2006

An Online Discriminative Approach to Background Subtraction

Li Cheng

Shaojun Wang
Wright State University - Main Campus, shaojun.wang@wright.edu

Terry Caelli

An Online Discriminative Approach to Background Subtraction

Li Cheng
National ICT Australia
li.cheng@nicta.com.au

Shaojun Wang, Dale Schuurmans
Department of Computing Science
University of Alberta, Canada
(swang),(dale)@cs.ualberta.ca

Terry Caelli, S.V.N. Vishwanathan
National ICT Australia
(terry.caelli),(SVN.Vishwanathan)@nicta.com.au

Abstract

We present a simple, principled approach to detecting foreground objects in video sequences in real-time. Our method is based on an on-line discriminative learning technique that is able to cope with illumination changes due to discontinuous switching, or illumination drifts caused by slower processes such as varying time of the day. Starting from a discriminative learning principle, we derive a training algorithm that, for each pixel, computes a weighted linear combination of selected past observations with time-decay. We present experimental results that show the proposed approach outperforms existing methods on both synthetic sequences and real video data.

1 Introduction

A key problem in video analysis is, given a relatively static camera, to detect the moving objects in image sequences. This task provides fundamental low-level visual cues that are necessary for further analysis. Traditionally, separating foreground from background in video analysis has been referred to as *background subtraction*. Essentially, the problem is to classify each pixel in a given frame into one of two categories: either part of a foreground object, or part of the background scene.

Many existing background subtraction algorithms have their roots in the 20th century film industry, where it was observed that a background scene could be recovered by exposing a film long enough to allow the moving objects to wash out. This simple observation holds in many practical situations, and combined with the static camera assumption, motivates the *pixel process* approach to background subtraction that tracks the temporal variation of a single pixel over the time. However, a naive pixel process approach based on fixed uni-modal distributions cannot cope with natural situations, such as those presented in Fig. 1 (drifting, jumping, and shaking), where the temporal variations of pixels are drawn from multi-modal and changing distributions.

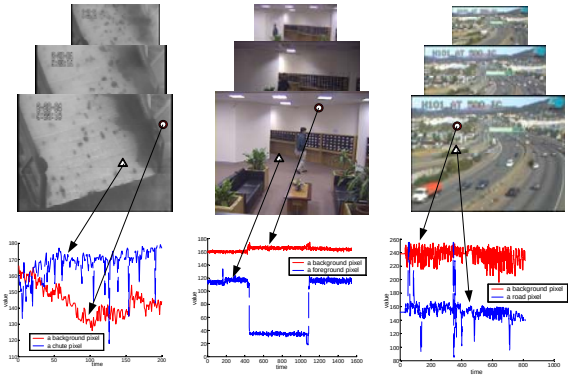


Figure 1: Three representative video sequences are presented that covers various situations, where for each video sequence, three key frames are presented in the upper panels. The left displays the rock video where the ore rocks falling through the rejection chute. The middle one is an indoor sequence with lights switched on and off. The right sequence contains a road traffic taken by a shaking camera. The three panels on the bottom present the temporal dynamics of selected pixel values for the corresponding scenarios.

To cope with more realistic situations, many background subtraction algorithms have been developed over the past fifteen years. Although taking various forms, they tend to follow a common scheme: since the foreground distribution is unknown, one instead maintains a temporal model of the relatively static background scene, and detects foreground objects as outliers in the background distribution. Such an approach leads to great effort in properly estimating the background distribution, which might drift over the time. Existing methods of this form can be generally classified into two categories: those that build generative models

for the background scene, and those model the background distribution with a non-parametric form.

The simplest, but still effective, *generative* approaches are Adjacent Frame Difference (FD) [13] and Mean-Filter (MF) [13]. As its name suggests, FD operates by subtracting a current pixel value from its previous value, marking it as foreground if absolute difference is greater than a threshold. MF computes an on-line estimate of a Gaussian distribution from a buffer of recent pixel values, and marks the current value as foreground if its deviation from the mean is more than a threshold. Wren et al. [15] and Friedman et al. [4] proposed to use a single Gaussian, and a mixture of Gaussians (MoG) respectively, to model the density of a background pixel. Later, [12, 7] considered recursive updating schemes for the mixture and Gaussian parameters $\{w, \mu, \sigma\}$, to maintain a currently relevant model. To handle a background that exhibits structured changes, such as waves through water or trees, two research groups [16, 9] have proposed to incorporate principal component analysis (PCA) in MoG models.

Non-parametric methods maintain a background distribution for a pixel based on a list of past examples. Various kernel density estimates (KDEs) have been used for this purpose in the past [3], including Parzen windows [2], and more recently KDEs with variable bandwidth [8]. A major drawback of these approaches, however, is that they ignore the time-series nature of the problem. Moreover, KDE requires training data from a sequence of examples that have a relatively 'clean' background. These shortcomings can be partially remedied by using a sequential approach [5] that uses an iterative algorithm to predict the modes of the KDEs in an on-line fashion.

In general, what we seek is a principled background subtraction algorithm for *real-time* video analysis that: (1) is computationally efficient, (2) accurate, and (3) can rapidly adapt to dynamic situations where the foreground background distributions change with time. Motivated by these goals, and influenced by the work of online learning and one-class support vector machines (1-SVM) [10, 6], we propose an *Implicit online Learning with Kernels* (**ILK**) that exploits kernels to address this problem.

On-line discriminative learning Let d be the dimensionality of the input space, and define a kernel mapping $k(\cdot,\cdot)$ from input space to a Hilbert feature space, $\mathbb{R}^d \rightarrow \mathcal{H}$ as $x \mapsto k(x,\cdot) \in \mathcal{H}$. Here $x \in \mathbb{R}^d$ is an example and $\mathcal{H}$ denotes the reproducing kernel Hilbert space (RKHS) with induced kernel $k(\cdot,\cdot)$ such that $f(x) = \langle k(x,\cdot), f(\cdot)\rangle_{\mathcal{H}}$, and $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ gives the inner product. To simplify notation, we use $\langle x,\cdot\rangle_{\mathcal{H}}$ to denote $\langle k(x,\cdot),\cdot\rangle_{\mathcal{H}}$. The norm in this case is naturally defined as $||\cdot||_{\mathcal{H}} =< \cdot,\cdot >_{\mathcal{H}}^{1/2}$.

Assume over the time, we observe a sequence of T examples $(x_t)_{t=1}^T$. In our approach, a separating function $f(\cdot)$ is to be predicted as *a weighted combination of examples* $f : (x_t) \mapsto \sum_t \alpha_t k(x_t,\cdot)$, where past examples are associated with different weights $(\alpha_t)_{t=1}^T$ (with a time decay) derived formally from the large margin principle. Unfortunately, past examples still have to be stored when evaluating the separating hyperplane $f \in \mathcal{H}$, which violates the constraints for real time video analysis. To reduce the time and memory requirements, we propose a modification to obtain kernel classifiers based on limited support N, with $N << T$, and approximates f by heuristically truncating past examples with insignificant weights.

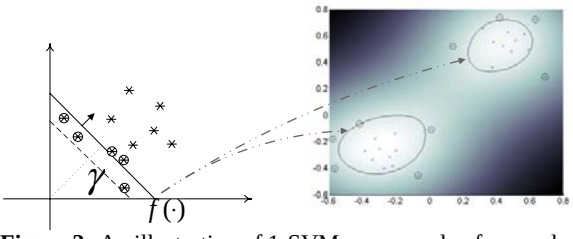


Figure 2: An illustration of 1-SVM on a sample of examples that motivates the proposed approach. The left panel depicts one example of feature space with the separating hyperplane $f(\cdot)$ (the solid lines), while the right panel shows the corresponding scenarios in input space, where the preimage of $f(\cdot)$ behaves as a nonlinear function. The crosses (in the left panels) and the corresponding dots (in the right panels) represent observed examples accumulated during a certain amount of time. γ is the radius measuring the distance of the separating hyperplane from origin.

The benefits of the proposed approach are two fold. First, it is simple to implement, and fast in evaluation time with reasonable space complexity. Moreover, experiments empirically demonstrates its good performance. Second, it provide an alternative viewpoint of background subtraction, which helps in understanding the nature of the problem.

2 The Proposed Approach

1-SVM and the Proposed Risk Functional

The proposed discriminative approach is motivated by the large-margin principle [14] and especially 1-SVM [10] for stationary distributions, where we are interested in predicting the optimal separating hyperplane $f(\cdot)$ in some high-dimensional feature space. When ignoring the sequential nature of the examples sequence $(x_t)_{t=1}^T$, Fig. 2 presents one example of such scenarios where the set of past examples scattered in input space (right panel, as a mixture of two Gaussians) and in corresponding feature space (left panel). Regardless of the shape of the background distribution, we estimate, in a proper feature space, a separating hyperplane f that encloses as many background examples as possible in the input space. The examples are then partitioned into the background data and the outliers accordingly. Notice that $f(\cdot)$ are linear operators in the RKHS space (straight lines in left panels), and due to the mapping function ϕ, the preimage of f is a nonlinear function in the input space (curved lines in right panel). Further, the func-

tion $f(\cdot)$ is determined solely from a weighted average of the support vectors, shown as circular points in Fig.2, with zero influence from the remaining examples. Therefore, the proposed algorithm predicts a separating boundary $f(\cdot)$ that is determined solely by a collection of the support vectors. The presence of noise in the real world would make it almost impossible to find such a well-separated hyperplane $f(\cdot)$.

In video surveillance, one observes a sequence of images. Each pixel over the time forms a sequence of examples S, where the example x_t is observed at time t. For the pixel process at time $t+1$, we would like to predict a (non-stationary) separating hyperplane f from the large-margin principle. Intuitively, the new f should (a) be not far from its past predictor, f_t; (b) be bounded in term of its norm $\|f\|_{\mathcal{H}}$; (c) satisfies the following 1-SVM type loss function:

$$\begin{aligned} L(x_t, f) &= (1 - f(x_t))_+ \\ &= (1 - \langle f, k(x_t, \cdot)\rangle)_+, \end{aligned} \tag{1}$$

where $(\cdot)_+ \triangleq \max\{\cdot, 0\}$. Define $\lambda \geq 0$ as a regularization parameter, and and $C > 0$ a constant. The risk functional is then formulated as

$$R(f) = \underbrace{\frac{1}{2}\|f - f_t\|^2_{\mathcal{H}}}_{R_{\mathrm{div}}(f)} + \underbrace{\frac{\lambda}{2}\|f\|^2_{\mathcal{H}}}_{\lambda R_{\mathrm{cap}}(f)} + \underbrace{CL(x_t, f)}_{R_{\mathrm{inst}}(f)}, \tag{2}$$

which consists of three terms: (a) the divergence risk, $R_{\mathrm{div}}(f)$, measures the distance of predicted f from previous prediction f_t. (b) the capacity risk, $R_{\mathrm{cap}}(f)$, controls the complexity of the prediction f. (c) the instantaneous risk, $R_{\mathrm{inst}}(f)$ which is the 1-SVM type loss function (1) times a constant. We would like to choose f minimizing $R(f)$. The following lemma which is an immediate result of the well known representer theorem (Theorem 4.2 of Scholkopf et al. [11]), tells us the representation form that f will take.

Lemma 2.1. *Let R_{cap} be a composite of functions as $R_{\mathrm{cap}} = \Omega(\|f\|_{\mathcal{H}})$ where $\Omega : [0, \infty) \to \mathbb{R}$ is a strictly monotonic increasing function. Let $R_{\mathrm{inst}} : \mathcal{H} \to \mathbb{R}$ and $R_{\mathrm{div}} : \mathcal{H} \to \mathbb{R}$ be strictly convex functions. Then there exists a global minimizer $f \in \mathcal{F}$ for the regularized risk $R_{\mathrm{div}}(f) + \lambda R_{\mathrm{cap}}(f) + R_{\mathrm{inst}}(f)$, which admits a representation of the form*

$$f(\cdot) = \sum_{i=1}^{t} \alpha_i k(x_i, \cdot) \qquad \forall \alpha_i \geq 0 \tag{3}$$

Robust Estimate of the Weights α

As a consequence of the above lemma, the optimal f of problem (2) equals a weighted combination of past examples in the kernel space. Further, the weights $\{\alpha_i\}_{i=1}^t$ is explicitly obtained by the following theorem.

Theorem 2.2. *In the online learning scenario, at time t, the global minimizer of the risk functional (2) is unique, and admits*

$$f_{t+1}(\cdot) = \sum_{i=1}^{t} \alpha_i k(x_i, \cdot) \tag{4}$$

where

$$\begin{cases} \alpha_i & \Leftarrow (1-\tau)\alpha_i \quad \forall i < t \\ \alpha_t & \Leftarrow \begin{cases} \hat{\alpha}_t & \text{if } \hat{\alpha}_t \in [0, (1-\tau)C], \\ 0 & \text{if } \hat{\alpha}_t < 0, \\ (1-\tau)C & \text{if } \hat{\alpha}_t > (1-\tau)C, \end{cases} \end{cases} \tag{5}$$

and

$$\hat{\alpha}_t = \frac{1 - (1-\tau) f_t(x_t)}{k(x_t, x_t)}.$$

The proof (detailed in [1]) turns out to be rather straightforward, after introducing the pair of variables, $\tau, \upsilon \in \mathcal{R}_+$, such that

$$\lambda = \frac{\tau}{1-\tau}. \tag{6}$$

To ease the notation, also denote

$$\hat{L}_t = 1 - (1-\tau) f_t(x_t). \tag{7}$$

The resultant representation formula of Theorem 2.2 is efficient for predicting f_{t+1}, since we only have to compute α_t taking into account the most recently observed example x_t, while the weights for those examples that occur a while ago are naturally less-weighted by an exponential damping term. That is,

$$\begin{aligned} f_{t+1}(\cdot) &= \textstyle\sum_{i=1}^{t} (1-\tau)^{t-i} \alpha_i k(x_i, \cdot) \\ &= (1-\tau) f_t(\cdot) + \alpha_t k(x_t, \cdot). \end{aligned} \tag{8}$$

Analysis of the parameters By solving the optimization problem, we end up with an update form of the weight α_t that is always upper bounded by $(1-\tau)C$. This formulation ensures limited influence from outliers. Besides, the damping rate $\tau \in [0, 1)$ mainly balances the preference between the capacity risk term $R_{cap}(f)$ and the rest risk terms.

The ILK algorithm

The proposed ILK algorithm is able to cope with stationary background distributions, as described previously. In particular, when the examples are drawn from a time-varying distribution $P(t)$, the algorithm can still predict a reasonable separating hyperplane f, as long as the damping rate τ matches the drifting speed of $P(t)$. However, In ILK, the kernel expansion of the estimated f, as shown in Eq. (8), requires entire past examples. In many practical domains, especially in real-time applications, storing and manipulating all the past frames turns out to be non-realistic as the amount of examples increases as $t \to \infty$. This leads to **SILK**, a heuristic modification that aims at approximating

Algorithm 1: The (**ILK**) Algorithm

Input *The radius γ, cut-off value C, threshold ϵ, damping rate τ*
Initialize $f_1 \Leftarrow \underline{0}$
for $t \leftarrow 1$ **to** T **do**
  Receive example x_t, compute $\|x_t\|^2_{\mathcal{H}} = k(x_t, x_t)$;
  Compute $\langle f_t, x_t \rangle_{\mathcal{H}}$ according to Eq. (8);
  Update $\hat{L}_t$ according to Eq. (7);
  Update $(\alpha_i)_{i=1}^t$ according to Eq. (5);
  Assign label as

$$y_t = \begin{cases} 1, & \hat{L}_t \geq \epsilon; \\ 0, & otherwise. \end{cases} \tag{9}$$

end
Output *The sequences (α_t), (y_t)*

the original f by kernel expansion of a sparse set of examples.

The SILK Algorithm

SILK aims at modifying algorithm **ILK** slightly to keep a truncated set of past samples containing those with significant weights. That is, for each pixel, a buffer of fixed size N is held that contains the weight sequence $\beta \triangleq (\beta_1, \cdots, \beta_N)$ in non-decreasing order, and the associated sparse set of examples as $\mathcal{Z} \triangleq (z_1, \cdots, z_N)$.

In brief, algorithm **SILK** works as follows. Initially the buffer is empty with weights assigning to $\beta = \underline{0}$. At time t, a weight α_t is computed for the observed sample x_t, we attempt to insert α_t into the weight sequence β in buffer, and if successful, we also insert x_t into $\mathcal{Z}$. Notice that the symbol "✠" indicates the remainder of the line is a comment. For algorithmic convenience, We denote the augmented weight sequence $\overline{\beta} \triangleq (\beta_1, \cdots, \beta_N, \beta_{N+1})$ while always set $\beta_{N+1} = \infty$. Note that the function f_t evaluated at x_t is thus written as

$$\langle f_t, x_t \rangle_{\mathcal{H}} = \sum_{n=1}^{N} \beta_i k(z_n, x_t). \tag{10}$$

Let $\alpha_{i,t}$ $(i \leq t)$ denote the weight of kernel expansion $k(x_i, \cdot)$ at time t. Denote the mapping of the index of example in buffer $\hat{S}$ of size N to the set of examples S of size T $(N << T)$ as $\hat{N} : \hat{S} \rightarrow S : j \mapsto i$, where $|\hat{S}| = N$, $|S| = T$. We present the result of the analysis about the effect of truncation errors in the following lemma, its proof can be found in [1].

Lemma 2.3 (Truncation Error of **SILK**). *Let the non-truncated representation of the separating function be*

$$\tilde{f}(\cdot) = \sum_{i=1} \alpha_{i,t} k(x_i, \cdot), \tag{12}$$

and the truncated approximation is

$$f(\cdot) = \sum_{j=1, i \in \hat{N}(j)}^{N} \alpha_{i,t} k(x_i, \cdot). \tag{13}$$

Then the truncation error is upper bounded by

$$\|f - \tilde{f}\|_{\mathcal{H}} \leq \frac{(1-\tau)^{N+1}}{\tau} CR, \tag{14}$$

which decreases exponentially as the size of buffer N increases.

Algorithm 2: The **SILK** Algorithm

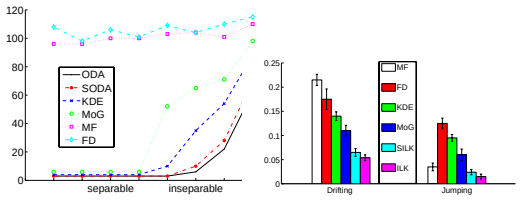
Input *The radius γ, cut-off value C, threshold ϵ, damping rate τ, and the set of sparse kernels Z with associated weights β in non-decreasing order*
Initialize $(\beta_1, \cdots, \beta_N) \leftarrow \underline{0}$, $\beta_{N+1} \leftarrow \infty$ *and* $\mathcal{Z} \leftarrow$ *random values*
for $t \leftarrow 1$ **to** T **do**
  Receive data x_t, compute $\|x_t\|^2_{\mathcal{H}} = k(x_t, x_t)$;
  Compute $\langle f_t, x_t \rangle_{\mathcal{H}}$ according to Eq. (10);
  Update $\hat{L}_t$ according to Eq. (7);
  Update α_t according to Eq. (5);
  ✠ *Insert α_t and x_t into the sorted sequences β and $\mathcal{Z}$.*;
  for $j \leftarrow 1$ **to** $N+1$ **do**
    if $\alpha_t < \beta_j$ **then**
      if $j > 1$ **then**
        Insert α_t into the sequence
        $\beta \leftarrow (\cdots, \beta_{j-1}, \alpha_t, \beta_j, \cdots)$;
        Insert x_t into the sequence
        $\mathcal{Z} \leftarrow (\cdots, z_{j-1}, x_t, z_j, \cdots)$;
        Remove β_1 from β;
        Remove z_1 from $\mathcal{Z}$;
      end
      Break;
    end
  end
  Assign label as

$$y_t = \begin{cases} 1, & \hat{L}_t \geq \epsilon; \\ 0, & otherwise. \end{cases} \tag{11}$$

end
Output β, $\mathcal{Z}$ *and the sequence* (y_t)

Complexity Analysis

By looking at the proposed algorithms above, it is easy to conclude that the SILK algorithm is both computationally more efficient and less memory demanding than ILK. To see in detail, consider making predictions on a sequence of examples $(x_t)_{t=1}^T$. The space complexity of ILK is $(d+1)T$ for the $\{(\alpha_t)_{t=1}^T, (x_t)_{t=1}^T\}$ sequences, comparing with $(d+1)N$ for the sequences $\{(\beta_i)_{i=1}^n, (Z_i)_{i=1}^n\}$ of SILK, where $N << T$. In terms of computational complexity for a sequence of length T, the running time for ILK is $O(T^2)$, comparing with $O(NT)$ for SILK.



(a) **(b)**

Figure 3: Comparisons of six background subtraction algorithms (a) on sequences of examples drawn from stationary distributions of mixture of two Gaussians, where The y-axis presents the cumulative mistakes. (b) on the drifting (left) and the jumping (right) scenarios, respectively, where The y-axis presents the average mistakes.

3 Experiments

We conduct experiments on both synthetic data sequences and real world video sequences, and compare the results of the proposed ILK/SILK algorithms with existing background subtraction methods: FD, MF, MoG and KDE. The FD and MF algorithms are implemented as stated previously, the MoG algorithm is implemented based upon [12, 7], and the KDE algorithm is adopted from [3]. Gaussian kernels are adopted for the proposed approach for all experiments, and the internal parameters of all above background subtraction algorithms are tuned for best performance.

Experiments on Synthetic Data

To explore the performance of the proposed and existing background subtraction algorithms, *three types* of two-dimensional synthetic sequences are considered, where a large quantity of background examples are maintained against a relative small amount of foreground examples in all scenarios. The first scenario is generated from *stationary* distributions of the background (and the foreground), where each consists a mixture of two Gaussians. We then consider the scenarios of drawing from changing distributions, in specific, we will investigate the *drifting* and the *jumping* background distributions. We focus on the separable scenario where all examples that would be misclassified by the Bayes optimal classifier are dropped off. In all of the three cases, a mixture of Gaussians is employed to generate the background examples, and similarly for foreground.

In Fig. 3a, each position along the horizontal axis presents one stationary distribution, while a set of stationary distributions are arranged from left to right according to the decrease of separability that ranges from separable cases to inseparable cases. The vertical axis shows the average cumulative mistakes. The curve of each algorithm presents average cumulative mistakes over 20 *iid* sequences. As such, three non-parametric algorithms – KDE, ILK and SILK – make least average cumulative mistakes. MoG makes comparably more mistakes, while MF and FD make frequent mistakes since the stationary background model is a mixture of Gaussians.

ILK and SILK seem to be good at dealing with scenarios where the sequence of examples are drawing from either drifting or jumping distributions. This is clearly revealed from Fig. 3b, where SILK makes slightly more mistakes than ILK, but are superior than other algorithms. Besides, KDE empirically perform on par with MoG on both situations.

Experiments on Video Sequences

Extensive experiments have been conducted on various scenarios of video sequences. A network of SILK machines is employed with each concentrated on tracking the temporal dynamic of *one* pixel from the video sequence, and is compared against the MoG, KDE, MF and FD algorithms.

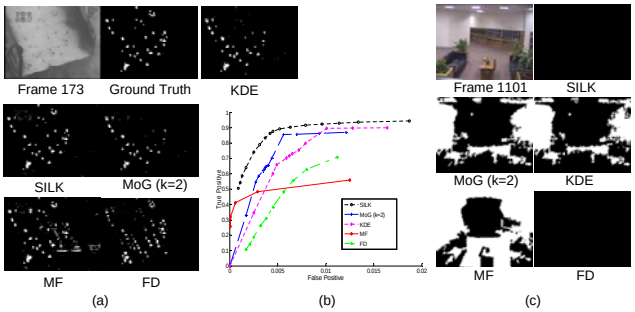


Figure 4: (a)A comparison of five algorithms on the 173th frame of the rock sequence. FD fails for this task due to the fast sliding speed of the rocks. Notice KDE and MoG miss the clumps in the red curve while SILK still detects them. (b) ROC curve of five algorithms on the indoor data. (c) A comparison on the 1101th frame of the indoor sequence when the lights are just switched off. The result of SILK and FD are shown to adapt to this jumping situation faster than the rest algorithms.

During the following comparisons, A buffer size $N = 20$ is used for SILK, 30 past frames are used for MF, $k = 2$ (or $k = 3$) is used for MoG, and the related parameters of the algorithms concerned are turned for best performance.

The rock video of Fig. 1 (left) is taken from an ore mining site, where the ore rocks are falling through the rejection chute. A grey-scale surveillance camera is mounted on top of the chute to monitor the realtime processing, where statistical information is to be collected about the number and sizes of the ore fragments passed by. The bottom-left panel presents the characteristic temporal dynamics of one background pixel and one chute pixel, where both exhibit drifting behaviors, mostly due to the spreading of the dust fog. Even for a human observer, some fragments are very hard to be distinguished from the background scene due to similar appearance. Fig. 4a displays the obtained results on the $173th$ frame, where SILK detects the true foregrounds (ore rocks) as good as MF, with much less false alarms. Obviously, both KDE and MoG miss the several clumps on the top-right corner. FD does the worst, mostly due to the fast sliding speed of the ore rocks passing through the chute.

The middle panels of Fig. 1 presents an indoor video sequence where the lights are switched off and back on. As illustrated in the bottom-middle panel, it is close to the situation with a jumping distribution. Again, the ROC curve in Fig. 4b demonstrates good overall performance of the SILK algorithm. In specific, SILK is shown to be more adaptive to this jumping situation than most of the background subtraction algorithms, as presented in Fig. 4c for frame 1101, where FPs are fixed to be close to 0.01. MF perform worst due to its slow adaption to the switch of lights.

The right panels of Fig. 1 present one traffic sequence taken by a camera that shakes irregularly. This results in a switched multi-modal distributions as demonstrated in the

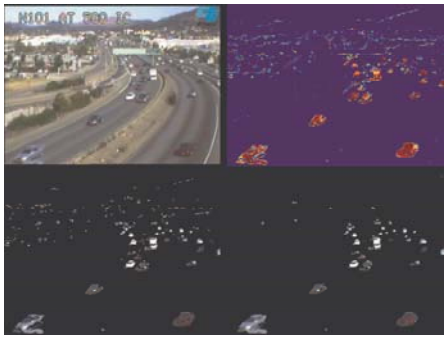


Figure 5: This figure presents, in scan-line order, the observed image, the confidence map, the result without false suppression and the result with false suppression, of the SILK algorithm on one road traffic sequence of frame 1552.

bottom-right panel. However, since the shaking motion is neither periodical nor with a constant strength, the sequence turns out to be very challenging for any background subtraction algorithm. As presented in Fig. 5b, SILK still manages to obtain satisfactory results. Further details including the ROC curve are described in [1].

4 Conclusion and Future Work

In this paper, an online discriminative approach is proposed to address a key problem in video analysis — foreground background separation. The proposed approach is derived from an online risk minimization framework, and is shown in experiments to outperform existing algorithms. The current work involves more about the temporal dynamics of the pixel processes, our future work will focus on exploiting the spatial properties of the sensor fields and the label fields to further improve the performance.

References

[1] Anonymous. In preparation. Technical report, 2006.

[2] R. Duda and P. Hart. *Pattern classification and scene analysis*. Wiley, New York, 1973.

[3] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *ECCV*, 2000.

[4] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *UAI*, 1997.

[5] B. Han, D. Comaniciu, , Y. Zhu, and L. Davis. Incremental density approximation and kernel-based bayesian filtering for object tracking. In *CVPR*, 2004.

[6] J. Kivinen, A. Smola, and R. Williamson. Online learning with kernels. *IEEE Trans. on Signal Processing*, 52(8):2165–2176, 2004.

[7] D. Lee. Effective gaussian mixture learning for video background subtraction. *PAMI*, 27(5):827–832, 2005.

[8] A. Mittal and N. Paragios. Motion-based background subtraction using adaptive kernel density estimation. In *CVPR*, 2004.

[9] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background modeling and subtraction of dynamic scenes. In *ICCV*, 2003.

[10] B. Scholkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13:1443–1471, 2001.

[11] B. Scholkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[12] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *PAMI*, 22:747–757, 2000.

[13] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practices of background maintenance. In *ICCV*, 1999.

[14] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[15] C. Wren, A. Azarbayejani, T. Darrell, and A. Pantland. Pfinder:real-time tracking of the human body. *PAMI*, 19(7):780–785, 1997.

[16] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust Kalman filter. In *ICCV*, 2003.