the description hyper-sphere is calculated:

$$\|\mathbf{z}_t - \mathbf{a}\|^2 = (\mathbf{z}_t \cdot \mathbf{z}_t) - 2\sum_i \alpha_i (\mathbf{z}_t \cdot \mathbf{x}_i) + \sum_{i,j} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{20}$$

A pixel is classified as a background pixel if its distance to the center of the hyper-sphere is less than or equal to R:

$$\|\mathbf{z}_t - \mathbf{a}\|^2 \leq R^2 \tag{21}$$

R is the radius of the description. Therefore, it is equal to the distance of each support vector from the center of the hyper-sphere:

$$R^2 = (\mathbf{x}_k \cdot \mathbf{x}_k) - 2\sum_i \alpha_i (\mathbf{x}_i \cdot \mathbf{x}_k) + \sum_{i,j} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{22}$$

Note that in the implementation of the algorithm, since the boundaries of the data description are more complicated than a hyper-sphere, a kernel is used to map the training samples into a

Memory Req.	Intensity	Chrominance	Intensity+Chrominance
Bytes per pixel	1024	2048	3072

Table 1. Per-pixel memory requirements for the RM method.

higher dimension. As the result the mapped samples in the higher dimension can be described by a high dimensional hyper-sphere and the above discussion can be used.

5. Performance Evaluation

This section presents an evaluation of the performance of the RM as well as the SVDDM techniques in terms of memory requirements, speed, and other relevant parameters.

5.1 The RM Evaluation

In this section the RM method performance is evaluated. As it will be discussed later the RM method memory requirements and computation cost are independent of the number of training samples. This property makes the RM method a suitable candidate to be used in scenarios where the background changes are very slow.

5.1.1 Parameters

In the RM method there are 5 parameters: the learning and forgetting rate α and β, thresholds th and κ, and the bandwidth Σ. As described earlier in this chapter these parameters are trained and estimated from the data to generate an accurate and robust model. The reason that the RM technique is robust is that it uses most of the information in the data set and there is no limit on the number of training samples. With all parameters being automatically updated, the system performance does not require manually chose values for these parameters.

5.1.2 Memory requirements

- *Using only intensity values.*

  Since the model is a 1-D function representing the probability mass function of the pixel,it only needs 256×4 bytes per pixel to be stored. Notice that in this case, for each pixel the intensity values are integer numbers. If the memory of the system is scarce larger bin sized can be used by quantizing the intensity values.

- *Using chrominance values.*

  In this case the model is 2-D and needs $256^2 \times 4$ bytes in memory. The current implementation of the RM method uses a simple assumption of independence between color features which results in 8×256 bytes memory requirements (Tavakkoli et al., 2006c).

Table 1 shows the memory requirements in bytes per pixel for the RM method, using intensity, chrominance values and their combinations, respectively. In conclusion the asymptotic memory requirement of the RM algorithm is large but constant $O(1)$.

5.1.3 Computation cost

- *Using only intensity values.*

  If we only use pixel intensity values for pixels according to equation (1) we need 256 addition and 2×256 multiplication operations. Both the kernel and the model range from 0 to 255.

Operations	Addition	Multiplication	Asymptotic
Intensity	256	512	$O(1)$
Chrominance	512	1024	$O(1)$

Table 2. Per-pixel computational cost for the RM method.

Memory Req.	Intensity	Chrominance	both	asymptotic
Bytes per pixel	$f(C,\sigma) \times 5 \geq 10$	$f(C,\sigma) \times 8 \geq 24$	$f(C,\sigma) \geq 32$	$O(1)$
No. of SVs	$f(C,\sigma) \geq 2$	$f(C,\sigma) \geq 3$	$f(C,\sigma) \geq 4$	$O(1)$

Table 3. Per-pixel memory requirements for the SVDDM method.

- *Using chrominance values.*
  Similarly, if we use 2-D chrominance values as pixel features and use the independence assumption discussed earlier, the system requires only 2×256 addition and 4×256 multiplication operations to update the model.

Table 2 summarizes the per-pixel computational cost of the RM algorithm using only intensity values or red/green chrominance values for each pixel. The asymptotic computation cost for this system is constant, $O(1)$, since the updating process merely consists of adding two functions. Note that this technique does not need to compute the exponential function and acts as an incremental process. The algorithm is inherently fast and an efficient implementation runs in real-time reaching frame rates of 15 frames per second (fps).

5.2 The SVDDM Evaluation

In this section the SVDDM performance in terms of memory requirements and computation cost is discussed. The key to evaluate the performance of this technique is to analyze the optimization problem solved by the system to find support vectors.

5.2.1 Parameters

In order to generate the data description, a hyper-sphere of minimum size containing most of the training samples is constructed to represents the boundary of the known class. The training has three parameters including the number of training samples N, the trade off factor C and the Gaussian kernel bandwidth σ. As mentioned in Section 4.2 for all of the experiments the values for C and σ are taken 0.10 and 5, respectively. This leaves the system with only the number of frames as a scene-dependent parameter.

5.2.2 Memory requirements

It is not easy to answer how many data samples are required to find an accurate description of a target class boundary. It not only depends on the complexity of the data itself but also on the distribution of the outlier (unknown) class. However, there is a trade-off between the number of support vectors and the description accuracy. In that sense, a lower limit can be found for the number of samples required to describe the coarsest distribution boundary.
In theory, only $d+1$ support vectors in d dimensions are sufficient to construct a hyper-sphere. The center of the sphere lies within the convex hull of these support vectors.

- *Using only intensity values.*
  Since by using intensity for each pixel there is only one feature value, the support vectors are 1-D and therefore the minimum number of support vectors required to describe

Training Set Size	Incremental[1] SVDD	Online[2] SVDD	Canonical[3] SVDD
100	0.66	0.73	1.00
200	1.19	1.31	8.57
500	2.19	2.51	149.03
1000	4.20	6.93	1697.2
2000	8.06	20.1	NA
n	$O(1)$	$\Omega(1)$	$O(n)$

1- (Tavakkoli, Nicolescu, M., Nicolescu & Bebis, 2008)
2- (Tax & Laskov, 2003)
3- (Tax & Duin, 2004)

Table 4. Speed comparison of the incremental, online and canonical SVDD.

the data will be 2. For each support vector 2 bytes are required to store the intensity and 8 bytes to store the Lagrange multipliers, requiring at least 10 bits per pixel.

- *Using chrominance values.*
  By using red and green chrominance values, c_r and c_g, the minimum of 3 support vectors are needed to be used. This requires at least 24 bytes per pixel.

The above reasoning provides a lower limit on the number of support vectors. In practical applications this lower limit is far from being useful for implementation. However, notice that the number of support vectors required to sufficiently describe a data set is related to the target description accuracy. Therefore, the memory requirement of the SVDDM method is independent of the number of training frames. Table 3 shows memory requirements in bytes per pixel for the SVDDM method using intensity, chrominance values and their combinations, respectively. The asymptotic memory requirement of the SVDDM algorithm is $O(1)$.

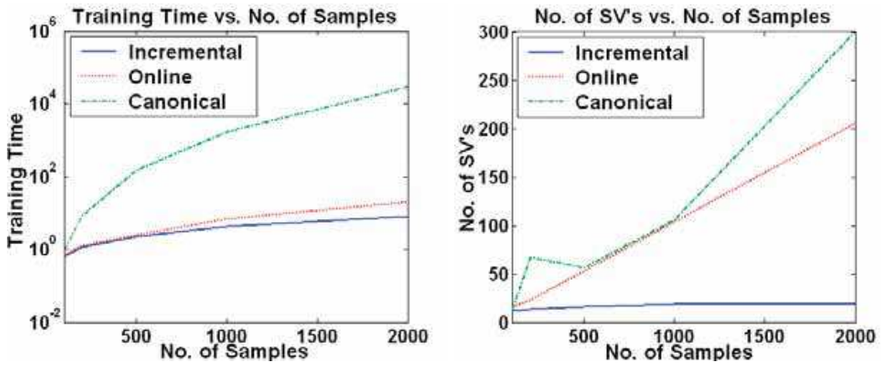
5.2.3 Computation cost

Training the SVDDM system for each pixel needs to solve a quadratic programming (QP) optimization problem. The most common technique to solve the above QP is the Sequential Minimal Optimization (Platt, 1998c); (Platt, 1998a), runing in polynomial time $O(n^k)$.
In order to show the performance of the proposed incremental training method and its efficiency we compare the results obtained by our technique with those of the online SVDD (Tax & Laskov, 2003) and canonical SVDD (Tax & Duin, 2004).
The SVVD Training Speed. In this section we compare the speed of incremental SVDD against its online and canonical counterparts. The experiments are conducted in Matlab 6.5 on a P4 Core Duo processor with 1GB RAM. The reported training times are in seconds. Table 4 Shows the training speed of the incremental SVDD, online and canonical versions on a data set of various sizes. The proposed SVDD training technique runs faster than both canonical and online algorithms and its asymptotic speed is linear with the data set size. As expected, both online and our SVDD training methods are considerably faster than the canonical training of the classifier. Notice that the training time of a canonical SVDD for 2000 training points is not available because of its slow speed.
Number of Support Vectors. A comparison of the number of retained support vectors for our technique, canonical, and online SVDD learning methods is presented in Table 5. Both online and canonical SVDD training algorithm increase the number of support vectors as the size

Training Set Size	Incremental[1] No. of SV's	Online[2] No. of SV's	Canonical[3] No. of SV's
100	12	16	14
200	14	23	67
500	16	53	57
1000	19	104	106
2000	20	206	NA
n	$O(1)$	$O(n)$	$O(n)$

1- (Tavakkoli, Nicolescu, M., Nicolescu & Bebis, 2008)
2- (Tax & Laskov, 2003)
3- (Tax & Duin, 2004)

Table 5. The number of support vectors retained.



(a) training speed (b) number of support vectors

Fig. 6. Speed and the number of support vectors comparison between the canonical learning ($\cdot-$ curve), the online learning ($--$ curve), and the incremental method ($-$ line).

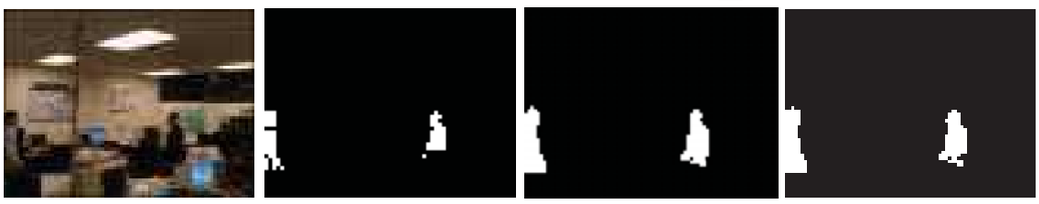
of the data set increases. However, our method keeps almost a constant number of support vectors. This can be interpreted as mapping to the same higher dimensional feature space for any given number of samples in the data set.
Notice that by increasing the number of training samples the proposed SVDD training algorithm requires less memory than both online and canonical algorithms. This makes the proposed algorithm suitable for applications in which the number of training samples increase by time. Since the number of support vectors is inversely proportional to the classification speed of the system, the incremental SVDD classification time is constant with respect to the number of samples compared with the canonical and the online methods. Figure 6 (a) and (b) shows the training speed and the number of retained support vectors, respectively.

6. Experimental Results and Comparison

In this section the performances of our approaches on a number of challenging videos are discussed and their results are compared with those of existing methods in the literature. A number of challenging scenarios are presented to the algorithms and their ability to handle issues are evaluated. The comparisons are performed both qualitatively and quantitatively.

(a) (b) (c) (d)

Fig. 7. Rapidly fluctuating background: (a) *Handshake* video sequence. Detected foreground regions using (b) AKDE. (c) RM. (d) SVDDM.



(a) (b) (c) (d)

Fig. 8. Low contrast videos: (a) *Handshake* video sequence. Detected foreground regions using (b) AKDE. (b) RM. (d) SVDDM.

6.1 Foreground Detection in Videos

This section compares the performance of the proposed techniques using several real video sequences that pose significant challenges. Their performances are also compared with the mixture of Gaussians method (Stauffer & Grimson, 2000), the spatio-temporal modeling presented in (Li et al., 2004) and the simple KDE method (Elgammal et al., 2002). We use different scenarios to test the performance of the proposed techniques and to discuss where each method is suitable. In order to have a unified comparison and evaluation we use a baseline system based on Adaptive Kernel Density Estimation (AKDE) (Tavakkoli et al., 2006b). The following a several scenarios which the comparisons and evaluations are performed on.

6.1.1 Rapidly fluctuating backgrounds

Our experiments showed that for videos where possible fluctuations in the background occur in about 10 seconds, the AKDE technique needs less memory and works faster compared to the RM and SVDDM.

Figure 7 shows the detection results of the AKDE, RM and the SVDDM algorithms on the *Handshake* video sequence. From this figure the AKDE performs better than both the RM and the SVDDM. Note that in this particular frame the color of foreground objects is very close to the background in some regions. The SVDDM technique results in very smooth and reliable foreground regions but may result in missing some parts of the foreground which are very similar to the background. Moreover, all methods successfully modeled the fluctuations seen on monitors as a part of the background.

(a) (b) (c) (d)

Fig. 9. Slowly changing background: (a) *Water* video sequence. Detected foreground region using (b) AKDE. (c) RM. (d) SVDDM.

6.1.2 Low contrast videos

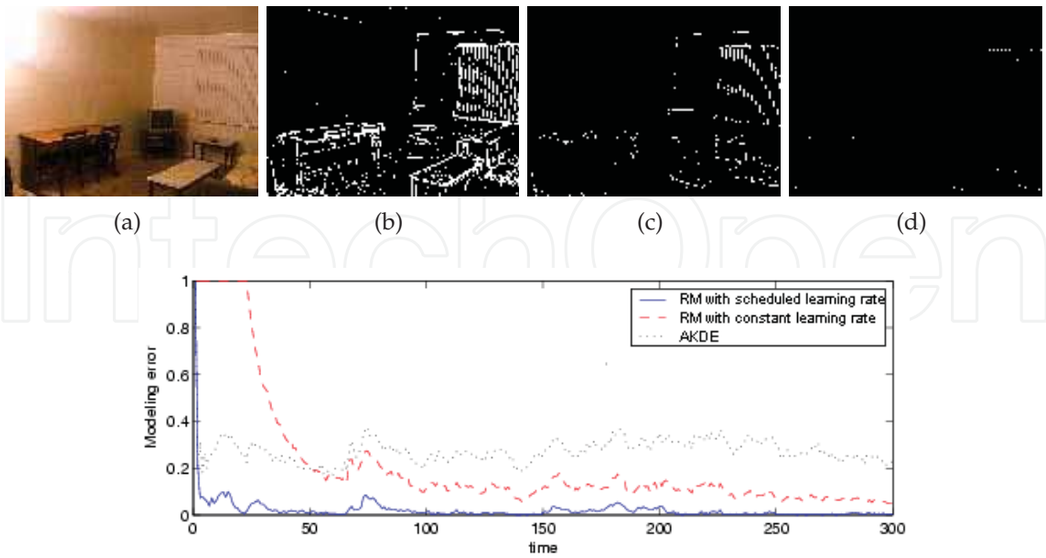
To evaluate the accuracy of the SVDDM technique in low contrast video sequences and to compare it with the AKDE technique, the experiment is performed on the *Handshake* video sequence. Figure 8 shows a frame where the background and foreground colors are different. In this experiment the quality of the images in video sequence are decreased by blurring the video. The accuracy of the foreground regions detected using the SVDDM technique is clearly better than those of the AKDE method. The reason is that the SVDDM fixes the false reject rate of the classifier. This produces a description without estimating the probability density function of the background.

6.1.3 Slowly changing backgrounds

In videos with slowly changing backgrounds the AKDE requires more training frames to generate a good background model. Therefore the system memory requirements is increased resulting in drastic decrease in its speed. In these situations the RM technique is a very good alternative, since its performance is independent of the number of training frames.
Figure 9(a) shows an arbitrary frame of the *Water* video sequence. This example is particularly difficult because waves do not follow a regular motion pattern and their motion is slow. From Figure 9, the AKDE without any post-processing results in many false positives while the detection results of the RM and the SVDDM which uses more training sample are far better.
We can conclude that the RM method has a better performance compared to both the AKDE and the SVDDM in situations in which the background has slow and irregular motion. The AKDE employs a sliding window of limited size which may not cover all changes in the background. The model is continuously updated in the RM method therefore keeping most of the changes that occurred in the past. The SVDDM method performs better than the AKDE technique in this scenario because the model that the SVDDM builds automatically generates the decision boundaries of the background class instead.

6.1.4 Hand-held camera

In situations when the camera is not completely stationary, such as the case of a hand-held camera, the AKDE and the current batch implementation of the SVDDM methods are not suitable. In these situations there is a consistent, slow and irregular global motion in the scene. These changes can not be modeled by a limited size sliding window of training frames. In such cases the RM method outperforms other tecniques.
Figure 10 shows the modeling error of the RM method in the *Room* video sequence. In Figure 10(a) an arbitrary frame of this video is shown. Figure 10(b)-(d) show the false positives de-



(a) (b) (c) (d) (e)

Fig. 10. Hand-held camera: (a) *Room* video sequence. False positives after (b) 2 frames, (c) 32 frames, ans (d) 247 frames using the AKDE method (e) Modeling error in a hand-held camera situation using different methods.

tected as foreground regions using the RM method. As expected early into the video the RM models are not very accurate resulting in a lot of false positives. However, as more and more frames are processed the model becomes more and more accurate (Figure 10(d)). Figure 10(e) compares the modeling error of the RM with and without scheduling as well as the AKDE (constant window size).

6.1.5 Non-empty backgrounds

In situations in which the background of the video is not empty (that is, there is no clear background at any time in the video sequence), the AKDE and SVDDM methods fail to accurately detect the foreground regions. In these situations the RM technique has to be used.

(a) (b) (c)

Fig. 11. Non-empty background: (a) *Mall* video sequence. (b) Background model after 5 frames using the RM method. (c) Background model after 95 frames using the RM method.

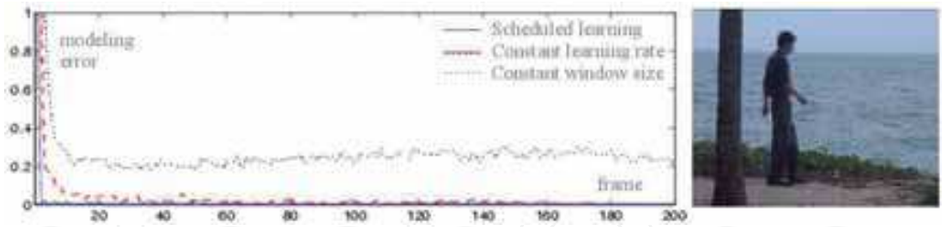


Fig. 12. Convergence speed.

Figure 11 shows the background model in the *Mall* video sequence in which the background is almost never empty. In the RM method however, the background model is updated every frame from the beginning of the video. When an object moves, the new pixel information is used to update the background model and converges to the new one. Figure 11(b) shows the background model after 5 frames from the beginning of the video and Figure 11(c) shows the model after 95 frames into the scene. The model converges to the empty background since each background pixel is covered by moving people only a short time compared to the length of the time it is not covered.

6.1.6 The RM convergence speed

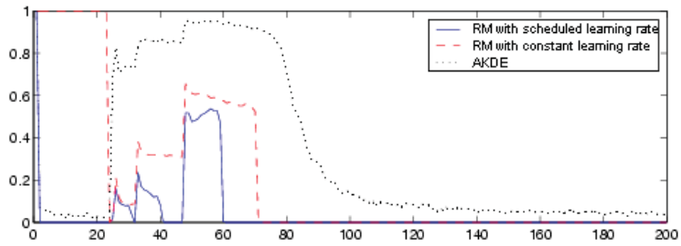
An important issue in the recursive learning is the convergence speed of the system (how fast the model converges to the actual background). Figure 12 illustrates the convergence speed of the RM with scheduled learning rate, compared to constant learning and kernel density estimation with constant window size. In this figure the modeling error of the RM with scheduled learning and constant learning rate as well as the AKDE modeling error are plotted against frame number. From Figure 12, the AKDE modeling error (the black $(-\cdot)$ curve) drops to about 20% after about 20 frames – the training window size. The modeling error for this technique does not converge to 0 since the constant window size does not cover all of the slow changes in the background. In contrast, the error for an RM approach decreases as more frames are processed. This is due to the recursive nature of this algorithm and the fact that every frame contributes to the generation and update of the background model. The effect of the scheduled learning proposed in section 3 can be observed in Figure 12.

6.1.7 Sudden global changes

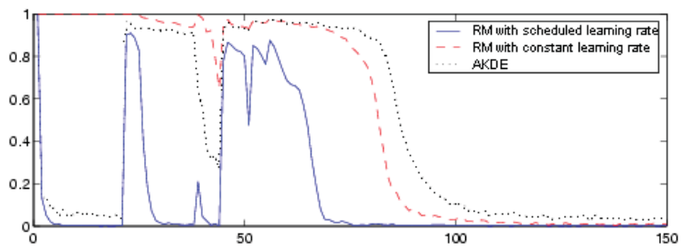
In situations where the video background suddenly changes, such as lights on/off, the proposed RM technique with scheduled learning recovers faster than the AKDE method. Generally, with the same speed and memory requirements, the RM method results in faster convergence and lower modeling error.

Figure 13 shows the comparison of the recovery speed from an expired background model to the new one. This happens in the *Lobby* video sequence when the lights go off (Figure 13(a)) or they go on (Figure 13(b)). In our example, lights go from on to off through three global, sudden illumination changes at frames 23, 31 and 47 (Figure 13(c)). The Figure shows that the scheduled learning RM method (solid curve) recovers the background model after these changes faster than non-scheduled RM and AKDE with constant window size. The constant, large learning rate recovers much slower (dashed curve) and the AKDE technique (dotted curve) is not able to recover even after 150 frames. A similar situation with lights going from off to on through three global, sudden illumination changes is shown in Figure 13(d).

(a) (b)



(c)



(d)

Fig. 13. Sudden global changes in the background: (a) the *Lobby* video sequence with lights on. (b) Lights off. (c) Recovery speed comparison in lights turned off scenario. (d) Recovery speed comparison in lights turned on scenario.

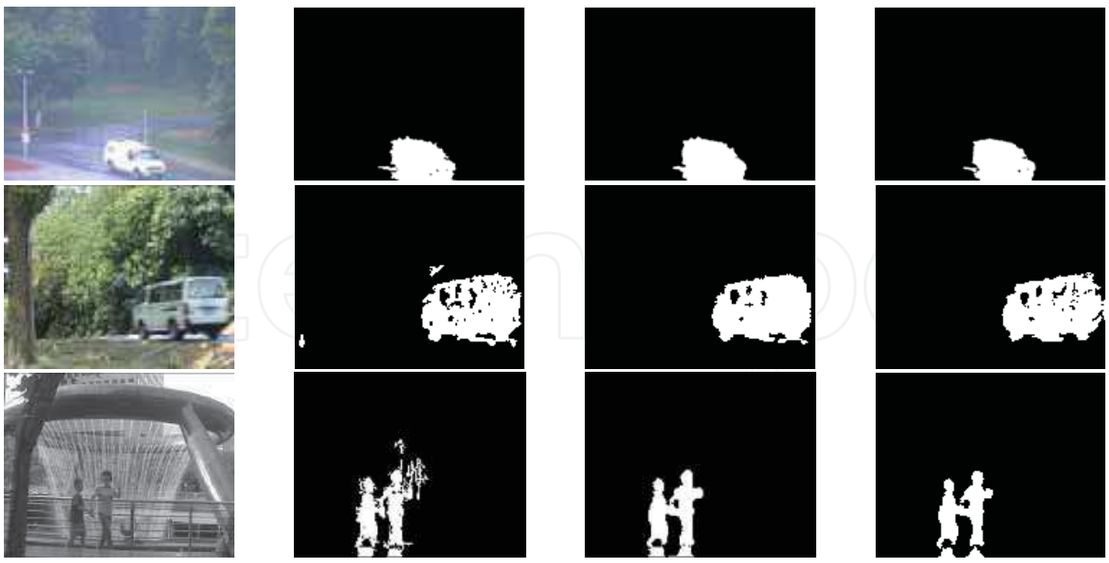
6.1.8 Other difficult examples

Figure 14 shows three video sequences with challenging backgrounds. In column (a) the original frames are shown; while columns (b), (c), and (d) show the results of the AKDE, the RM and the SVDDM methods, respectively. In this figure, from top row to the bottom; heavy rain, waving tree branches, and the water fountain pose significant difficulties in detecting accurate foreground regions.

6.2 Quantitative Evaluation

Performances of our proposed methods, RM and SVDDM are evaluated quantitatively on randomly selected samples from different video sequences, taken from (Li et al., 2004).

To evaluate the performance of each method a value "called similarity" measure is used. The similarity measure between two regions $\mathcal{A}$ (detected foreground regions) and $\mathcal{B}$ (ground



(a) (b) (c) (d)

Fig. 14. Other difficult examples: (a) Original frame. Detected foreground region using (b) AKDE. (c) RM. (d) SVDDM.

truth) is defined by (Li et al., 2004):

$$\mathcal{S}(\mathcal{A},\mathcal{B}) = \frac{\mathcal{A} \cap \mathcal{B}}{\mathcal{A} \cup \mathcal{B}} \tag{23}$$

This measure increases monotonically with the similarity between detected masks and the ground truth, ranging between 0 and 1. By using this measure we report the performance of the AKDE method, the RM method, the SVDDM, the spatio-temporal technique presented in (Li et al., 2004), and the mixture of Gaussians (MoG) in (Stauffer & Grimson, 2000). By comparing the average of the similarity measure over different video sequences in Table 6, we observed that the RM and the SVDDM methods outperform other techniques. This also shows that the AKDE, RM and SVDDM methods work consistently well on a wide range of video sequences. The reason for such desirable behavior lies under the fact that these techniques automatically deal with the novelty detection problem and do not need their parameters to be fine-tuned for each scenario.
However, from this table one might argue that AKDE does not perform better than the method presented in (Li et al., 2004). The reason is that in (Li et al., 2004) the authors used a morphological post-processing stage to refine their detected foreground regions while the results shown for the AKDE are the raw detected regions. By performing a morphological post-processing on the results obtained by the AKDE it is expected that the average similarity measure increase.

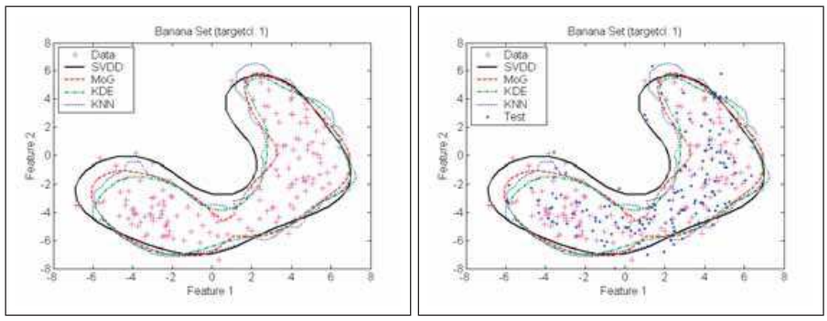
6.3 Synthetic Data Sets

We used a synthetic data set, which represents randomly distributed training samples with an unknown distribution function (*Banana* data set). Figure 15 shows a comparison between different classifiers. This experiment is performed on 150 training samples using the support

Video / Method	MR	LB	CAM	SW	WAT	FT	Avg. $\mathcal{S}(\mathcal{A},\mathcal{B})$
RM	0.92	0.87	0.75	0.72	0.89	0.87	0.84
SVDDM	0.84	0.78	0.70	0.65	0.87	0.80	0.77
Spatio-Temp[1]	0.91	0.71	0.69	0.57	0.85	0.67	0.74
MoG [2]	0.44	0.42	0.48	0.36	0.54	0.66	0.49
AKDE[3]	0.74	0.66	0.55	0.52	0.84	0.51	0.64

1: (Li et al., 2004)
2: (Stauffer & Grimson, 2000)
3: (Tavakkoli et al., 2006b)

Table 6. Quantitative evaluation and comparison. The sequences are *Meeting Room, Lobby, Campus, Side Walk, Water* and *Fountain*, from left to right from (Li et al., 2004).



(a) (b)

Fig. 15. 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.

vector data description (SVDDM), the mixture of Gaussians (MoG), the kernel density estimation (AKDE) and a 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 AKDE classifier is considered $\sigma = 1$. For the KNN we used 5 nearest neighbors. In the SVDDM classifier the Gaussian kernel bandwidth is chosen to be 5.
Figure 15(a) shows the decision boundaries of different classifiers on 150 training samples from the *Banana* data set. As it can be seen from Figure 15(b), SVDDM 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. Therefore this should be classified as the known class.

Method	SVDDM	MoG	AKDE	KNN
FRR	0.1067	0.1400	0.1667	0.1333
RR	0.8933	0.8600	0.8333	0.8667

Table 7. Comparison of False Reject Rate and Recall Rate for different classifiers.

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

$$\text{FRR} = \frac{\text{\#Missed targets}}{\text{\#Samples}} \qquad \text{RR} = \frac{\text{\#Correct predictions}}{\text{\#Samples}} \tag{24}$$

Table 7 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. From the above example, the FRR for SVDDM is less than that of the other three classifiers, while its RR is higher. This proves the superiority of this classifier for the purpose of novelty detection.

Method	SVDDM	MoG	AKDE	KNN	RM
Memory needs (bytes)	1064	384	4824	4840	1024

Table 8. Comparison of memory requirements for different classifiers.

Table 8 shows memory requirements for each classifier. Since in SVDDM we do not need to store all the training data, as can be seen from the table, it requires much less memory than the KNN and KDE methods. Only the MoG and the RM methods need less memory than the SVDDM technique. However, the low memory requirements of the RM are achieved by coarse quantization of the intensity value.

6.3.1 Classification comparison

Table 9 compares the classification error, the F_1 measure, as well as the training and the classification asymptotic time for various classifiers. The incremental training of the SVDD reaches good classification rates compared to the other methods. The trade-off parameter is set to be $C = 0.1$ in SVDD. Kernel bandwidth for the three SVDD methods and the Parzen window is $\sigma = 3.8$. $K = 3$ is selected for the number of Gaussians in the MoG and number of nearest neighbors in the K-NN method. The F_1 measure combines both the recall and the precision rates of a classifier:

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{25}$$

Classifier	Error	F_1	Training	Classification
Proposed	0.015	0.992	$O(1)$	$O(1)$
Batch SVD	0.100	0.947	$O(N)$	$O(N)$
Online SVD	0.103	0.945	$O(N)$	$O(N)$
KDE(Parzen)	0.114	0.940	$O(N)$	$O(N)$
MoG	0.143	0.923	$O(1)$	$O(1)$
K-means	0.150	0.919	$O(1)$	$O(1)$

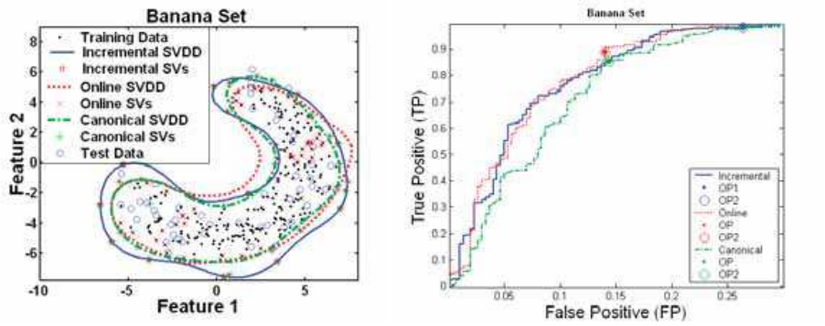
Table 9. Comparison of the classification error, F_1 measure, and asymptotic speeds with various classifiers on a *complex data set* of size 1000.

Training	Data Set	Error	F_1	No. SV's	Time
Banana	Proposed	0.005	0.997	19	4.2
	Online	0.075	0.961	104	6.9
	Canonical	0.085	0.956	106	1697
Ellipse	Proposed	0.013	0.993	6	3.72
	Online	0.100	0.947	105	4.1
	Canonical	0.110	0.994	108	2314
Egg	Proposed	0.065	0.966	8	3.85
	Online	0.095	0.950	101	3.7
	Canonical	0.128	0.932	87	1581

Table 10. Comparison of the incremental SVDD training algorithm with, online and batch methods on *Banana*, *Ellipse* and *Egg* data sets of size 1000.

6.3.2 Error evaluation

Table 10 compares the classification error, the F_1 measure, the number of the support vectors, and the learning time for the three learning methods. The experiments are performed on three data sets ('*Banana*', '*Ellipse*', '*Egg*') with 1000 training samples and 1000 test samples.

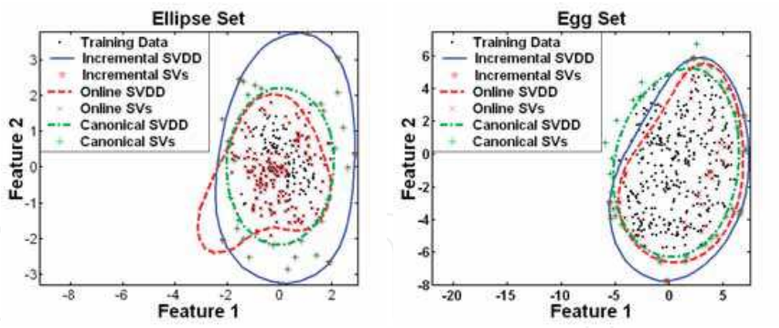


(a) (b)

Fig. 16. Comparison of incremental with canonical and online SVDD: (a) Classification boundaries . (b) Receiver Operating Curve (ROC).

6.3.3 Classification boundaries and Receiver Operating Curves

In Figure 16 (a) the classification boundaries of the three SVDD training algorithms are shown. In this figure the blue dots are the training samples drawn from the *Banana* data set and the circles represent the test data set drawn from the same probability distribution function.
The $\star$, $\times$, and $+$ symbols are the support vectors of the Incremental, Online and Canonical SVDD training algorithms, respectively. The proposed incremental learning had fewer support vectors compared to both online and canonical training algorithms. From Figure 16 (a) the decision boundaries of the classifier trained using the Incremental algorithm (solid curve) is objectively more accurate than those trained by Online (dotted curve) and Canonical (dashed curve) methods.
Figure 16 (b) shows the comparison between the Receiver Operating Curve (ROC) of the three algorithms. The solid curve is the ROC of the Incremental learning while dotted and dashed



(a) (b)

Fig. 17. Comparison of incremental with online and canonical SVDD: (a) Normal data set. (b) Complex (egg) data set.

	AKDE	RM	SVDDM	Spatio-temporal[1]	MoG[2]	Wallflower[3]
Automated	Yes	Yes	Yes	No	No	No
Classifier	Bayes	MAP	SVD	Bayes	Bayes	K-means
Memory req.*	$O(n)$	$O(1)$	$O(n)$	$O(n)$	$O(1)$	$O(n)$
Comp. cost*	$O(n)$	$O(1)$	$O(n)$	$O(n)$	$O(1)$	$O(n)$

$*$: Per-pixel memory requirements or computational cost
n: number of training frames or training features used per pixel
1 : (Li et al., 2004)
2 : (Stauffer & Grimson, 2000)
3 : (Toyama et al., 1999)

Table 11. Comparison between the proposed methods and the traditional techniques.

curves represent the Online and the Canonical learning algorithms, respectively. In this figure the operating point (OP) of the three ROC's (for the given trade-off value) are represented by the circle and the dot symbols. The true positive rate for the incremental SVDD is higher than the others. Therefore, the proposed method – under the same conditions – has higher precision and recall rates.
Figure 17 shows a comparison of the classification boundaries, and the support vectors between the three SVDD training algorithms. The classification boundaries on a 2-D normal distribution (Figure 17(a)) and a more complex distribution function in 2-D (Figure 17 (b)) are extracted using the three SVDD algorithms. From the figure the incremental SVDD results in more accurate classification boundaries than both online and canonical versions.

6.4 Comparison Summary

Table 11 provides a comparison between different traditional methods for background modeling in the literature and our methods. The SVDDM explicitly deals with the single-class classification. Other methods shown in the table – except the RM – use a binary classification scheme and use heuristics or a more sophisticated training scheme to make it useful for the single-class classification problem of background modeling. The RM method which has the adaptive threshold updating mechanism solves this issue and acts as a novelty detector.

	AKDE	RM	SVDDM	Spatio-temp	MoG	Wallflower
Low contrast	S*	NS**	S	NS	NS	NS
Slow changes	NS	S	S	S	S	S
Rapid changes	S	S	S	S	NS	S
Global changes	NS	S	NS	S	S	NS
Non-empty	NS	S	NS	S	S	S
Hand-held camera	NS	S	NS	NS	NS	NS

∗ : Suitable
∗∗: Not suitable

Table 12. Scenarios where each method appears to be particularly suitable.

Table 12 shows different scenarios and illustrates where each method is suitable for foreground region detection. As expected the RM method is suitable for a wide range of applications except when the contrast of images in the video is low. From this table, the only method suitable for the hand-held camera scenario is the RM. The other methods fail to build a very long term model for the background because of the fact that their cost grows with the number of training background frames.

7. Conclusion

In this chapter the idea of applying a novelty detection approach to detect foreground regions in videos with quasi-stationary is investigated. In order to detect foreground regions in such videos the changes of the background pixel values should be modeled for each pixel or a groups of pixels. In the traditional approaches the pixel models are generally statistical probabilities of the pixels belonging to the background. In order to find the foreground regions the probability of each pixel in new frames being a background pixel is calculated from its model. A heuristically selected threshold is employed to detect the pixels with low probabilities.
In this chapter two approaches are presented to deal with the single class classification problem inherent to foreground detection. By employing the single class classification (novelty detection) approach the issue of heuristically finding a suitable threshold in a diverse range of scenarios and applications is addressed. These approaches presented in this chapter are also extensively evaluated. Quantitative and qualitative comparisons are conducted between the proposed approaches and the state-of-the-art, employing synthetic data as well as real videos. The proposed novelty detection mechanisms have their own strengths and weaknesses. However, the experiments show that these techniques could be used as complimentary to one another. The establishment of a universal novelty detection mechanism which incorporates the strengths of both approaches can be considered as a potential future direction in this area.

8. References

Bishop, C. (1994). Novelty detection and neural network validation., *In IEE proceedings on Vision, Image and Signal Processing. Special Issue on Application of Neural Networks.* **141**(4): 217–222.

Elgammal, A., Duraiswami, R., Harwood, D. & Davis, L. (2002). Background and foreground modeling using nonparametric kernel density estimation for visual surveillance., *In proceedings of the IEEE* **90**: 1151–1163.

Li, L., Huang, W., Gu, I. & Tian, Q. (2004). Statistical modeling of complex backgrounds for foreground object detection., *IEEE Transactions on Image Processing.* **13**(11): 1459–1472.

McKenna, S., Raja, Y. & Gong, S. (1998). Object tracking using adaptive color mixture models., *In proceedings of Asian Conference on Computer Vision* **1**: 615–622.

Mittal, A. & Paragios, N. (2004). Motion-based background subtraction using adaptive kernel density estimation., *In proceedings of CVPR* **2**: 302–309.

Osuna, E., Freund, R. & Girosi, F. (1997). Improved training algorithm for support vector machines, *In Proc. Neural Networks in Signal Processing* .

Platt, J. (1998a). *Advances in Kernel Methods - Support Vector Learning.*, Editors: B. Schlkopf, C. Burges, A. J. Smola, MIT Press.

Platt, J. (1998b). Fast training of support vector machines using sequential minimal optimization, *Advances in Kernel Methods - Support Vector Learning.* **MIT Press**: 185–208.

Platt, J. (1998c). Sequential minimal optimization: A fast algorithm for training support vector machines, *Microsoft Research Technical Report MSR-TR-98-14* .

Pless, R., Brodsky, T. & Aloimonos, Y. (2000). Detecting independent motion: The statistics of temporal continuity., *IEEE Transactions on PAMI* **22**(8): 68–73.

Pless, R., Larson, J., Siebers, S. & Westover, B. (2003). Evaluation of local models of dynamic backgrounds., *In proceedings of the CVPR* **2**: 73–78.

Stauffer, C. & Grimson, W. (1999). Adaptive background mixture models for real-time tracking., *In proceedings of CVPR* **2**: 246–252.

Stauffer, C. & Grimson, W. (2000). Learning patterns of activity using real-time tracking., *IEEE Transactions on PAMI* **22**(8): 747–757.

Tavakkoli, A., Kelley, R., King, C., Nicolescu, M., Nicolescu, M. & Bebis, G. (2007). A vision-based approach for intent recognition, *In Proceedings of the 3rd International Symposium on Visual Computing* .

Tavakkoli, A., Nicolescu, M. & Bebis, G. (2006a). An adaptive recursive learning technique for robust foreground object detection., *In proceedings of the International Workshop on Statistical Methods in Multi-image and Video Processing (in conjunction with ECCV06)* .

Tavakkoli, A., Nicolescu, M. & Bebis, G. (2006b). Automatic statistical object detection for visual surveillance., *In proceedings of IEEE Southwest Symposium on Image Analysis and Interpretation* pp. 144–148.

Tavakkoli, A., Nicolescu, M. & Bebis, G. (2006c). Robust recursive learning for foreground region detection in videos with quasi-stationary backgrounds., *In proceedings of 18th International Conference on Pattern Recognition* .

Tavakkoli, A., Nicolescu, M. & Bebis, G. (2007). A support vector data description approach for background modeling in videos with quasi-stationary backgrounds., *to appear in the International Journal on Artificial Intelligence Tools* .

Tavakkoli, A., Nicolescu, M. & Bebis, G. (2008). Efficient background modeling through incremental support vector data description, *In Proceedings of the 19th International Conference on Pattern Recognition* .

Tavakkoli, A., Nicolescu, M., Bebis, G. & Nicolesu, M. (2008). Non-parametric statistical background modeling for efficient foreground region detection, *International Journal of Machine Vision and Applications* pp. 1–16.

Tavakkoli, A., Nicolescu, M., M., Nicolescu & Bebis, G. (2008). Incremental svdd training: Improving efficiency of background modeling in videos, *In Proceedings of the 10th IASTED Conference on Signal and Image Processing* .

Tax, D. (2005). Ddtools, the data description toolbox for matlab. version 1.11.

Tax, D. & Duin, R. (2004). Support vector data description., *Machine Learning* **54**(1): 45–66.
Tax, D. & Laskov, P. (2003). Online svm learning: from classification and data description and back., *Neural Networks and Signal Processing* (1): 499–508.
Toyama, K., Krumm, J., Brumitt, B. & Meyers, B. (1999). Wallflower: principles and practice of background maintenance., *In proceedings of ICCV* **1**: 255–261.
Wern, C., Azarbayejani, A., Darrel, T. & Pentland, A. (1997). Pfinder: real-time tracking of human body., *IEEE Transactions on PAMI* **19**(7): 780–785.

IntechOpen

Pattern Recognition
Edited by Peng-Yeng Yin

ISBN 978-953-307-014-8
Hard cover, 568 pages
Publisher InTech
Published online 01, October, 2009
Published in print edition October, 2009

For more than 40 years, pattern recognition approaches are continuingly improving and have been used in an increasing number of areas with great success. This book discloses recent advances and new ideas in approaches and applications for pattern recognition. The 30 chapters selected in this book cover the major topics in pattern recognition. These chapters propose state-of-the-art approaches and cutting-edge research results. I could not thank enough to the contributions of the authors. This book would not have been possible without their support.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen