Realtime Background Subtraction from Dynamic Scenes

Li Cheng
TTI-Chicago
Chicago, IL, USA
licheng@tti-c.org

Minglun Gong
Memorial University of Newfoundland
St. John's, NL, Canada, A1B 3X5
gong@cs.mun.ca

Abstract

This paper examines the problem of moving object detection. More precisely, it addresses the difficult scenarios where background scene textures in the video might change over time. In this paper, we formulate the problem mathematically as minimizing a constrained risk functional motivated from the large margin principle. It is a generalization of the one class support vector machines (1-SVMs) [18] to accommodate spatial interactions, which is further incorporated into an online learning framework to track temporal changes. As a result it yields a closed-form update formula, a central component of the proposed algorithm to enable prompt adaptation to spatio-temporal changes. We also analyze the mistake bound and discuss issues such as dealing with non-stationary distributions, making use of kernels and efficient inference by a variant of dynamic programming. By exploiting the inherently concurrent structure, the proposed approach is designed to work with the highly parallel graphics processors (GPUs) to facilitate realtime analysis. Our empirical study demonstrates that the proposed approach works in realtime (over 80 frames per second) and at the same time performs competitively against state-of-the-art offline and quasi-realtime methods.

1. Introduction

A fundamental problem in video content analysis is to detect moving foreground objects from background scenes. This procedure provides necessary low-level visual cues to facilitate further analysis such as object tracking [8],, action or activity recognition, and is crucial to many applications including surveillance, human computer interaction, animation and video event analysis.

In this paper, the problem is mathematically formulated as minimizing a constrained risk functional, motivated from the large margin principle. More precisely, it is a generalization of the 1-SVMs [18] to accommodate spatial interactions, which is further incorporated into an online learning framework based on the work of *e.g*. [10, 2] to track temporal background changes. As a result it yields a closed-form update formula that is a central component of the proposed algorithm to enable prompt adaptation to spatio-temporal background changes caused by *e.g*. camera jitter and dynamic background textures. We also analyze the mistake bound and discuss issues such as dealing with non-stationary distributions, making use of kernels, and efficient inference by a variant of dynamic programming. We note in the passing that the online 1-SVM algorithm of [2] can be seen as a special case of the proposed framework. By exploiting its inherently concurrent structure, the proposed approach is designed to work with the highly parallel graphics processors (GPUs), to take advantage of this widely available and affordable computing platform. This leads to a system that is able to work in realtime: empirically it parses images at over 80 frame per second (FPS) using a middle-class GPU.

Related Work In Machine Learning Literature Vapnik and his co-workers develop the theory of support vector machines (SVMs) and advocate large margin learning principle through the seminar work of [27], which is recently extended to structured output learning [24, 26](*i.e*. the output space consists of structured objects such as sequences, strings, trees). Similarly kernel methods are introduced for various pattern analysis problems [19]. Moreover, in contrast to the usual batch learning scenario where by training from a finite set of examples, an algorithm is expected to predict well on unseen examples generated from an underlining *stationary* distribution, online learning *e.g*. [10, 1], on the other hand, takes place in a sequence of trials, thus is able to deal with cases where this distribution might change with time or for stream data such that the number of examples grows over time and might not fit into the memory — both are exactly what have been encountered in our problem. Kernel methods can similarly be incorporated into online learning, *e.g*. [9, 2]. [2] is the first attempt of using online learning framework to deal with the dynamic background changes and stream data nature of our problem, but spatial inconsistency still persists with the *i.i.d.* pixels as-

sumption. Generalizing from the online 1-SVM of [2], this paper explicitly incorporates spatial interactions of proximal pixels to account for the spatio-temporal background scene changes.

Related Work In Computer Vision Literature A variety of techniques have been presented in the vision community [5, 28, 25, 23, 4, 30, 12]. The limitation of these methods comes from the two commonly adopted assumptions: a static background scene as well as independent pixel processes. Both are often violated in real-life situations: the background scene might change over time due to *e.g*. illumination changes, waves, wind or camera jitter, while ignoring spatial dependency among neighboring pixels inevitably leads to inconsistent and noisy predictions. To accommodate temporal changes, variants of autoregression models are employed [29, 14] that however rely on the strong and often unrealistic assumption of the state spaces being linearly structured. In [17], a Lambertian model is used to address sudden illumination changes. Meanwhile, attempts have also been made to reconcile spatial correlations between neighbor pixels, including the local search of [4], the incorporation of principal component analysis (PCA) in mixture of Gaussians (MoG) models [16], and the biologically motivated center-surround method [14]. In particular, [20, 15, 3] use Markov random fields (MRF) models, which are unfortunately computationally demanding thus not suitable for real-time video analysis. In addition, most existing approaches are *generative* methods [25, 15, 3], while it has been widely accepted that *discriminative* methods often deliver better results [13]. On the other hand, while attempts have been made to utilize GPUs for efficient foreground background segmentation, existing methods (*e.g*. [7]) are still limited to handling scenes with static backgrounds.

Our Contribution The proposed algorithm (namely *online struct 1-SVM*) has three main contributions. First, the problem of background subtraction is connected to work in online learning and learning with structured output, which possess a rich literature(*e.g*. [10, 1, 24, 26]) and well-studied theoretical principles. Second, we present a new online learning algorithm that generalizes the previous 1-SVM [18] and online 1-SVM [2] to incorporate spatial interactions of neighbor nodes over the induced structured output graph. A formal mistake bound analysis is provided. This provides a principled method of modeling the spatio-temporal dependencies commonly existed in videos, as demonstrated in various real-life scenarios during empirical simulations. Third, unlike previous approaches using CPUs that often run offline or quasi-realtime, it is explicitly designed to work with GPUs. This leads to a processing speed of over 80 frames per second (FPS), sufficiently efficient for follow-up realtime analysis. In practice our realtime algorithm is shown to perform comparable or even superior to the state-of-the-art *non-realtime* methods.

2. The Online Struct 1-SVM Approach

2.1. Problem Formulation

Let $\boldsymbol{x} \in \mathcal{X}$ denote an observed image and let $\mathcal{Y}$ be the set of feasible labels. A label $\boldsymbol{y} \in \mathcal{Y}$ is defined over a grid graph $G = (V, E)$, where $i \in V$ indexes a local pixel and $(i, j) \in E$ denotes an edge connecting neighboring pixels i and j. For the ith pixel let $y = y_i \in \{-1, +1\}$ indicate whether this pixel belongs to foreground (-1) or not $(+1)$. When taking a video stream $\mathcal{T} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$ as input, the task is to detect foreground objects from these images by assigning labels $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T)$ on the fly.

Denote $\boldsymbol{w}$ a model parameter vector and $L(\boldsymbol{x}, \boldsymbol{w})$ a loss function, the problem can be abstractly casted as learning a model that incurs least accumulative losses on $\mathcal{T}$. Let us start by considering a batch learning scenario where the *entire* set of examples (or images), $\mathcal{T}$, are available and the learning procedure involves minimizing a regularized risk function

$$\min_{\boldsymbol{w}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{\eta}{T}\sum_{t=1}^{T} L(\boldsymbol{x}_t, \boldsymbol{w}), \tag{1}$$

where $\eta > 0$ is a trade-off scalar. Denote the discriminant function $\boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{y}) = \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_t, \boldsymbol{y}) \rangle$, where $\boldsymbol{\phi}(\cdot, \cdot)$ is the feature function over the joint input-output space.

Now, by the over-simplified assumption that each pixel is independent of others, *i.e*. $E = \emptyset$ for the label graph G, the loss L can be factorized as a sum of individual pixel loss as $L(\boldsymbol{x}_t, \boldsymbol{w}) = \sum_i l(\boldsymbol{x}_t, \hat{w}_i)$, where $\boldsymbol{w} = \{\hat{w}_i : i \in V\}$ with each element $\hat{w}_i$ being a parameter vector to predict the ith pixel label. Motivated from the large margin principle, a vanila 1-SVM can be used for each pixel as $l(\boldsymbol{x_t}, \hat{w}_i) = (\rho - \hat{f}_i(\boldsymbol{x}_t))_+$, where $\rho > 0$ is the margin, and $(\cdot)_+ := \max(0, \cdot)$ denotes the hinge function. In this paper, we propose a *generalization* of batch learning 1-SVM [18] to structured output

$$\begin{aligned} \min_{\boldsymbol{w}, \xi} \quad & \frac{\|\boldsymbol{w}\|^2}{2} + \frac{\eta}{T}\sum_t \xi_t \\ s.t. \quad & \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y}) \geq \Delta(\mathbf{1}, \boldsymbol{y}) - \xi_t, \qquad \forall t, \boldsymbol{y}, \end{aligned} \tag{2}$$

where $\Delta(\mathbf{1}, \boldsymbol{y})$ denotes the label distance between all one (*i.e*. all background label) and $\boldsymbol{y}$, $\Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y}) := \boldsymbol{f}(\boldsymbol{x}_t, \mathbf{1}) - \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{y})$, and $\xi_t \geq 0$ for all t. This provides the loss function as

$$L(\boldsymbol{x}_t, \boldsymbol{w}) = \sum_{\boldsymbol{y} \in \mathcal{Y}} \Big(\Delta(\mathbf{1}, \boldsymbol{y}) - \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y}) \Big)_+, \tag{3}$$

and the optimal label field configuration can be obtained by solving an integer assignment problem

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}} \quad \Delta(\mathbf{1}, \boldsymbol{y}) - \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y}). \tag{4}$$

2.2. Online Learning

We further consider an online learning framework based on that of [10, 2], where at time (or trial) t, given the current parameter $\boldsymbol{w}_t$ and presented with an example $\boldsymbol{x}_t$, we update the parameter $\boldsymbol{w}_{t+1}$ by minimizing a regularized risk function on *current* example

$$\boldsymbol{w}_{t+1} \;=\; \operatorname*{argmin}_{\boldsymbol{w}} \;\; \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}_t\|^2 + \eta L(\boldsymbol{x}_t, \boldsymbol{w}). \tag{5}$$

A label $\boldsymbol{y}$ is said to be margin violating if $\Delta(\mathbf{1}, \boldsymbol{y}) - \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y}) > 0$ with its violation magnitude as $e(\boldsymbol{y}) := \Delta(\mathbf{1}, \boldsymbol{y}) - \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y})$. This gives the set of violation labels as $\mathcal{E}_t := \{\boldsymbol{y} \;\; \text{s.t.} \;\; \Delta(\mathbf{1}, \boldsymbol{y}) - \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y}) > 0\}$. Denote $\boldsymbol{\psi}(\boldsymbol{x}_t, \boldsymbol{y}) := \boldsymbol{\phi}(\boldsymbol{x}_t, \mathbf{1}) - \boldsymbol{\phi}(\boldsymbol{x}_t, \boldsymbol{y})$, the parameter vector is then additively updated by

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \sum_{\boldsymbol{y} \in \mathcal{E}_t \cup \mathbf{1}} \alpha_{t,\boldsymbol{y}} \boldsymbol{\psi}(\boldsymbol{x}_t, \boldsymbol{y}). \tag{6}$$

When $\mathcal{E}_t \neq \emptyset$, following the implicit update principle of [2], it is easy to derive that the coefficients $\alpha_{t,\boldsymbol{y}} \in [-\eta, 0]$ for $\boldsymbol{y} \in \mathcal{E}_t$, $\alpha_{t,\mathbf{1}} = -\sum_{\boldsymbol{y} \in \mathcal{E}_t} \alpha_{t,\boldsymbol{y}} \leq \eta$ and all other $\alpha_{t,\boldsymbol{y}} = 0$. In particular, assume $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y})$ can be factored as $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) = \varphi(\boldsymbol{x}) \otimes \delta(\boldsymbol{y})$ with $\delta(\cdot)$ being the delta function and $\otimes$ the tensor product, the optimal solution to (5) becomes

$$\alpha_{t,\mathbf{1}} = \frac{\sum_{\boldsymbol{y}' \in \mathcal{E}_t}}{(\,|\,\mathcal{E}_t\,|\,+1)} \min\left\{ \frac{\Delta(\mathbf{1}, \boldsymbol{y}') - \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y}')}{\|\varphi(\boldsymbol{x}_t)\|^2}, \eta \right\} \tag{7}$$

$$\alpha_{t,\boldsymbol{y}} = \alpha_{t,\mathbf{1}} - \min\left\{ \frac{\Delta(\mathbf{1}, \boldsymbol{y}) - \Delta \boldsymbol{f}_t(\mathbf{1}, \boldsymbol{y})}{\|\varphi(\boldsymbol{x}_t)\|^2}, \eta \right\} \quad \forall \boldsymbol{y} \in \mathcal{E}_t\,. \tag{8}$$

This online learning framework naturally addresses the issue of temporal changes in video stream, and the online 1-SVM method used in [2] is in fact a special case by ignoring the edges E. It is unfortunately not practical: as the cardinality of the label graph space grows exponentially with the graph size, it is essentially prohibitive to compute the exact solutions of (7) and (8) for a reasonable sized image, since they rely on exhaustively search for every violation label in $\mathcal{E}_t$. Instead, in what follows we approximate the original optimization problem (2) by a set of decomposed constraints.

According to the graph structure of label $\boldsymbol{y}$, we can decompose the discriminant function into local (pixel and edge) parts as $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}) = \sum_i f_i(\boldsymbol{x}, y_i) + \sum_{(i,j)} f_{ij}(\boldsymbol{x}, y_i y_j)$, similarly the feature function becomes $\boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) = \sum_i \phi_i(\boldsymbol{x}, y_i) + \sum_{(i,j)} \phi_{ij}(\boldsymbol{x}, y_i y_j)$, where $f_i(\boldsymbol{x}, y_i) = \langle w_i, \psi_i(\boldsymbol{x}, y_i) \rangle$ and $f_{ij}(\boldsymbol{x}, y_i y_j) = \langle w_{i,j}, \psi_{ij}(\boldsymbol{x}, y_i y_j) \rangle$. (2) is approximated by solving

$$\begin{aligned} \min_{\boldsymbol{w}, \xi} \quad & \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}_t\|^2 + \eta \left(\sum_i \xi_i + \sum_{ij} \xi_{ij} \right) \\ s.t. \quad & f_i(\boldsymbol{x}_t, 1) - f_i(\boldsymbol{x}_t, -1) \geq \Delta_i - \xi_i, \qquad \forall i, \\ & f_{ij}(\boldsymbol{x}_t, \mathbf{1}) - f_{ij}(\boldsymbol{x}_t, y_i y_j) \geq \Delta_{ij}(\mathbf{1}, y_i y_j) - \xi_{ij}, \;\; \forall (i,j), y_i y_j, \end{aligned} \tag{9}$$

where we set $\rho := \Delta_i = 1$, $\Delta_{ij}(\mathbf{1}, y_i y_j)$ the label distance between the two edge label assignments, and $f_i(\boldsymbol{x}, y_i) = \frac{1}{2} y_i \hat{f}_i(\boldsymbol{x})$. Comparing with the original optimization problem (2), this decomposed loss induces a set of tighter constraints than the constraints of (2), as each decomposed (node or edge) discriminant function f_i (or f_{ij}) is now desired to outscore its competitors by a margin. The loss function is decomposed accordingly as

$$\begin{aligned} L_d(\boldsymbol{x}_t, \boldsymbol{w}) = & \sum_i \left(\rho - \hat{f}_i(\boldsymbol{x}_t) \right)_+ \\ & + \sum_{(i,j),(y_i y_j)} \left(\Delta_{ij}(\mathbf{1}, y_i y_j) - \left(f_{ij}(\boldsymbol{x}_t, \mathbf{1}) - f_{ij}(\boldsymbol{x}_t, y_i y_j) \right) \right)_+, \end{aligned} \tag{10}$$

Further, by assuming each local (pixel or edge) feature function $\phi_i(\boldsymbol{x}, y_i)$ or $\phi_{ij}(\boldsymbol{x}, y_i y_j)$ can be factored into a tensor product $\varphi(\boldsymbol{x}) \otimes \delta(y_i)$ (or $\varphi(\boldsymbol{x}) \otimes \delta(y_i) \otimes \delta(y_j)$), as the optimum solution of (9), the parameter vector is now updated by

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \sum_i \alpha_{t,i} \psi_i(\boldsymbol{x}_t, y_i) + \sum_{(i,j),(y_i,y_j)} \alpha_{t,ij,y_i y_j} \psi_{ij}(\boldsymbol{x}_t, y_i y_j), \tag{11}$$

with

$$\alpha_{t,i} = \min\left\{ \frac{\left(\rho - \hat{f}_i(\boldsymbol{x}_t)\right)_+}{\|\varphi_i(\boldsymbol{x}_t)\|^2}, \eta \right\}, \qquad \forall t, i \tag{12}$$

$$\alpha_{t,ij,y_i y_j} = \min\left\{ \frac{\left(\Delta_{ij}(\mathbf{1}, y_i y_j) - \left(f_{ij}(\boldsymbol{x}_t, \mathbf{1}) - f_{ij}(\boldsymbol{x}_t, y_i y_j) \right) \right)_+}{\|\varphi_{ij}(\boldsymbol{x}_t)\|^2}, \eta \right\}, \tag{13}$$

$$\forall t, (i,j), (y_i, y_j).$$

This can be viewed as an generalized version of the online 1-SVMs [2], where $E = \emptyset$ and we only compute the update of local node parts (12).

2.3. Mistake bound

Let $P(\boldsymbol{w}) := \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{\eta}{T} \sum_{t=1}^{T} L_d(\boldsymbol{x}_t, \boldsymbol{w})$ denote the optimum of the batch version of the primal risk, let $\epsilon \in$

$(0, \rho)$ be the 1-SVM threshold, and denote the minimum non-trival edge margin $\rho_e := \min_{y_i y_j \neq \mathbf{1}} \{\Delta_{ij}(\mathbf{1}, y_i y_j)\} > 0$. In what follows we provide a mistake bound.

Theorem 1. *Let $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_T, y_T)\}$ be an arbitrary sequence of observations such that $\|\phi_i(\boldsymbol{x}_t, y_i)\|^2 \leq X^2$ and $\|\phi_{ij}(\boldsymbol{x}_t, y_{ij})\|^2 \leq X^2$ hold for any t, i, (i,j), $\boldsymbol{x}$ and $\boldsymbol{y}$. The number of mistakes M made by Algorithm 1 is at most*

$$\frac{\min_{\boldsymbol{w}} P(\boldsymbol{w})}{\frac{\rho-\epsilon}{2}|V| \min\{\eta, \frac{\rho-\epsilon}{X^2}\} + \frac{\rho_e}{2}|E| \min\{\eta, \frac{\rho_e}{X^2}\}}. \tag{14}$$

Proof. (**sketch**) Denote the Lagrange dual up to trial t as $D(\alpha^t)$, $\mathcal{M}$ the trials on which the algorithm makes mistakes. Following the proof technique of [22], as a consequence of weak duality and the fact $D(\alpha^0) = 0$, the telescoping sum of the dual progresses is upper bounded as $\min_{\boldsymbol{w}} P(\boldsymbol{w}) \geq \max_\alpha D(\alpha) \geq \sum_{t \in \mathcal{T}} (D(\alpha^t) - (\alpha^{t-1}))$. In addition, the dual can be additively decomposed into local parts as $D(\alpha^t) = \sum_i D_i(\alpha^t) + \sum_{i,j} D_{i,j}(\alpha^t)$ and we can compute the dual progress of each local part at each trial. Putting these together and rearranging terms yield the desired result. □

Algorithm 1 The Online Struct 1-SVM Algorithm

Input: The trade-off value η and video stream $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T$
Output: labels $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_T$
$\boldsymbol{w}_0 \Leftarrow \underline{0}$
for $t = 1$ to T **do**
  Observe image $\boldsymbol{x}_t$
  Update $\boldsymbol{w}_t$ according to Equations (11), (12) and (13)
  Predict label $\boldsymbol{y}^*$ using Equation (4)
end for

2.4. Dealing with Non-stationary Distributions

So far we have considered the stationary scenario where examples (images) are drawn in hindsight from the same distribution. In practice, however, the examples might drift slowly and we would like to track them. This can be accommodated by extending (5) as

$$\boldsymbol{w}_{t+1} = \operatorname*{argmin}_{\boldsymbol{w}} \ \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}_t\|^2 + \left(\frac{\lambda}{2}\|\boldsymbol{w}\|^2 + c \cdot L(\boldsymbol{x}_t, \boldsymbol{w})\right), \tag{15}$$

where η is incorporated into the two constants $\lambda, c > 0$. As a result it leads to a geometrical decay of previous estimates: $\boldsymbol{w}_{t+1} = (1-\tau)\boldsymbol{w}_t + \sum_i \alpha_{t,i}\psi_i(\boldsymbol{x}_t, y_i) + \sum_{(i,j),(y_i,y_j)} \alpha_{t,ij,y_i y_j}\psi_{ij}(\boldsymbol{x}_t, y_i y_j)$ with $\tau := \frac{\lambda}{1+\lambda}$.

2.5. Kernels

To make use of the powerful kernel methods, the proposed framework can be lifted to reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ by letting $\boldsymbol{w} \in \mathcal{H}$, with the defining kernel $k : (\mathcal{X} \times \mathcal{Y})^2 \to \mathbb{R}$ satisfying the reproducing property, $\langle \boldsymbol{w}, k((\boldsymbol{x}, \boldsymbol{y}), \cdot)\rangle_{\mathcal{H}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y})$. The celebrated representer theorem guarantees that $\boldsymbol{w}$ can be expressed uniquely as $\boldsymbol{w}_{t+1} = \sum_{i=1}^t \sum_{\boldsymbol{y}} \alpha_{i,\boldsymbol{y}} k((\boldsymbol{x}_i, \boldsymbol{y}), \cdot)$, and the kernels can be decomposed into local parts as well.

The embrace of kernels however introduces the issue of support vectors which usually grow linearly with the number of examples up to time t. Rather than storing all past examples which is prohibitively expensive, we truncate the function expansion by following the strategy of [2] to maintain a set of buffers with fixed size, $\{\mathcal{B}_i, \mathcal{B}_{ij}\}$, each dedicates one local part and each starts from empty. Once a buffer $\mathcal{B}_i$ (or $\mathcal{B}_{ij}$)'s size limit ω is exceeded, the local part with the lowest coefficient value is discarded.

2.6. Dynamic programming (DP) inference

The inference problem (4) can be equivalently represented as its decomposed form

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}} \sum_i \Big(\Delta_i(1, y_i) + f_i(\boldsymbol{x}, y_i)\Big) + \sum_{(i,j)} f_{ij}(\boldsymbol{x}, y_i y_j), \tag{16}$$

where $\Delta_i(1, y_i) = \rho$ if $y_i = -1$ and 0 otherwise. Graph cuts might be the first option to solve this binary labeling problem. However, our edge functions f_{ij} may not satisfy the submodular constraint, which then becomes a NP-hard problem for a graph cuts algorithm to find the optimal solution [11]. Although several variants of graph cuts have been devised to deal with the non-submodular functions [11], they tend to be much more complicated thus difficult to be made parallel. Rather we adopted a variant of DP that dedicated to efficient image processing [6]. This method performs two DP passes along orthogonal scanline directions, which helps to remove the streak artifacts suffered by most existing DP type algorithms. In brief, the first DP pass searches for optimal solution of each individual horizontal scanline and selects a limited set of reliable label assignments, which are then used to guide the second DP pass performing along vertical scanlines. While the original algorithm assumes the spatial interactions follow the multi-class Pott's model, we extend the algorithm to deal with our non-submodular edge functions f_{ij}, as well as simplify the computation to solve our binary labeling problem more efficiently. In particular, we utilize a two-dimensional cost array S to store the partial optimal solutions. That is, $S[i, y]$ is the cost of the optimal solution for the first i pixels, given that label y is assigned to pixel i. Now the iterative function

for optimal path search is defined as:

$$S[j,+1] = \max\Big\{S[i,+1] + f_{ij}\big(\boldsymbol{x},(+1,+1)\big), \\ S[i,-1] + f_{ij}\big(\boldsymbol{x},(-1,+1)\big)\Big\} + f_j(\boldsymbol{x},+1), \\ S[j,-1] = \max\Big\{S[i,+1] + f_{ij}\big(\boldsymbol{x},(+1,-1)\big), \\ S[i,-1] + f_{ij}\big(\boldsymbol{x},(-1,-1)\big)\Big\} + 1 + f_j(\boldsymbol{x},-1), \quad (17)$$

with boundary conditions:

$$\begin{aligned} S[0,+1] &= f_0(\boldsymbol{x},+1), \\ S[0,-1] &= 1 + f_0(\boldsymbol{x},-1). \end{aligned} \quad (18)$$

3. Working with Graphics Processors

Graphics Processing Units (GPUs) on modern graphics cards allow developers to write their own computational kernels that are executed on multiple data (vertices or pixels) in *parallel*. Traditionally GPUs are dedicated to 3D graphics applications where the computational kernels are used to calculate the transformation and lighting of each vertex (called vertex shaders), or to compute the shading of each rasterized pixel (called pixel shaders). For general purpose applications, the computation process is normally cast as a rendering process that involves one or more rendering passes. Within each rendering pass, the following operations are sequentially performed: (1) represent the input data as 2D or 3D arrays and load them into the video memory as textures; (2) load the algorithm into the GPU as a pixel shader; (3) set either the screen or a pixel buffer in video memory as the rendering target; and (4) execute the shader by rendering a image-sized rectangle. Our goal is to process live feed video in real time. To achieve this, we carefully exploit the inherently concurrent structure of the proposed algorithm to work with the highly parallel graphics processors as described below.

Our algorithm is implemented here in three main steps: The first two steps compute the per-pixel function f_i and the per-edge function f_{ij}, respectively, while the third step uses the modified DP algorithm to solve (16). As the first two steps are very similar, in what follows we concentrate on describing the first step (computing f_i). For space concerns we refer interested readers to [6] for a detailed account of the GPU-based DP implementation, as our modified version is implemented in the same manner.

When computing per-pixel loss on the GPU, a 2D color texture of ω times image size is created as local buffers for kernel expansions. As shown in Fig. 1, this texture is used to store the support vectors (SVs) and corresponding coefficients for each pixel in the image. To retain the most important support vectors, all observation-coefficient pairs are sorted in descending order according to the absolute values of the coefficients and only those have higher weights are kept. Throughout the experiments we fix $\omega = 50$.

Figure 1. A portion of the 2D texture that holds all the pixel buffers' content of the top 5 most significant coefficients. Here the RGB channels of each pixel keep the SV observation values and the alpha channel keeps the corresponding coefficients.

When a new frame t is presented, three rendering passes are executed as follows. In the first pass, the pixel shader takes the new observation and the existing local buffers as two input textures, computes the function $\hat{f}_i(\boldsymbol{x}_t)$ as kernel expansion using the ith pixel buffer, and stores the result into a new texture. This texture is then used in the second pass to compute $\alpha_{t,i}$ using a different shader that implements (12). Finally in the third rendering pass, the local buffers are updated using the $\alpha_{t,i}$ values obtained.

4. Experiments

We evaluate the performance of the proposed algorithm (online struct 1-SVM) by comparing to the online 1-SVM of [2], the recent variant of mixture of Gaussians (MoG) [23, 30, 12] using recursive updates [30], as well as the Bayesian method of [21] (referred to as Sheikh et al.). In particular, precision-recall (PR) curves are utilized to account for the highly skewed nature of our object detection datasets. For online 1-SVM and the proposed algorithm, The same set of algorithm parameters are used during the experiments, including the RBF kernel with $\sigma = 8$, the margin $\rho = 1$, trade-off value $\eta = 0.2$, the decay factor $\tau = 0.95$, and the buffer size $\omega = 50$, and the raw image data are directly used as features. For a fair comparison, we adopt the original implementation of [30], and that of Sheikh et al. [21]. The internal parameters of these methods are also tuned to obtain good performance. During the experiments five datasets are employed that cover a number of dynamic backgrounds scenarios:

Jug [1] A foreground jug floats through the background rippling water [29].

Railway [2] A strong breeze causes the camera to jitter during the capture [21].

Beach [3] Multiple foreground people walk through a background beach of moving waves.

Lights [4] The lights in the scene are switched on then back off during the capture [2].

[1] Downloaded from http://www.cs.bu.edu/groups/ivc/data.php

[2] Downloaded from http://www.cs.cmu.edu/~yaser/new_background subtraction.htm

[3] Downloaded from http://www.wisdom.weizmann.ac.il/~vision/Behavior Correlation.html

[4] Downloaded from http://ttic.uchicago.edu/~licheng/Bksbt/videos/lights_data.avi

Trees [5] A well-known Wallflower dataset that has a waving tree in background [25].

Fig. 2 presents sample frames of each video dataset, where for each column, the top row shows the first frame in the video, the second row displays a test frame with corresponding hand-labeled ground-truth given in the third row. Fig. 3 provides a visual comparison of recursive MoG, Sheikh et al., online 1-SVM and our algorithm. Overall recursive MoG gives inferior results as it tends to produce noisy output and is slower to adapt to changes. While the pixel-based online 1-SVM of [2] adapts quickly to illumination changes *e.g*. for the *Lights* dataset, it nevertheless yields label noises, and in contrary, our method produces much smoothed segmentation results while still responding quickly to illumination changes, as shown visually in Figure 3. We notice that Sheikh et al. also produce very competitive foreground labels, while in term of preserving the shape or silhouette details, it performs less successful when comparing to our method. These results empirically support that the proposed approach is capable of dealing with the challenging situations such as illumination changes, dynamic water backgrounds, and camera jitters.

A quantitative comparison is evaluated based on the precision recall (PR) analysis. The PR curves are generated by adjusting the threshold parameters of the evaluating algorithms followed by computing the precision and recall of the predicted labels against the ground truth. As in Figure 5, online 1-SVM and Sheikh et al. both perform consistently better then the recursive MoG method, but at the same time are outperformed by the proposed approach. An interesting observation is that Sheikh et al. seems always outperform other comparison methods (except for the proposed one) by a large margin, in area of medium to large precisions (meanwhile recalls are from small to medium), while it seems to be inferior to 1-SVM or sometimes MoG as we emphasis more on recalls. Here our method performs consistently very well in the PR curves of these testbeds. In addition, in Figure 4 we visually compare our result to two approaches on the *Jug* sequence, namely the autoregressive method of [29] and the method of [3] that exploits spatial neighbors, as both report only visual results. Our method is also shown to provide visually competitive results.

Both the quantitative analysis and this visual comparison suggest that our approach perform better or at least comparable to these state-of-the-arts. Moreover, unlike existing approaches, our result is obtained *without* resorting to any preamble pure background images for model training and initialization. We also would like to emphasize that these state-of-the-art methods including Sheikh et al. [21] (11 fps), [29] (0.125 fps), and [3] (no report on fps) are non-realtime methods, while our method works in realtime. During the experiments, we employ an ATI Radeon X1950 GPU (which is a widely available middle class graphics processor) running on an IBM IntelliStation M Pro desktop with Intel 3.4GHz Pentium 4 CPU. Our GPU implementation executes at 81.5 FPS for video feed with 320×240 resolution, which gives on average 40-fold speed-up over our CPU implementation on the same computer.

[5] Downloaded from http://research.microsoft.com/~jckrumm/wallflower/testimages.htm

5. Conclusion and Discussion

We presented a competitive algorithm for realtime foreground segmentation from videos which uses the ideas from large margin classifiers and online learning to extend the 1-SVM formalism to accommodate spatial dependency among neighboring pixels, and the algorithm is adapted and implemented to efficiently work with highly parallel graphics processors. Experimental evaluation on a number of datasets shows that our realtime algorithm is comparable to the state-of-the-art offline algorithms. As future work, we plan on devising dedicated global inference procedure, as well as to extend our algorithm to other interesting but difficult scenarios such as realtime foreground segmentation under water, in the presence of fog or during the night.

Acknowledgement

The authors thank Mr. G. Dalley, Dr. J. Krumm, Dr. Y. Sheikh, Dr. S. Sclaroff, Dr. Eli Shechtman and Dr. Z. Zivkovic for sharing their datasets and codes.

References

[1] N. Cesa-bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50:2050–2057, 2004.

[2] L. Cheng, S. V. N. Vishwanathan, D. Schuurmans, S. Wang, and T. Caelli. Implicit online learning with kernels. In *Neural Information Processing Systems*. MIT Press, 2006.

[3] G. Dalle, J. Migdal, and W. Grimson. Background subtraction for temporally irregular dynamic textures. In *WACV*, 2008.

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

[5] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Proc. Thirteenth Conf. on Uncertainty in Artificial Intelligence*, 1997.

[6] M. Gong and Y.-H. Yang. Real-time stereo matching using orthogonal reliability-based dynamic programming. *IEEE TIP*, 16(3):879–884, 2007.

[7] A. Griesser, S. D. Roeck, A. Neubeck, and L. V. Gool. Gpu-based foreground-background segmentation using an extended colinearity criterion. In *Vision, Modeling, and Visualization*, 2005.

[8] Z. Kim. Real time object tracking based on dynamic feature grouping with background subtraction. In *CVPR*, 2008.

[9] J. Kivinen, A. Smola, and R. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8), Aug 2004.

[10] J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–64, January 1997.

Figure 2. From left to right: sample images of the four datasets (*Jug*, *Railway*, *Beach*, *Lights*, and *Trees*), each presented in one column. Top row: the first video frame; Middle row: a test frame; Bottom row: hand-labeled ground truths.

Figure 3. First row: results of the MoG method [12]. Second row: Sheikh et al. [21]. Third row: results of results of the online 1-SVM method [2]. Fourth row: result of our approach. Bottom row: the extracted foregrounds using our approach. While effectively removes noises, Our approach preserves the detailed shapes of foreground objects, *e.g*. the pedestrian in the *Railway* dataset.

[11] V. Kolmogorov and C. Rother. Minimizing nonsubmodular functions with graph cuts-a review. *IEEE T. PAMI*, 29:1274–1279, 2007.

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

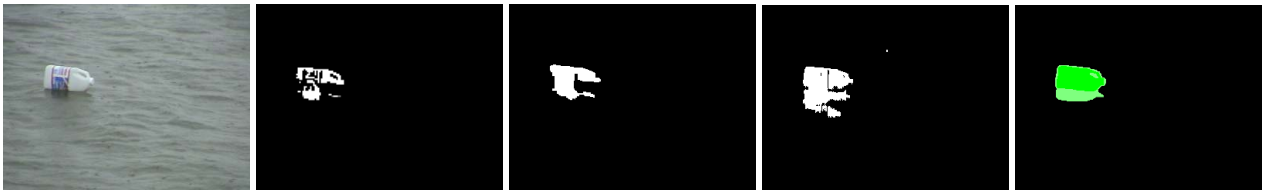

Figure 4. Comparisons to other recent offline methods on the *Jug* dataset. From left to right: test frame, result from Figure 5 of [29], result from Figure 4 of [3], our result, and ground truth from Figure 4 of [3].

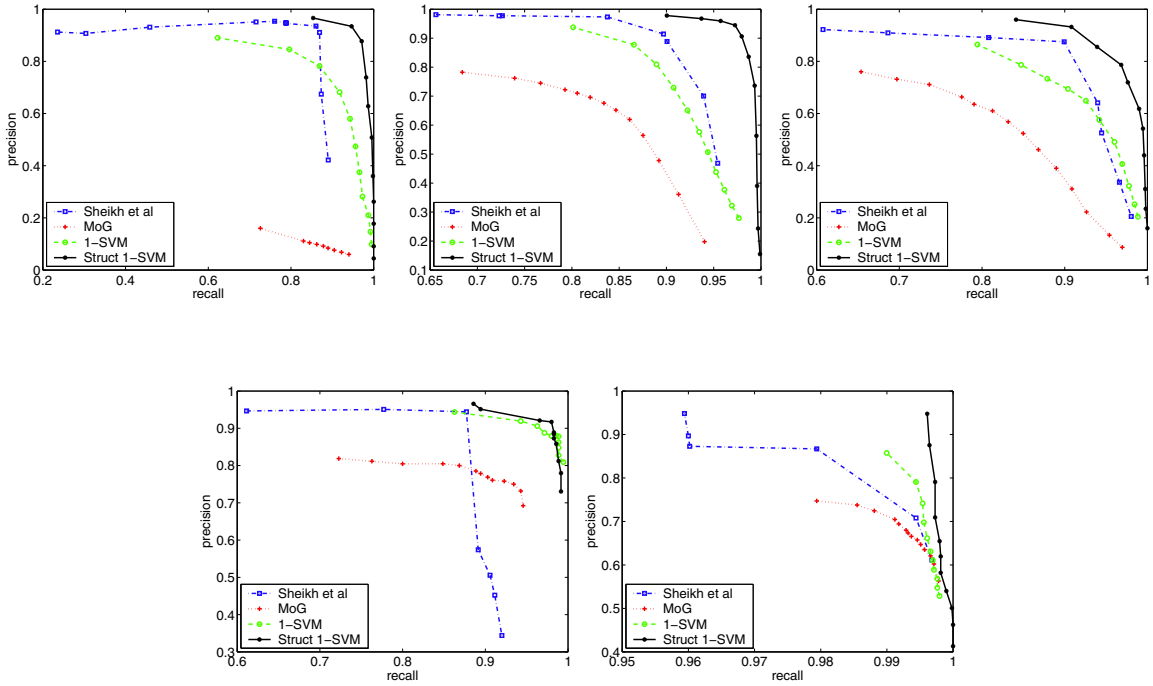

Figure 5. Precision-Recall curves of four algorithms (MoG as baseline, online 1-SVM, Sheikh et al.[21], and the proposed algorithm on the *Jug*, *Railway*, *Beach*, *Lights*, and *Trees* datasets.

[13] P. Long, R. Servedio, and H. Simon. Discriminative learning can succeed where generative learning fails. *Inf. Process. Lett.*, 103(4):131–135, 2007.

[14] V. Mahadevan and N. Vasconcelos. Background subtraction in highly dynamic scenes. In *CVPR*, 2008.

[15] J. Migdal and W. Grimson. Background subtraction using markov thresholds. In *WMVC*, pages 58–65, 2005.

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

[17] J. Pilet, C. Strecha, and P. Fua. Making background subtraction robust to sudden illumination changes. In *ECCV*, pages 567–580, 2008.

[18] 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.

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

[20] Y. Sheikh and M. Shah. Bayesian modeling of dynamic scenes for object detection. *IEEE T. PAMI*, 27(11):1778–1792, 2005.

[21] Y. Sheikh and M. Shah. Bayesian object detection in dynamic scenes. In *CVPR*, 2005.

[22] A. Smola, S. V. N. Vishwanathan, and Q. Le. Bundle methods for machine learning. In *Advances in Neural Information Processing Systems 20*, Cambridge MA, 2007.

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

[24] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Neural Information Processing Systems*, pages 25–32, Cambridge, MA, 2004. MIT Press.

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

[26] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6:1453–1484, 2005.

[27] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

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

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

[30] Z. Zivkovic and F. Heijden. Recursive unsupervised learning of finite mixture models. *IEEE T. PAMI*, 26(5), 2004.