Adaptive Appearance Modeling for Video Tracking: Survey and Evaluation

Samuele Salti, Andrea Cavallaro, Luigi Di Stefano

***Abstract*—Long–term video tracking is of great importance for many applications in real-world scenarios. A key component for achieving long–term tracking is the tracker's capability of updating its internal representation of targets (the appearance model) to changing conditions. Given the rapid but fragmented development of this research area, we propose a unified conceptual framework for appearance model adaptation that enables a principled comparison of different approaches. Moreover, we introduce a novel evaluation methodology that enables simultaneous analysis of tracking accuracy and tracking success, without the need of setting application-dependent thresholds. Based on the proposed framework and this novel evaluation methodology, we conduct an extensive experimental comparison of trackers that perform appearance model adaptation. Theoretical and experimental analyses allow us to identify the most effective approaches as well as to highlight design choices that favor resilience to errors during the update process. We conclude the paper with a list of key open research challenges that have been singled out by means of our experimental comparison.**

***Index Terms*—Adaptive Appearance Models, Video tracking, Evaluation Methodology.**

I. Introduction

TRACKING a previously unseen target in video streams is an important building block for the automation of many image-based tasks, such as behavior analysis, annotation of image sequences, natural human-computer interfaces and autonomous navigation. Long–term tracking in real-world conditions is made difficult by several factors, including illumination and pose changes, occlusions, deformable targets, distractors and clutter. Video trackers rely on an internal representation of target appearance, the *appearance model*, which is compared to measurements extracted from incoming frames at candidate target positions to estimate the most likely target location. To create the appearance model and the measurements, trackers project image regions at candidate target positions onto lower dimensionality feature spaces that highlight relevant information for the tracking task. [1].

Manuscript received July 27, 2011; revised April 30, 2012; accepted May 17, 2012. This work was done when the first author was visiting Queen Mary, University of London. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mary Comer.

S. Salti is with the Department of Electronics, Computer Science, and Systems, University of Bologna, Bologna 40135, Italy (e-mail: samuele.salti@unibo.it).

A. Cavallaro is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, U.K. (e-mail: andrea.cavallaro@elec.qmul.ac.uk).

L. Di Stefano is with the Department of Electronics, Computer Science, and Systems, University of Bologna, Bologna 40135, Italy (e-mail: luigi.distefano@unibo.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2012.2206035

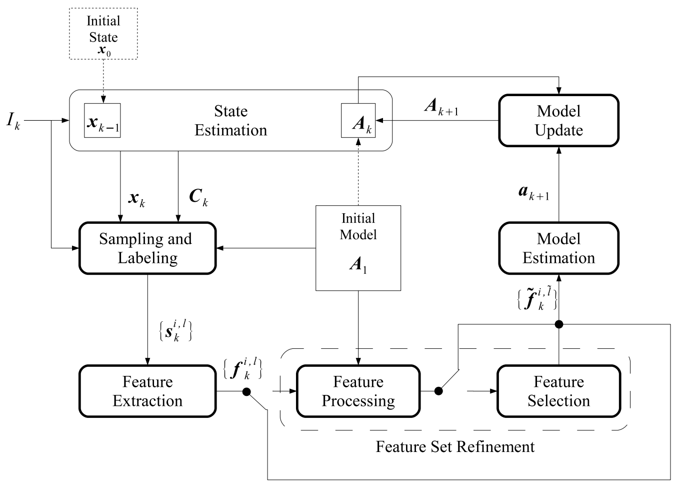


Fig. 1. Proposed block diagram of the unified framework for model adaptation in video tracking, which consists of the building blocks highlighted by a thicker box line. The state estimation block represents a generic tracker that receives as input the incoming frames and uses an internal representation to model targets in order to estimate their state.

Early trackers kept the appearance model fixed throughout a sequence [2]–[7], while focusing on robust localization and matching strategies. Recently, several methods have been proposed to track targets while evolving the appearance model in order to adapt it to changing imaging conditions [8]–[16]. Appearance model adaptation introduces several challenges, such as the need for simultaneous fulfillment of the contradicting goals of rapid learning and stable memory. This is referred to as the stability/plasticity dilemma [17]. Adaptive trackers also incur the risk of including wrong measurements while updating the target model, thus leading to *drifts* from the target. To avoid drifting, model adaptation has to be robust to outliers generated by the appearance of the background or other objects that may partially occlude the target. Finally, an adaptive tracker must be able to evaluate on–line (without ground truth) the quality of the estimated target location, so as to adjust accordingly its contribution to model update. These challenges, together with the different solutions adopted to address them, make the evaluation and the choice of an adaptive tracker a difficult task.

In this paper, we propose a unified conceptual framework that identifies the common building blocks of trackers that perform adaptive appearance modeling (Fig. 1). The framework is general and can accommodate in its model also video trackers designed to work with a fixed appearance model.. We also propose a novel quantitative performance evaluation methodology, independent from thresholding, which simultaneously considers the accuracy of the tracking estimation and the ability of the tracker to maintain a long–term successful operation. According to the proposed framework and based on the proposed evaluation methodology, we perform a comprehensive review of existing methods and conduct an in-depth experimental comparison using publicly available test sequences. We critically discuss the performance of different algorithms in various real-world scenarios and highlight their strengths and weaknesses. Finally, based on this analysis, we list the main open research challenges in adaptive target modeling for video tracking.

The paper is organized as follows: Section II introduces notations and definitions that will be used throughout the paper; in Section III, we propose the general framework for target model update; in Section IV, we present and discuss the proposed evaluation methodology; next, Section V describes the selected trackers and test sequences and Section VI presents the experimental comparison. Finally, Section VII summarizes the main contributions and highlights open research challenges.

II. Notations and Definitions

Let $\mathbf{I}_k = \{I_m\}_{m=1}^k$ be an image sequence. A video tracker estimates a sequence of hidden variables $\mathbf{X}_k = \{\mathbf{x}_m \in \mathcal{X} \subset \mathbb{R}^d\}_{m=1}^k$, where $\mathcal{X}$ is the state space and $\mathbf{x}_m$ is the estimated target state at time m. $\mathbf{X}_k$ is the trajectory of the target up to time k. The target state includes position and, optionally, other geometric or kinematic characteristics such as scale, aspect ratio, orientation with respect to image axes and velocity. The target state can be represented as:

$$\mathbf{x}_k = [\, x\, y\,]\,, \tag{1}$$

with $x \in \{0, \ldots, W-1\}$, $y \in \{0, \ldots, H-1\}$, W and H denoting the image width and height, respectively. This choice results in a fixed–scale, fixed–aspect–ratio and fixed–orientation tracker [11], [13]–[16], [18]–[20]. Other trackers allow for isotropic scale variation [4], [5], [21]:

$$\mathbf{x}_k = [\, x\, y\, \sigma\,]\,, \tag{2}$$

where $\sigma \in \mathbb{R}_0^+$ is the ratio between the initial and the current width and height of the target. A–BHMC [22] accounts for a non–fixed aspect ratio:

$$\mathbf{x}_k = [\, x\, y\, \sigma_x\, \sigma_y\,]\,, \tag{3}$$

with $\sigma_x, \sigma_y \in \mathbb{R}_0^+$ denoting the ratio between the initial and the current width and height, respectively. IVT [23] and AMFT [24], instead, include orientation θ as state component:

$$\mathbf{x}_k = [\, x\, y\, \sigma\, \theta\,]\,, \tag{4}$$

with $\sigma \in \mathbb{R}_0^+, \theta \in [0 \ldots 2\pi]$. Finally, ClosedLoopMatting [25] combines tracking with automatic matting of the target, thereby extending the notion of state estimation to the ability of precisely estimating the target silhouette in each frame.

Trackers consider several state candidates $\hat{\mathbf{x}}_k^i$ in order to estimate the current state $\mathbf{x}_k$. Candidates are drawn according to a variety of *sampling strategies*, typically in a neighborhood of the previous state so as to enforce temporal smoothness [1], [26]. The estimation of $\mathbf{x}_k$ is based on assigning a score C_k^i to each candidate. Examples of scores are the weight of a particle in a particle filter [5], the similarity in the feature space in a mean-shift tracker [4] or the confidence of a classifier in a tracking–by–detection approach [11]. We shall refer to the set of pairs $\langle \hat{\mathbf{x}}_k^i, C_k^i \rangle$ as the confidence map $\mathbf{C}_k$.

Let $\mathbf{A}_k$ denote the instance of the overall appearance model used at time k and $\mathbf{a}_{k+1}$ the part of the model estimated from frame I_k. In order to estimate $\mathbf{x}_k$ and $\mathbf{C}_k$, trackers compare $\mathbf{A}_k$ with measurements from I_k. Non-adaptive trackers learn (or acquire) $\mathbf{A}_k$ once either off-line or on-line using the first frame(s) of a sequence and then keep the model unchanged throughout a sequence. Adaptive trackers update the model $\mathbf{A}_k$ over time by estimating $\mathbf{a}_{k+1}$ from $\mathbf{x}_k$ and $\mathbf{C}_k$ and then merging it with $\mathbf{A}_k$, yielding $\mathbf{A}_{k+1}$. According to the type of the adopted appearance model, trackers can be grouped into three classes, namely generative, discriminative and hybrid trackers.

Generative trackers drive the localization procedure using a maximum–likelihood or maximum–a–posterior formulation relying only on the target appearance model [8], [9], [21], [23]–[25], [27], [28]. Generative appearance models represent object appearance without considering its discriminative power with respect to the appearance of the background or other targets. *Discriminative trackers* localize the target using a classifier that learns a decision boundary between the appearance of the target and that of the background or other targets [10]–[16], [18], [29], [30]. Finally, *hybrid trackers* use a combination of the previous two approaches. Examples include switching between discriminative and generative observation models according to targets proximity in a multi–target scenario [31]; using co–training [32] between a long–term generative observation model and a short–term discriminative one [33]; using several generative models but discriminatively learning in each frame the weights to combine them in order to maximize the distance to the background appearance [34]; storing and updating two generative non–parametric models of target and background appearances and using them to train a discriminative tracker in each frame [20]; or using a generative tracker to bootstrap a discriminative tracker [15]. Adaptive trackers will be discussed in details in the next section.

III. A Unified Framework for Adaptive Target Modeling

In this section, we identify and compare the common steps (Fig. 1) performed by adaptive trackers in order to carry out appearance model update.

A. Overview

Given $\mathbf{x}_k$ and $\mathbf{C}_k$, a set of *samples* $\{\mathbf{s}_k^i\}$ of the new target appearance is extracted from the current frame. Discriminative trackers extract a set of samples also from the background. These samples are hard-*labeled* ($l \in \{-1, 1\}$) or soft-*labeled* ($l \in [-1, 1]$) as target or background, thus yielding a labeled sample set $\{\mathbf{s}_k^{i,l}\}$.

TABLE I
REVIEWED METHODS. THE TRACKERS CONSIDERED IN THE EXPERIMENTAL COMPARISON ARE IN BOLD. LEGEND: SL: SAMPLING AND LABELING; FSR: FEATURE SET REFINEMENT; ME: MODEL ESTIMATION; MU: MODEL UPDATE; CS: CURRENT STATE; CSC: CURRENT-STATE-CENTERED; FL: FIXED LABELER; AL: ADAPTIVE LABELER; FAL: FIXED AND ADAPTIVE LABELER; CT: CO-TRAINING. BP: BLENDED PIVOT; PA: PIVOT ADDED; PT: PIVOT TRACKING; LS: LABEL SWITCH; ROF: REDUNDANT AND OUTLIERS FILTERING. DUF: DIRECT USE OF FEATURES; NCT: NEW CLASSIFIER TRAINING; CU: CLASSIFIER UPDATE. LM: LAST MODEL; SW: SLIDING WINDOW; R: RANKING; B: BLENDING; C: CLUSTERING; S: SUBSPACE; M: MANIFOLD.

		SL	FSR	ME	MU
Template Update	[9]	CS	PT	DUF	LM
IVT	[23]	CS	-	DUF	S
AdaptiveManifold	[27]	CS	-	DUF	M
$\mathcal{WSL}$	[8]	CS	-	DUF	B
OnlineAppearanceModel	[28]	CS	-	DUF	B
AMFT	[24]	CS	BP	DUF	B
VTD	[21]	CS	PA	DUF	SW
Ensemble Tracking	[11]	CS	LS	NCT	SW
Non-Parametric Tracker	[20]	AL	ROF	DUF	R
A-BHMC	[22]	CS	-	DUF	LM
Co-Tracking - SVM 1	[12]	CT	-	CU	SW
Co-Tracking - SVM 2		CT	-	CU	SW
CT - Generative	[33]	CT	-	DUF	M
CT - Discriminative		CT	-	CU	SW
Adaptive Weights	[34]	CS	BP	DUF	B
Discriminative Features Sel.	[10]	CS	BP	DUF	LM
GMM Clustering	[35]	CS	-	DUF	C
Boost	[13]	CS	-	NCT	R
SemiBoost	[18]	FL	-	NCT	R
BeyondSemiBoost	[29]	FAL	-	NCT	R
MILBoost	[14]	CSC	-	NCT	R
PROST	[36]	FAL	-	CU	R
DataDriven	[30]	AL	-	DUF	S
TLD	[15]	AL	-	CU	R
CLM - Short Term	[25]	CS	-	DUF	LM
CLM - Colors		CS	-	DUF	B
CLM - Long Term		CS	-	DUF	C
STRUCK	[16]	CS	-	CU	R

Next, samples are projected onto the feature space $\mathbf{F}$, generating a set of labeled features $\{\mathbf{f}_k^{i,l}\}$. Features can then be filtered and/or selected in order to provide a more representative feature set $\{\tilde{\mathbf{f}}_k^{i,\tilde{l}}\}$, $\tilde{l} \in [-1, 1]$. In case of *filtering*, the set of features may be pruned to remove outliers or augmented with reliable features. During this phase, labels may be switched or modified. In case of *selection*, the most effective features for tracking the target in the current frame are identified out of the set of available features.

Given the refined labeled features, the appearance model $\mathbf{a}_{k+1}$ of the target in the current frame is first estimated and then merged with the previous model $\mathbf{A}_k$, so as to attain the model $\mathbf{A}_{k+1}$ to be used in the next frame for state estimation.

Table I summarizes adaptive trackers within the proposed framework.

B. Sampling and Labeling

Given $\mathbf{x}_k$ and $\mathbf{C}_k$, the *sampling and labeling* step selects regions from the current frame to be used to update the appearance model and, in a discriminative tracker, classifies them as belonging to the target or to the background. To distinguish between the labeling and the tracking classifier we will refer to the former as *labeler*.

Current state sampling uses only the region defined by $\mathbf{x}_k$ to update the appearance model [8], [9], [21], [23]. Discriminative trackers use as background appearance samples from a region surrounding $\mathbf{x}_k$ [10], [11], [13], [16], [25]. This approach assumes the tracker output to be correct and only subsequent stages may mitigate the effects of a wrongly estimated $\mathbf{x}_k$. For example, STRUCK [16] couples this simple strategy with an elaborate model update stage, which is based on the use of Structured-Output Support Vector Machines (SVMs) to mitigate the effects of wrong labeling [42].

Current–state–centered sampling extracts samples from the region defined by a neighborhood of $\mathbf{x}_k$ [14]. Samples extracted in proximity of $\mathbf{x}_k$ are grouped in bags of samples (and at least one sample from each bag is assumed to be a target sample), whereas samples from the outer sampling region are used as background samples. Assuming the state to be at most just slightly off-target, subsequent stages disambiguate the uncertainty left in the target samples, for instance by using Multiple Instance Learning [14].

Co–training sampling combines two sub-trackers deploying independent features [12]. Each sub-tracker aims at discriminating the cases that are ambiguous for the other sub-tracker, implicitly assuming that, in a given frame, one of the two features can correctly track the target. The state $\mathbf{x}_k$ is given by combination of the outputs of the two sub-trackers, whereas the sampling and labeling for appearance model update of each tracker is carried out independently within the co–training framework [32]. Each sub-tracker provides training samples to the other. Target samples come from the global maximum of the confidence maps, while background samples are taken from local maxima.

When an *external labeler* is used, samples are extracted in the region defined by $\mathbf{x}_k$ and its neighborhood but not labeled according to their position with respect to $\mathbf{x}_k$. Samples are soft–labeled as target or background according to the confidence of the labeler [18], [20], [29]. The use of a labeler to guide model updates aims at breaking the *self–learning loop*, when the tracker output is used to update the appearance model that, in turn, is used to produce the next tracker output. The labeler can be fixed, adaptive or a combination of both. A *fixed labeler*, such as an object detector or a similarity function with a fixed pivotal appearance model, is trained off–line or in the first frame(s) and no longer updated [18], thus limiting the degree of adaptability of the tracker. No assumptions are made about the correctness of the current state, besides the proximity to the target. An *adaptive labeler* is either a similarity function with respect to the previous model [20], [30] or an object detector trained on-line using the tracker output [15]. This approach does not assume reliability of the current state but requires that no sudden changes happen in the evolution of the target or background appearance. This assumption is relaxed in the LOOP strategy [15]: the appearance is allowed to suddenly vary if it becomes similar again to the previous appearance in a limited number of frames; all the appearances between these two events are considered as positive samples of the target appearance. The degree of adaptability in such methods (the maximum variation in appearance between consecutive frames and the number of frames in the LOOP strategy) is dictated by hard thresholds. As it uses the previous model or a detector trained with the previous target appearance to label

TABLE II
SUMMARY OF THE FEATURES USED BY ADAPTIVE TRACKERS. FEATURES ARE LISTED UNDER TWO GROUPS BASED ON WHETHER A TRACKER USES A PART-WISE OR A TARGET-WISE OBJECT REPRESENTATION. '*' INDICATES THE USE OF MULTIPLE TRACKERS OR MULTIPLE WORKING CONDITIONS, HENCE NOT ALL THE LISTED FEATURES ARE USED SIMULTANEOUSLY. LEGEND: SF: STEERABLE FILTERS [37]; LBP: LOCAL BINARY PATTERNS [38]; CFB: COLOUR FILTER BANK [10]; BOF: BAG OF FEATURES; HOG: HISTOGRAM OF ORIENTED GRADIENTS [39]; ISM: IMPLICIT SHAPE MODEL [40].

	Part-wise						Target-wise											
	Pixel	Patch	Histogram		Haar Filters		Template			Histogram					Contour	Detector		
	RGB	Gray	Colour	HOG	SF	Haar-like	Intensity	HSI	Edges	Colour	HOG	LBP	BoF	CFB	Ellipse	Head	HOG	ISM
Single	[35]	[15]*			[8]	[15]*, [36]	[9], [23], [30], [28], [27]											
Mixture	[11], [20] [31]*, [25]*	[25]*		[11], [20], [31]*, [25]*		[16]	[33]*, [16]			[41]*, [12]*, [24], [31]*, [34], [16]	[12]*, [33]*, [24]	[41]*	[34]		[34]	[34]	[41]*	[41]*
Selection			[29]	[29]		[13], [18], [29], [14]		[21]*	[21]*					[10]				

current samples, this approach is prone to drift due to self–learning, although the loop depends on models rather than on states. Finally, a *combination* of fixed and adaptive labelers can be used [29], [36]. The fixed labeler is trained on the first frame and never updated, whereas the adaptive labeler is used to label samples for model update. This method aims at reducing the risk of drifting due to self-learning by controlling when the labeler is updated. The labeler is updated when the tracker and the fixed labeler are in agreement [29] or when the adaptive labeler is in agreement with one of them [36]. However, this approach has comparable limitations on the degree of adaptability with respect to those deploying a fixed labeler, as we shall see in the experimental evaluation.

C. Feature Extraction and Refinement

The features extracted for each sample $s_k^{i,l}$ of I_k produce a set of labeled feature vectors $\{f_k^{i,l}\}$. These features can be categorized as part–wise or target–wise and are summarized in Table II.

Part–wise features are extracted from small patches or even individual pixels. Part–wise features make it possible to reason explicitly about occlusions and, potentially, to avoid using features from occluding objects to update the appearance model. They also help dealing with the approximation inherent in modeling targets as rectangular objects, by allowing features inside the target bounding box to be classified either as foreground or background. Popular part–wise features are Haar wavelets, mainly due to their invariance to affine brightness changes and to their low computational cost, as they can be computed efficiently in constant time with respect to the wavelet size by integral images [43]. Another popular representation uses RGB values and a coarse Histogram of Oriented Gradients (HOG) of a small image patch [11]. This choice trades descriptiveness for robustness, since a small patch makes it easier to handle partial occlusions and target deformation. In contrast, *target–wise features* represent the whole target appearance, such as by means of a target template [9], [23] or color [4], [34] and gradient histograms [21], or their combination [12], [33]. Histograms tolerate target rotations and scaling without explicit appearance model adaptation, whereas templates need to be coupled with an appearance model update algorithm [23].

Given the features $\{f_k^{i,l}\}$ extracted and labeled from the current frame, the *feature set refinement* step processes the features and the labels in order to obtain a modified set $\{\tilde{f}_k^{i,\tilde{l}}\}$ that is more effective for appearance model update. Two main strategies can be followed, which can be deployed alternatively or sequentially: feature processing and feature selection. In *feature processing*, a tracker can perform sample checking or use a pivot. *Sample checking* is based on the assumption that it is possible to define *a priori* samples that are not suitable to perform model update given the current appearance model. Adaptive trackers can perform *redundant sample removal*, when feature vectors are too similar to the current model [20], *outliers filtering*, when feature vectors are too different from the current model [20] or *label switching*, when the confidence on a target–labeled feature vector is too low and the label is switched to background[1] [11]. The initial appearance may act as a *pivot*, under the assumptions that the bounding box in the first frame is correct and that the target and background appearance remains similar to initial ones. Features from the pivot refine the current sample set when they are *added* to [21] or *blended* with the current feature set [10], [34]. Features from the pivot can be used to refine the feature set also by *tracking* them in the current frame, starting from the current state [9].

On–line feature selection is performed when a fixed set of features is used, whose composition is updated in each frame according to their effectiveness in the previous frame(s), under the assumption that the position estimated by the tracker in the previous frame(s) is correct. This can be achieved by evaluating the log-likelihood ratio of color values [10], [25] or can be performed by applying on–line boosting [44] to weak classifiers that act as feature selectors. The latter kind of feature selection is a key component of a family of discriminative tracking algorithms [13], [14], [18], [29] that performs model update by selecting features according to their discriminative ability to distinguish the target from the background. In addition to the inherent risk of self–learning, one of the main difficulties in performing on–line selection is due to the different score dynamics and ranges of the various cues that makes it hard to compare their scores [45]. In fact, all the cited works except BeyondSemiBoost [29] use only one kind of features (Haar wavelets or color values).

D. Model Estimation and Update

Given the filtered feature set and labels, a new partial model $\mathbf{a}_{k+1}$ that describes the target appearance in the current frame is built. The model estimated for the current frame can be a *non–parametric* ensemble of features extracted from the target or the background. For example, one can use as appearance

[1]This is done mainly to counteract the approximation inherent in the use of a rectangular box as target shape.

model the array of responses from a bank of steerable filters [8] or the HSI (Hue-Saturation-Intensity) and edge templates [21]. The model can be a *new classifier*, in particular a new weak classifier to be used in the boosting framework, trained to separate the target from the background in the current frame [11], [13]. Finally, the model $\mathbf{a}_{k+1}$ can be represented by positive and negative samples extracted from the current frame in order to update a *previously trained classifier* used for tracking [16], [33].

Once the model for the current frame, $\mathbf{a}_{k+1}$, is estimated, it is merged with the previous overall model, $\mathbf{A}_k$, to obtain the new overall model $\mathbf{A}_{k+1}$. This step addresses the stability/plasticity dilemma.

Five main strategies can be identified for this stage, namely last model only, sliding window, ranking, blending or subspace/manifold based strategy.

The model estimated in the *last frame* can be used as the model for the next frame. This strategy is prone to drift the most, and it has been used for template update in Lucas–Kanade trackers [9] as well as in one of the first proposals for discriminative tracking [10]. Using a *sliding window*, a fixed amount of samples/classifiers is kept after a frame is processed. The newest is added and the oldest is discarded. The sliding window strategy is used in combination with retraining of a classifier [12], [33] or by inserting and discarding weak classifiers in the strong classifier of the boosting framework [11]. Recently, it has been used also in the VTD generative tracker, where it provides in each frame the initial set of features to perform feature selection by sparse PCA [21]. *Ranking* keeps the most effective samples/classifiers, up to a fixed amount, after each frame is processed, with the new one always added. This raises the problem of assessing the effectiveness of single models on-line without ground-truth. This problem is usually solved by evaluating them on the last frame(s) and assuming correct tracking. Boosting of an ensemble of classifiers provides a natural framework to implement this strategy while simultaneously performing feature selection [13], [14], [18], [29]. STRUCK [16] ranks support vectors to perform on-line learning with a budget, *i.e.* a maximum number of allowed support vectors.

With *blending*, samples extracted in the current frame are blended with previous ones. Although, in principle, this approach is more stable than the previous strategies, because all the history up to the current frame influences the new model, it is more prone to drift as the inclusion of wrong samples in the appearance model cannot be mitigated afterwards. Only inclusion of correct samples will eventually render the influence of outliers negligible. Such strategy has been deployed only for generative trackers, as shown in Tab. I. A principled, though approximate, way to implement this strategy was proposed under the assumption that appearance is a Gaussian random variable [24]. In such case, Kalman filtering is used to estimate the appearance model in each frame, with weights for blending given by the filter adaptive gain. A *subspace* [23] or a set of subspaces (an approximation for a manifold in the feature space) [33] can also be updated with the new model for the current frame. A variant of the subspace idea is to keep only the subspace centroids in the feature space, *i.e.* to perform *on-line clustering* of the models and retain only the cluster centers [25] or a GMM approximation of the manifold [35]. On the one hand, all variants of the subspace strategy potentially retain all target appearances with a fixed amount of memory and are the most stable solution. On the other hand, with such a model, it is difficult to accommodate for sudden target appearance changes. A forgetting factor may be used to diminish over time the effect of the oldest samples on the shape of the subspace/manifold [23], or a threshold on the number of time steps a cluster is not updated may be used to discard those expected to be no longer representative of target appearance [35].

IV. Evaluation Methodology

In this section, we introduce a novel compact evaluation methodology for video trackers. We evaluate the performance of trackers by simultaneously considering the quality of the output and the ability to track for a long time interval. We define this methodology in order to achieve independence of application-dependent thresholds, since different choices meet different performance requirements.

We measure the quality of the output as overlap between the estimated state and the ground truth, by using Dice, d_k, that for each frame k is defined as [46]

$$d_k = \frac{2|B_{\mathbf{x}_k} \cap B_{\mathbf{x}_k^{GT}}|}{|B_{\mathbf{x}_k}| + |B_{\mathbf{x}_k^{GT}}|}, \tag{5}$$

where $B_{\mathbf{x}_k}$ ($B_{\mathbf{x}_k^{GT}}$) represents the set of pixels included in the bounding box defined by the estimated state $\mathbf{x}_k$ (the ground-truth $\mathbf{x}_k^{GT}$); $|\cdot|$ represents the cardinality of a set; $d_k \in [0, 1]$, with $d_k = 1$ in case of perfect overlap. Note that d_k is sensitive to small misalignment of the bounding boxes and therefore $d_k \geq 0.7$ usually corresponds to a satisfactory tracking result. Using d_k, we define the *Correct Track Ratio* [1], namely the percentage of frames in a sequence where the tracker is *on-target*. Usually this condition is determined by considering the frames for which d_k is larger than a threshold. However, in our methodology we do not fix such threshold and we analyze instead the curve reporting the Correct Track Ratio versus the corresponding Mean Dice (Fig. 2), obtained by varying the threshold. This curve allows one to choose a tracker according to the application requirements. Frames where the target is not visible are not assigned a score, whereas when a tracker does not yield a bounding box in a frame where the target is visible, we assign it a zero Dice overlap and consider the frame in the plots.

For instance, let us consider trackers represented with the yellow and blue lines in Fig. 2. Although both trackers lose the target for some frames (*i.e.* their lines do not intersect the vertical line $CTR = 1$), the tracker represented with the blue line is able to track the target longer. If large overlaps between the estimated state and the ground truth are important for the application, even though for less frames, then the tracker represented with the yellow line can provide higher overlaps than the blue one in 15% of the sequence (the yellow line is higher than the blue line in the first part of the chart).

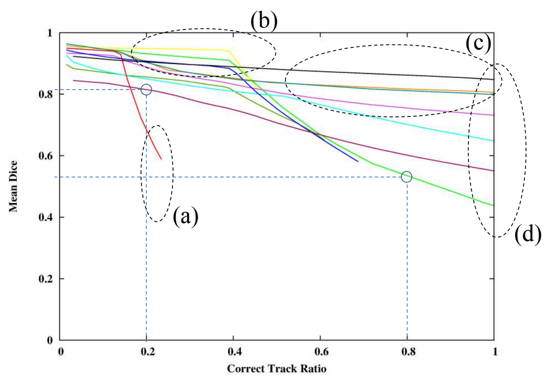


Fig. 2. Illustrative example of the usefulness of the proposed evaluation methodology based on Mean Dice (MD) vs. Correct Track Ratio (CTR). An ideal tracker is represented by the point (1,1), *i.e.* full overlap throughout the whole sequence. In general, however, MD decreases as CTR increases. For this reason the proposed evaluation methodology allows one to visualize the operational condition of trackers in order to choose the most appropriate to the constraints of a specific application. For example, considering the purple tracker, the point $CTR = 0.2, MD = 0.81$ shows that a Mean Dice as high as 81% can be delivered by the tracker only for 20% of the sequence. Conversely, with the green tracker, the point $CTR = 0.8, MD = 0.55$ shows that if one wishes to track the target for 80% of the sequence a Mean Dice as low as 55% has to be accepted. Further observations: (a) the red and the blue trackers fail to follow the target for the whole sequence; (b) if high Dices are required, the yellow tracker can be used, but it is able to provide this level of performance only on 40% of the sequence; (c) the black tracker is the best choice if it is important to track the target for more than 50% of the sequence; (d) all trackers that intersect the line CTR = 1 track the target for the whole sequence.

V. Evaluation setup

In this section, we list and briefly discuss the trackers under evaluation and their settings. Then, we present the dataset and categorize the considered sequences according to the main challenges for trackers.

A. Trackers under analysis

To perform the comparative evaluation, we consider eight adaptive and three non–adaptive trackers. The use of non–adaptive trackers is important to evaluate the impact of model adaptation as well as to assess the relative performance of adaptive solutions. The adaptive trackers are: Boost Tracker [13], SemiBoost Tracker [18], BeyondSemiBoost Tracker [29], A–BHMC (Adaptive Basin Hopping Monte Carlo) [22], IVT (Incremental Visual Tracker) [23], MILBoost Tracker [14], TLD (Track Learn Detect) [15] and STRUCK [16]. The non–adaptive trackers are FragTrack [19], a color–based particle filter (PF) [5] and Mean–Shift [4].

1) Non–adaptive trackers: FragTrack uses spatially localized histograms defined over two grids superimposed on the target and deploys integral histograms to efficiently locate the best matching position. Both Mean–Shift (MS) and Particle Filter (PF) use a target-wise feature, *i.e.* the color histogram. MS efficiently locates the most similar area in the surrounding of the previous target position by using the MS gradient ascent algorithm. The color-based PF, similar to MS, uses the Bhattacharyya distance as observation likelihood but deploys particle filtering for state tracking to ensure robustness to distractors.

2) Adaptive trackers: Boost is a simple discriminative tracker that uses the current state for sampling and labeling, Haar wavelet of random size and orientation as features, and performs feature selection via on–line boosting to update the target representation. Model update is based on ranking and the score is the weight of the classifier in the boosted pool. SemiBoost and BeyondSemiBoost aim at breaking the self–learning loop in Boost by adding priors. SemiBoost uses a fixed labeler, whereas BeyondSemiBoost uses an adaptive but slowly evolving labeler in combination with a fixed labeler trained on the first frame. MILBoost improves the drift-prone sampling and labeling strategy in Boost by using the current–state–centered choice. MILBoost relies on the MIL framework to handle the resulting label noise. Instead of sampling several positive candidates according to the current–state–centered choice, STRUCK uses the simple current state for labeling but then feeds the possibly wrong labeled samples to a Structured–Output SVM. TLD uses both a generative and a discriminative tracker. The generative tracker (Lucas-Kanade) self-learns the appearance model at each frame, *i.e.* it is highly adaptive. The output is also used to train on-line an object detector (Random Forest). The detector is then used in parallel to the tracker as a sliding window detector on the whole image. The most confident estimation produced by the tracker or the detector is used as output. Confident detections not in agreement with the output are used to provide negative samples to the detector and improve its performance. If both confidences are low, the object is considered occluded or out-of-sight. The above–mentioned trackers do not deploy any stochastic filters to enforce the temporal consistency, *i.e.* they implicitly assume a constant-position motion model and then select as the new state the one with the maximum confidence in a neighborhood of the previous state or in the whole frame.

A–BHMC and IVT rely on particle filtering for state tracking to ensure temporal consistency. A–BHMC is designed to cope with appearance changes generated by geometric changes: its appearance model is made out of patches that are allowed to move independently but not to vary much in appearance, since patches are matched across frames based on the assumption of brightness constancy. A Generalized Hough Transform voting scheme allows for estimating the likelihood of a bounding box out of the freely evolving patches. Unlike A–BHMC, IVT uses global features for the appearance model (*i.e.* the target template). A subspace of admissible target templates is built on–line via an efficient algorithm for SVD update. The distance from this subspace allows for the definition of the adaptive observation likelihood.

B. Experimental conditions

We use the publicly available code from the authors' websites without modifications for all the adaptive trackers and FragTrack, while MS and the color–based PF were implemented according to the descriptions in the original papers.

All trackers are initialized with the bounding box from the ground truth. As for parameters, we use the values reported in the original papers or in the available implementations. For Boost, SemiBoost and BeyondSemiBoost, we used 100 selectors working on a pool of 1000 features and a search window as large as four times the current window size. For MILBoost, we used 50 selectors working on a pool of 250 features and a search radius of 25 pixels. Color histograms for the PF and MS trackers are built in the RGB color space, with 4 bins for each channel. The gray–level histograms for FragTrack have 16 bins, the search window has a radius of 7 pixels and the metric to compare histograms is a variation of the Kolmogorov-Smirnoff statistic. STRUCK uses a search radius of 30 pixels, the parameter C of the SVM is 100, a budget of 100 support vectors and $\sigma = 0.2$ for the RBF kernel. For TLD, the threshold for target similarity during learning is 0.65 and the threshold to detect the absence of the target is 0.7. For trackers relying on particle filtering, we use 600 particles for IVT, 200 for the color-based PF and 20 for A-BHMC, since the Basin Hoping Monte Carlo filter can rely on a smaller amount of particles. The standard deviation for the noise of the transition model for the center of the bounding box is 10 pixels for PF, 3 for IVT and 10 for A-BHMC. The standard deviation for the noise of the transition model for the bounding box scales along the horizontal and vertical dimensions is 0.7 for PF, 0.15 and 0.045, respectively for IVT. The forgetting factor of IVT is 0.99. As for the observation likelihoods, IVT uses a standard deviation of 0.25 and A-BHMC uses 30 patches, 20 iterations, and 2.05 and 5.0 as the background to foreground similarity threshold and the density threshold, respectively.

C. Dataset

We use a dataset containing 5900 frames from 11 sequences (Fig. 4 and 5). The sequences are either selected from the original papers that describe algorithms under test or are standard tracking-evaluation sequences. We also include a sequence acquired for the evaluation, which includes all the considered challenges in one sequence. Table III summarizes the main characteristics of the test sequences used in the evaluation.

The overall dataset allows for assessing the performance of trackers and testing their robustness with respect to different working conditions. *Dollar* from the MILBoost dataset[2] and *OneStopMoveNoEnter1cor* from the CAVIAR dataset[3] exhibit both sharp and smooth appearance changes interleaved with several frames of constant appearance. *Dollar* has no clutter or illumination changes. This sequence is useful to assess the degree of adaptiveness of algorithms in a controlled and predictable situation. *OneStopMoveNoEnter1cor* helps testing adaptiveness to appearance changes due to the target deformation and rotation as well as to illumination changes. The PROST dataset[4] presents objects wandering in front of a static, highly cluttered background. In *Board*, the target undergoes a sharp and fast appearance and aspect ratio change when the board is suddenly rotated. However, the main challenge in this sequence is the target initialization misalignment even when using the ground truth. In *Box* and *Lemming*, the target is small and largely untextured and it undergoes partial occlusions. Moreover, both sequences have more than 1000 frames each, thereby allowing to assess also the performance of the trade-off between plasticity and stability associated with each method, that may not emerge in shorter sequences. We use also the first 450 frames of *Box*, *Box450*, to analyze robustness to partial occlusions. *Faceocc2* shows a face with frequent occlusions and a permanent appearance change in the middle of the sequence (a hat is worn by the target), followed by another occlusion. The first part of *ThreePastShop2cor* from the CAVIAR dataset is characterized by total occlusions of the target while he walks behind two other people at the beginning of the sequence. *Person*, the additional sequence, is characterized by two short-term total occlusions. In addition, the target undergoes severe lighting and scale changes. *Coke*, from the MILBoost dataset, presents partial and total occlusions of a small and untextured target that undergoes rotations causing appearance changes. Moreover, an artificial light is placed very close to the target, causing reflections and illumination changes. Finally, *OneStopEnter2front* from the CAVIAR dataset and *ISSIA_6* from the ISSIA dataset [47] exhibit small targets, a cluttered background and moderate variations of illumination and target appearance.

VI. Comparison

This section discusses the results of our experimental comparison based on the experimental setup presented in the previous section. A summarizing discussion concludes the section.

A. Analysis of the results

We will present and discuss the results based on the analysis of the behavior of the tracker with respect to the following tracking challenges: rare and continuous appearance changes, partial and (short-term) total occlusions, and initialization errors. We will also analyze the behavior of trackers under wide angle views.

To visualize bounding boxes for each sequence, we report the result of the closest run to the mean performance of each stochastic tracker (*i.e.* the run that minimizes throughout the sequence the $l1$ norm of the Dice with respect to the mean run)[5].

1) Rare appearance changes: As can be observed in Fig. 3, the best performing trackers on *Dollar* and *OneStopMoveNoEnter1cor* are TLD, STRUCK and IVT. All these trackers are able to adapt to the new appearance of targets and, in *Dollar*, do not confuse the target with the distractor in any run. The reason for their success lies in different configurations of the adaptive framework. Both STRUCK and TLD are discriminative trackers and their performance vouches the importance of background samples' selection for the classifier update: by letting the classifier itself participate

[2]http://vision.ucsd.edu/∼bbabenko/project_miltrack.shtml, last accessed: 25th January 2012.

[3]Data coming from the EC Funded CAVIAR project/IST 2001 37540, found at URL: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/, last accessed: 25th January 2012.

[4]http://gpu4vision.icg.tugraz.at/index.php?content=subsites/prost/prost.php, last accessed: 25th January 2012.

[5]All the results are available at http://www.vision.deis.unibo.it/adaptiveAppearanceTracking.html. The results are also provided as supplementary downloadable material for this paper at http://ieeexplore.ieee.org.

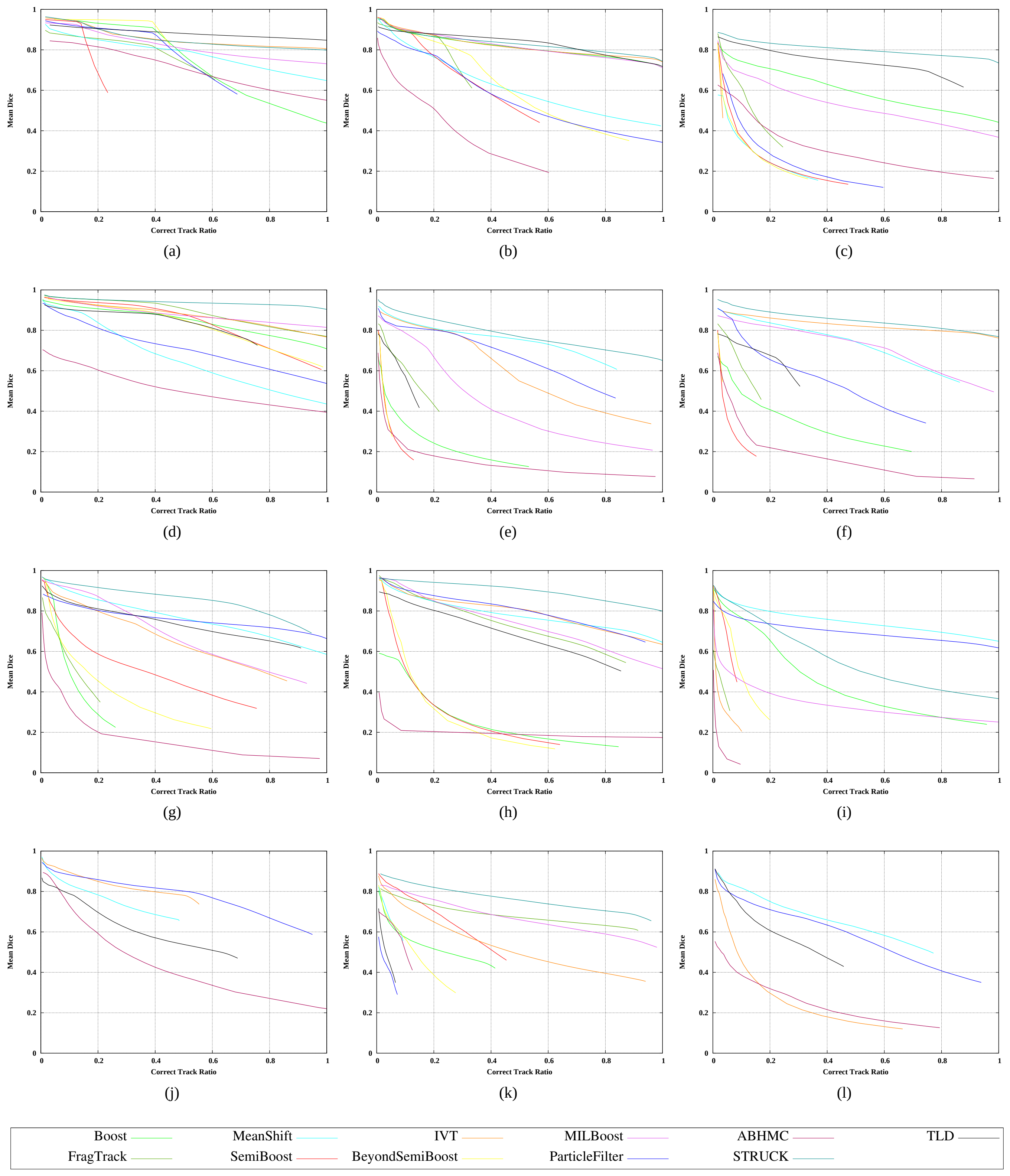


Fig. 3. Objective evaluation using the proposed methodology for performance comparison, which is independent of thresholds. Trackers that do not intersect the line CTR = 1 (CTR: Correct Track Ratio) did not track the target for the whole sequence. (a) *Dollar*. (b) *OneStopMoveNoEnter1cor*. (c) *Coke*. (d) *Faceocc2*. (e) *Box*. (f) *Box450*. (g) *Lemming*. (h) *Board*. (i) *ISSIA_6*. (j) *ThreePastShop2cor*. (k) *OneStopEnter2Front*. (l) *Person*.

TABLE III
SUMMARY OF THE CHARACTERISTICS OF THE DATASET USED FOR THE EXPERIMENTAL EVALUATION (FPS: FRAMES PER SECOND, OBJECT SIZE: SIZE IN PIXEL OF THE TARGET AT INITIALIZATION).

	Dataset	Frames	Frame Size	FPS	Object size
Dollar	MILBoost	0 - 327	320x240	10	62x98
Faceocc2	MILBoost	0 - 820	320x240	10	92x116
Coke	MILBoost	0 - 292	320x240	10	24x40
OneStopEnter2front	CAVIAR	684 - 837	384x288	25	30x60
OneStopMoveNoEnter1cor	CAVIAR	110 - 486	384x288	25	50x115
ThreePastShop2cor	CAVIAR	330 - 590	384x288	25	95x35
ISSIA_6	ISSIA	415 - 920	1920x1080	25	70x120
Box	PROST	0 - 1161	640x480	30	86x112
Board	PROST	0 - 698	640x480	30	195x153
Lemming	PROST	0 - 1336	640x480	30	69x113
Person	-	627 - 1435	640x480	30	33x111

in samples selection, either by maximizing the gradient for the SVM objective function (which basically selects samples with low overlap with the target and high confidence scores) or by selecting false detections with high confidence from the detector output, both methods can select as negative samples those more challenging for the current model. IVT, instead, despite discarding background information, outperforms the other trackers due to its high stability (subspace representation, see Tab. I) and ability to include rotation and scale in the state representation.

Adaptive trackers deploying priors are not successful. SemiBoost cannot deal with the sudden appearance change of *Dollar* (the red bounding box is not present in Fig. 4b) nor the target deformation in the CAVIAR sequence (the red bounding box is not present in Fig. 4f and Fig. 4h). The algorithm terminates the track due to the fixed external labeler used for sampling and labeling, which is trained on the first frame and hence cannot recognize the target after the appearance change. Interestingly, when the distractor appears in *Dollar* or when the target assumes a pose similar to the initial one in the CAVIAR sequence, the tracker tracks again (as shown in Fig. 4d and Fig. 4g). Unlike SemiBoost, BeyondSemiBoost can deal with the sudden appearance change (Fig. 4b) and handles target deformation (Fig. 4f). This is due to a slowly evolving adaptive prior in combination with a fixed prior from the first frame. However, both mechanisms are prone to errors. For example, the fixed prior misleads the tracker when the distractor appears in *Dollar* and the tracker drifts from the target (Fig. 4c). The use of an adaptive prior is detrimental in CAVIAR when the target moves in front of the window: the target bends toward it, uncovering parts of the background in the bounding box which are included in the tracker's prior appearance model making it lose the target (Fig. 4g and 4h).

Boost is not bound to the initial appearance by a prior and therefore should avoid the distractor and deal with the sudden change in *Dollar*: although it overcomes the appearance change, in several runs it tracks the distractor as soon as it appears, much like SemiBoost (Fig. 4c). This illustrates a limitation of discriminative trackers: if the most discriminative features with respect to the background come from a part of the object, the discriminative model concentrates only on them without attempting to obtain a full description of the object appearance. Similarly, MILBoost shows a great variability in its results: in the presence of similar objects, not even the current–state–centered sampling strategy in combination with the MIL framework can guarantee selection of the correct patches for model update.

A–BHMC is less stable than other trackers: as it allows its patches to move independently and works under the constant brightness assumption, it excludes from the appearance model parts of the target that change abruptly (Fig. 4b). Some patches are attracted by the distractor as it appears close to the target (Fig. 4c). Therefore, the output of the tracker lies halfway between target and distractor (Fig. 4d).

MS and the color–based PF can partially handle the challenges of these scenes, such as the target deformation in CAVIAR and the sudden appearance change in *Dollar*, due to the use of color histograms. However, they suffer from illumination changes in CAVIAR. In particular, the sharp color change on the trunk encourages non–adaptive solutions to exclude it from the bounding box, as shown in Fig. 4f. PF suffers also from the distractor in *Dollar* and exhibits a large variance. The use of part-wise features in FragTrack has mixed effects. It does not allow the tracker to avoid the distractor in *Dollar* but it is effective in CAVIAR, where the tracker follows the target until the end of the sequence, despite target deformations.

2) Continuous appearance changes: STRUCK is the best performer in *Coke* (Fig. 3c). TLD obtains good Dice overlaps but loses the target twice, whereas Boost and MILBoost are the only trackers, together with STRUCK, able to follow the target for the whole sequence. The difference in performance between STRUCK and Boost/MILBoost, which use the same features (Haar wavelet), illustrates the importance of the sampling and labeling step that is effectively guided by the classifier state in STRUCK.

TLD is mainly distracted by an occlusion at the beginning of the sequence (Fig. 4i). This is an inherent limitation of TLD: if at the beginning of a sequence the Lucas-Kanade tracker provides wrong training samples (as it happens during a partial occlusion), the trained detector becomes a persistent source of errors (Fig. 4j and 4l). Nevertheless, the method resists this initial noise: the tracker does not adapt completely to the occluder and overcomes both errors while the target keeps moving (Fig. 4k).

The longevity of Boost and MILBoost tracks stems from their continuous adaptation without priors and the use of a part-based model. The use of priors in SemiBoost and Be-

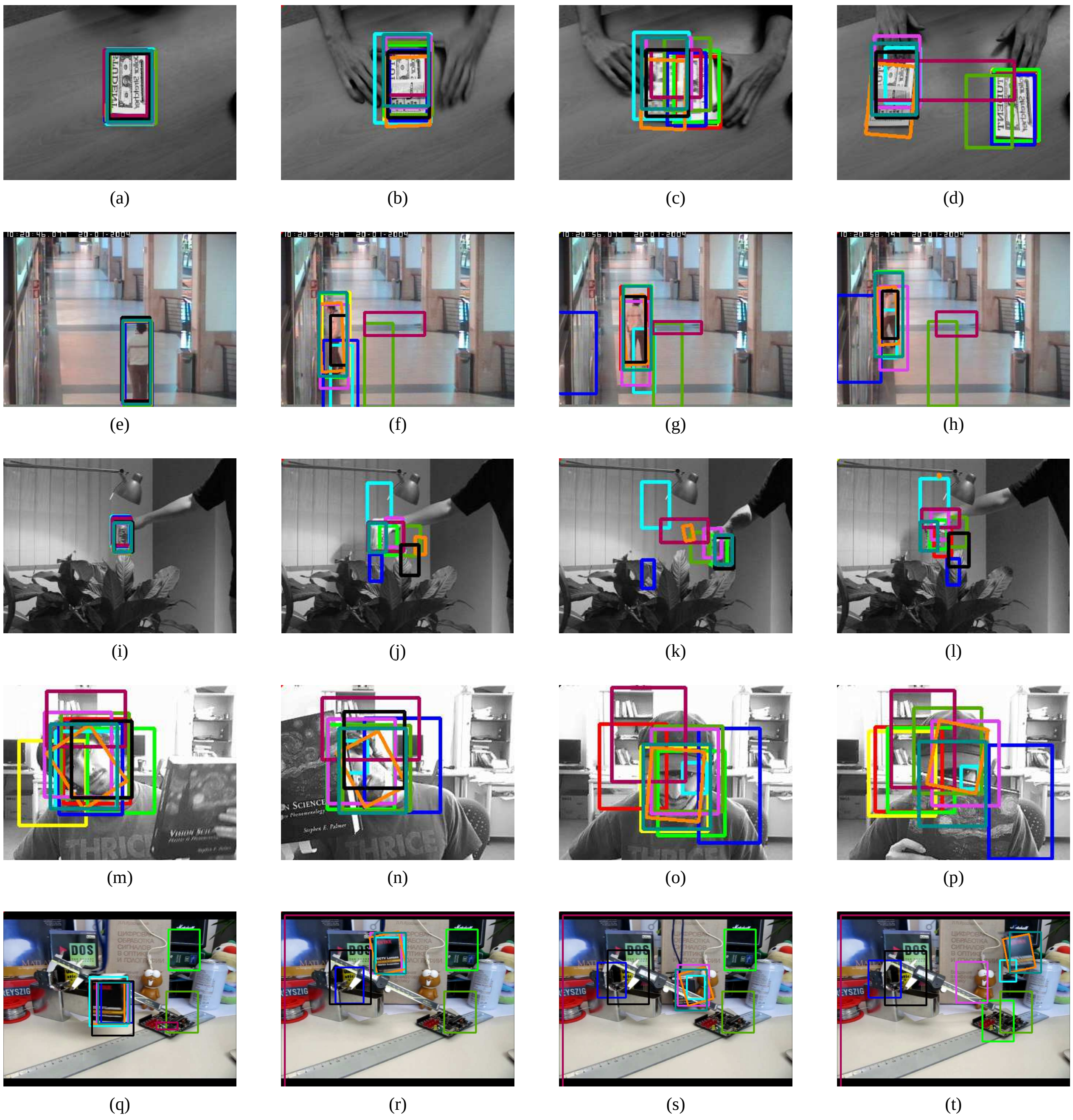
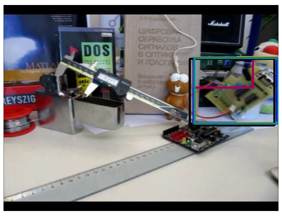


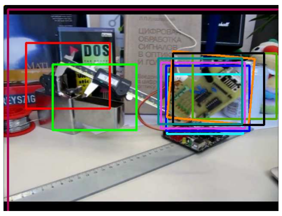
Fig. 4. Sample tracking results from the *mean run* of trackers on selected sequences. The *mean run* is the run minimizing throughout the sequence the $l1$ norm with respect to the Mean Dice of all the runs of a tracker (as defined in Sec. IV). From top to bottom: *Dollar*, *OneStopMoveNoEnter1cor*, *Coke*, *Faceocc2*, *Box*. Color code: Boost (light green); SemiBoost (red); BeyondSemiBoost (yellow); MS (cyan); PF (blue); FragTrack (dark green); A–BHMC (purple); IVT (orange); MILBoost (pink); TLD (black); STRUCK (dark cyan). (a) 1. (b) 90. (c) 130. (d) 225. (e)143. (f) 252. (g) 393. (h) 461. (i) 5. (j) 47. (k) 65. (l) 102. (m) 358. (n) 498. (o) 666. (p) 718. (q)184. (r) 249. (s) 313. (t) 353.

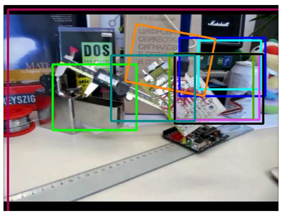
yondSemiBoost does not allow them to cope with a sequence showing several sudden appearance changes. Moreover, the prior is not informative as the object is relatively untextured, small and similar to the background. The use of salient regions in A–BHMC makes it lose the target when an untextured side of the object becomes visible (Fig. 4k). IVT loses the target immediately (Fig. 4j), as it does not have enough time to create an effective subspace representation for the appearance of the target during initial frames, when the target keeps rotating. Due to their high stability, subspaces and manifolds are not appropriate to cope with this sequence. Finally, non–adaptive trackers lose the target when it starts rotating (see Fig. 4j).

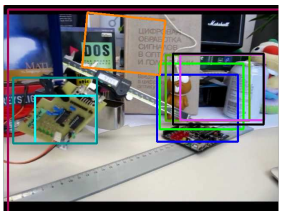
On *Person*, we tested only those trackers that include the scale as a state variable. No tracker is able to track the target for the whole sequence (Fig. 3l) and the overlap of all the methods is low. Interestingly, the best performers are the non-adaptive trackers, MS and PF. Adaptive trackers lose the target at the first appearance change caused by non–uniform lighting of the environment (Fig. 5r).

3) Partial occlusions: STRUCK exhibits the best performance during partial occlusions on *Faceocc2* and *Box450* (Fig. 3d and 3f), followed by IVT. MILBoost, Boost and FragTrack


(a)


(b)

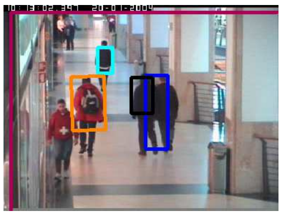

(c)

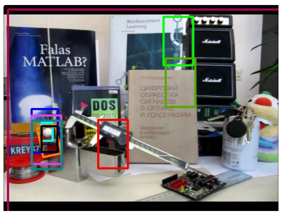

(d)

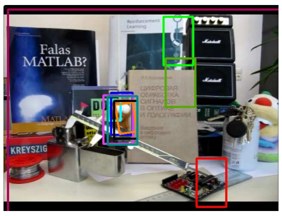
(e)

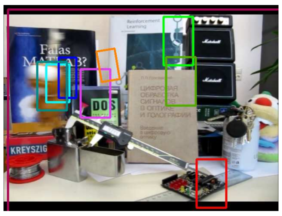
(f)

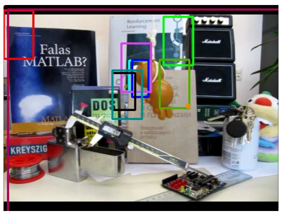
(g)


(h)


(i)


(j)


(k)


(l)

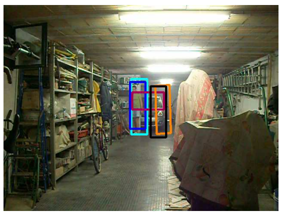
(m)

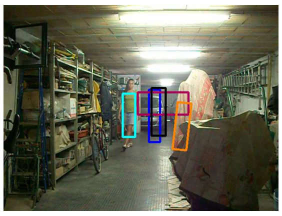
(n)

(o)

(p)

(q)


(r)


(s)

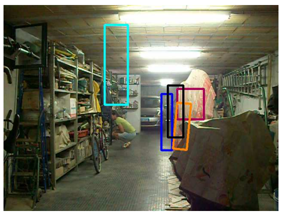

(t)

Fig. 5. Sample tracking results from the *mean run* of trackers on selected sequences. The *mean run* is the run minimizing throughout the sequence the $l1$ norm with respect to the Mean Dice of all the runs of a tracker (as defined in Sec. IV). From top to bottom: *Board*, *ThreePastShop2cor2*, *Lemming*, *OneStopEnter2front*, *Person*. Color code: Boost (light green); SemiBoost (red); BeyondSemiBoost (yellow); MS (cyan); PF (blue); FragTrack (dark green); A–BHMC (purple); IVT (orange); MILBoost (pink); TLD (black); STRUCK (dark cyan). (a) 106. (b) 167. (c) 543. (d) 679. (e) 330. (f) 443. (g) 467. (h) 506. (i) 359. (j) 454. (k) 548. (l) 985. (m) 688. (n) 710. (o) 754. (p) 821. (q) 627. (r) 707. (s) 877. (t) 1127.

are effective only in *Faceocc2*.

STRUCK is able to include the appearance of the occluder among positive support vectors without losing the past appearances (it uses a sufficiently stable strategy for model update, *i.e.* ranking, together with a classifier robust to wrong training samples). A good balance between stability and adaptation is offered also by IVT. Both trackers adapt to the inclusion of the hat (Fig. 4o) while rejecting occluders (Fig. 4p), as well as to the box rotations, motion blur and scale changes (Fig. 4r, 4s, 4t). IVT also adapts to the head-turning (Fig. 4n). Their success shows the importance of combining a stable model update strategy to compensate for the over-simplistic current-state sampling strategy. MILBoost is successful in *Faceocc2* because it performs on-line features selection, which allows the rejection of samples coming from the occluders and always sticks to the visible part of the face. This strategy turns out to be not robust in *Box450* for two reasons: the visible parts of the objects are mainly untextured; the occlusion occurs when the target appears smaller than in the first frame (Fig. 4s), therefore, some background texture is included into the bounding box. Both factors lead to drift (Fig. 4t).

TLD is effective at handling occlusions (Fig. 4m and 4n) but is unable to handle permanent appearance changes: it adapts to the hand when the subject is wearing the hat and drifts. Adaptive trackers deploying external labelers for the sampling and labeling stage (SemiBoost, BeyondSemiBoost) show good performance in *Faceocc2* until the head turns at frame 300 (Fig. 4m). The use of strong priors on target appearance is an effective solution for partial occlusions (see Fig. 4n, where the target is considered lost and, as a consequence, the occluder appearance is not included into the appearance model), but limits adaptability to appearance changes (Fig. 4o and 4p). When partial occlusions are coupled with changes in appearance as in *Box450*, the use of strong priors is ineffective (Fig. 3f). A–BHMC shrinks its bounding box in the presence of occlusions because of its highly flexible appearance model. It then focuses on smaller, uniform areas (Fig. 4n) that lead to drift (Fig. 4o). In *Box450*, the absence of textured regions onto the target and its blurred appearance in the initial frames make it lose the target.

PF fails in both sequences (Fig. 4n and Fig. 4r) whereas MS has similar performance to MILBoost in *Box450*, thanks to the invariance of the color histogram to appearance changes. FragTrack is robust to partial occlusions by design (Fig. 3d). However, the use of gray-level histograms is not discriminative enough to follow the untextured black target in *Box450*.

4) Short–term total occlusions: TLD, SemiBoost and BeyondSemiBoost explicitly model short–term total occlusions. We employ *Box* and *Lemming* for this analysis and also *ThreePastShop2cor* for trackers capable to adapt to scale changes. TLD is the only scale-adaptive tracker that correctly recovers the target after the first short–term total occlusion in the CAVIAR sequence (Fig. 5g), though it does not overcome the second one, ending onto the nearby distractor (Fig. 5h). Moreover, tracking of the scale is unsatisfactory even before the first occlusion, yielding low d_k (Fig. 3j). On the other hand, IVT correctly estimates the scale before the occlusion but stays on the occluder and adapts to it (Fig. 5h). Although IVT uses as appearance model a stable subspace in the feature space, the occlusion is too long for it to help. This happens in *Box* as well (Fig. 3j), where IVT can overcome the partial but not the total occlusion.

Among fixed-scale trackers, STRUCK is robust to short-term total occlusions (Fig. 3e and 3g): it is the only tracker which is able to follow the target in *Box* for the whole sequence and to exhibit higher overlaps than the color-histogram-based trackers in *Lemming*. *Lemming* is interesting also because other adaptive trackers (MILBoost, TLD, IVT) are robust to the total occlusion (Fig. 5i) as well as to the subsequent partial occlusion (Fig. 5j). They lose the target in the presence of rapid motion and consequent motion blur (Fig. 5k). STRUCK instead is misled by the out-of-plane rotation at the end of the sequence (Fig. 5l) and drifts (Fig. 3g).

5) Initialization errors: STRUCK is the most successful tracker in *Board* (Fig. 3h). Boost, SemiBoost and A-BHMC drift quickly (Fig. 5b) due to the inclusion of background in the first bounding box (Fig. 5a). The presence of background in the initial model is detrimental to handle rapid appearance changes: when the board turns, no tracker yields an acceptable overlap (Fig. 5c). Even more stable trackers, such as TLD, IVT and MILBoost, drift when the target moves away from the learned background structures (Fig. 5d).

6) Wide angle views: Many visual surveillance scenarios are characterized by small, untextured and highly deformable targets moving in a similar background and affected by lighting variations (*e.g. OneStopEnterTwo*Front, Fig. 5m). The best-performing algorithms are STRUCK, MILBoost and FragTrack that use gray-level, part-based features. MILBoost outperforms Boost because it is more robust to misalignment errors thanks to the current-state-centered sampling strategy and the MIL framework. Trackers relying on priors (SemiBoost, BeyondSemiBoost) are not reactive enough to adapt to changes of geometry and appearance and are distracted by clutter (Fig. 5o). Color-histogram-based trackers such as PF and MS fail as the background has a higher similarity to the appearance model than the target. Finally, the subspace update strategy deployed by IVT is too stable to deal with the continuous appearance changes of the target.

B. Discussion

STRUCK achieves a good trade-off between plasticity and stability based on the ranking of support vectors and the use of the on-line structured output SVM framework to guide the selection of challenging samples from the background; it is effective in dealing with rare as well as continuous appearance changes; and it is robust to partial and total occlusions and misaligned initial states. IVT is reliable in the presence of rare appearance changes, when the tracker has enough time to learn the new appearance after a change. However, a stable strategy, such as subspace fitting, prevents adaptation to sudden appearance changes and the use of templates is not always effective in the presence of highly deformable targets. When the target appearance is continuously changing, more plastic solutions such as MILBoost and TLD are preferable. Yet, MILBoost sampling and update strategy can be influenced by the presence of clutter and distractors and its results have usually a higher variance than those yielded by IVT. A non-adaptive solution like FragTrack is more effective than many adaptive trackers, *e.g.* IVT and MILBoost, during partial occlusions.

The use of part–wise features in STRUCK, FragTrack and MILBoost is key to deal with partial occlusions. In fact, FragTrack relies on a grid of gray level histograms and MILBoost, TLD and STRUCK use a set of Haar wavelets. When using target-wise features, robustness to partial occlusions can be achieved with a stable model update strategy, like subspace or manifold update. This motivates the good performance of IVT. Instead, the flexible model used by A–BHMC is prone to over-fitting while adapting, which results in drifts. Temporal smoothness must be enforced on the model to avoid drift due to over-fitting. In a real-world sequence like *Person* (Fig. 3l), no adaptive trackers provide acceptable performance. Finally, in *ISSIA_6*(Fig. 3i) all adaptive trackers and FragTrack fail. Only trackers using color succeed.

The conclusions of the experimental results are summarized in Tab. IV. The last column of the table reports the median frames per second of each algorithm on the whole dataset. The

TABLE IV
OVERALL SUMMARY OF THE PERFORMANCE OF TRACKERS UNDER ANALYSIS. THE SYMBOLS FROM '−−' TO '++' INDICATE PROGRESSIVELY BETTER PERFORMANCE . '0' INDICATES THAT IN GENERAL PERFORMANCE IS NOT SATISFACTORY ALTHOUGH THE ALGORITHM IS ABLE TO TRACK THE TARGET IF LOW OVERLAPS OR HIGH VARIANCE IN THE RESULTS ARE ACCEPTABLE. LEGEND: F02: *FaceOcc2*; CAV1: *OneStopMoveNoEnter1Cor*; CAV2: *OneStopEnter2Front*; CAV3: *ThreePastShop2Cor*; FPS: FRAMES PER SECOND.

	Dollar, CAV1	*Coke*	*Board*	*FO2, Box450*	*Box, Lemming*	*CAV2*	*Person, CAV3*	Median FPS
STRUCK	++	++	+	++	++	+		8
TLD	++	+	−	+	+	−−	0	18
IVT	++	−−	+	++	−−	−	0	18
MILBoost	+	0	−	+	−−	0		12
FragTrack	0	−−	−	++	0	+		8
MeanShift	−	−−	+	−−	−−	−−	+	65
ParticleFilter	−−	−−	+	−	−	−−	+	8
Boost	0	0	−−	−	−−	−−		5
BeyondSemiBoost	−	−−	−−	−−	−−	−−		2
SemiBoost	−−	−−	−−	−	−−	−−		3
A-BHMC	−	0	−−	−−	−−	−−	−	8

majority of trackers used in the evaluation are implemented in C++ without optimizations. Only IVT and TLD are in MATLAB. The publicly available C++ implementation of MILBoost that we used is optimized by exploiting the Intel Performance Primitives. We can distinguish clearly two groups among adaptive trackers: IVT, TLD and MILBoost run at near real-time frame rates, whereas A–BHMC, STRUCK, Boost, SemiBoost and BeyondSemiBoost are slower. *ISSIA_6* slows down all the trackers, because of the larger frame size. IVT and TLD are the most efficient, despite being implemented in MATLAB.

VII. CONCLUSIONS

We presented a unified framework for model adaptation in video tracking as well as a comprehensive evaluation of recent solutions on a heterogeneous dataset. The evaluation is based on a novel objective methodology that allows one to compare adaptive trackers without the need of thresholds. Performance of adaptive trackers was discussed, together with the various design choices using the building blocks identified in the proposed framework. Four trackers exhibited good performance overall: the discriminative tracker STRUCK outperformed other approaches, followed by TLD, IVT and MILBoost. These trackers deploy stable model representations and updates that are based on support vectors selection, ranking or subspace fitting and update. Trackers that instead aim at improving the sampling and labeling stage with priors, either fixed or adaptive, such as SemiBoost and BeyondSemiBoost, at times do not correctly update the prior and get meaningful labeled samples over time. From the study, it has emerged that is preferable to focus on being robust to label noise by letting the model estimator influence samples selection.

This systematic survey allowed us to identify important open challenges, related to on-line feature evaluation, on-line parameter tuning and dealing with multi-modality. The *first open challenge* is the on-line evaluation of different types of features, similar to what was done off-line in [45]. For instance, most adaptive trackers work on gray level features only. Moreover, even when they perform feature selection, adaptive trackers deploy one type of feature only, for example, by selecting size and orientation of Haar wavelets. An important open issue is therefore to understand how to effectively evaluate and select the appropriate features for a target and the appropriate subset of these features for each part of a sequence. The *second open challenge* is how to expose the level of plasticity of an algorithm through a limited set of parameters that may be also easily tuned (on-line) by the algorithm itself. Several proposals use a ranking strategy for model update [13], [15], [18], [29], but this requires the ability to evaluate the tracker output on–line in the absence of ground truth [48]. So far, only the naive idea of evaluating the model candidates on the current frame has been exploited [13], [14], [18]. The *third open challenge* is the ability to deal with multi-modal hypotheses in the state space when performing appearance model update. This issue is not taken into account even by adaptive trackers that rely on the particle filter, such as IVT. Moreover, several adaptive tracking algorithms assume a simple dynamic model for filtering and data association.

Finally, since adaptive modeling algorithms have been extended to multi-target tracking [31], [34], [41], an important future direction of research is to study how to exploit multi-target states to improve the performance of appearance model adaptation (*e.g.* by detecting target-to-target occlusions) and how to evaluate it effectively.

REFERENCES

[1] E. Maggio and A. Cavallaro, *Video tracking: theory and practice*. Wiley, 2011.

[2] M. Isard and A. Blake, "CONDENSATION : Conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[3] M. Isard and J. MacCormick, "BraMBLe: A bayesian multiple-blob tracker," in *Proc. of the International Conference on Computer Vision (ICCV) - Volume 2*, Vancouver, BC, July 2001, pp. 34–41.

[4] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–575, 2003.

[5] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proc. of the European Conference on Computer Vision (ECCV) - Part I*, Copenhagen, Denmark, 2002, pp. 661–675.

[6] R. T. Collins, A. J. Lipton, and T. Kanade, "A system for video surveillance and monitoring," Robotics Institute at Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-00-12, 1999.

[7] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: Real-time surveillance of people and their activities," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 809–830, 2000.

[8] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi, "Robust online appearance models for visual tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1296–1311, 2003.

[9] I. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 810–815, 2004.

[10] R. T. Collins, Y. Liu, and M. Leordeanu, "Online Selection of Discriminative Tracking Features." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–43, 2005.

[11] S. Avidan, "Ensemble tracking," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR) - Volume 2*, San Diego, CA, 2005, pp. 494–501.

[12] F. Tang, S. Brennan, Q. Zhao, and H. Tao, "Co-tracking using semi-supervised support vector machines," in *Proc. of the International Conference on Computer Vision (ICCV)*. Rio de Janeiro, Brazil: IEEE Computer Society Washington, DC, USA, 2007, pp. 1–8.

[13] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR) - Volume 1*, New York, NY, 2006, pp. 260–267.

[14] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, Aug. 2011.

[15] Z. Kalal, J. Matas, and K. Mikolajczyk, "Online Learning of Robust Object Detectors During Unstable Tracking," in *Proc. of the International Conference on Computer Vision (ICCV) - Workshop on On-line Learning for Computer Vision*, Kyoto, Japan, 2009.

[16] S. Hare, A. Saffari, and P. Torr, "STRUCK: Structured Output Tracking with Kernels," in *Proc. of the International Conference on Computer Vision (ICCV)*, Barcelona, Spain, 2011, pp. 263–270.

[17] S. Grossberg, *Competitive Learning: from Interactive Activation to Adaptive Resonance*. Norwood, NJ, USA: Ablex Publishing Corporation, 1988, pp. 243–283.

[18] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proc. of the European Conference on Computer Vision (ECCV) - Part I*, Marseille, France, 2008, pp. 234–247.

[19] A. Adam, E. Rivlin, and I. Shimshoni, "Robust Fragments-based Tracking Using the Integral Histogram," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR) - Volume 1*, New York, NY, 2006, pp. 798–805.

[20] L. Lu and G. D. Hager, "A nonparametric treatment for location / segmentation based visual tracking," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, MN, 2007, pp. 1–8.

[21] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, 2010, pp. 1269–1276.

[22] ——, "Tracking of a Non-rigid Object via Patch-based Dynamic Appearance Modeling and Adaptive Basin Hopping Monte Carlo Sampling," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, FL, 2009, pp. 1208–1215.

[23] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, 2008.

[24] E. E. Zelniker, T. M. Hospedales, S. Gong, and T. Xiang, "A unified approach for adaptive multiple feature tracking for surveillance applications," in *Proc. of the British Machine Vision Conference (BMVC)*, London, UK, 2009.

[25] J. Fan, X. Shen, and Y. Wu, "Closed-loop adaptation for robust tracking," in *Proc. of the Eleventh European Conference on Computer Vision (ECCV) - Part I*, Hersonissos, Greece, 2010, pp. 411–424.

[26] A. Yilmaz, O. Javed, and M. Shah, "Object Tracking: A Survey," *ACM Computing Surveys*, vol. 38, no. 4, pp. 1–45, Dec. 2006.

[27] K.-C. Lee and D. Kriegman, "Online Learning of Probabilistic Appearance Manifolds for Video-Based Recognition and Tracking," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR) - Volume 1*, San Diego, CA, 2005, pp. 852–859.

[28] S. K. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Trans. on Image Processing*, vol. 13, no. 11, pp. 1491 –1506, Nov. 2004.

[29] S. Stalder, H. Grabner, and L. van Gool, "Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition," in *Proc. of the International Conference on Computer Vision (ICCV) - Workshop on On-line Learning for Computer Vision*, Kyoto, Japan, 2009, pp. 1409–1416.

[30] M. Yang, Z. Fan, J. Fan, and Y. Wu, "Tracking nonstationary visual appearances by data-driven adaptation," *IEEE Trans. on Image Processing*, vol. 18, no. 7, pp. 1633 –1644, July 2009.

[31] X. Song, J. Cui, H. Zha, and H. Zhao, "Vision-based multiple interacting targets tracking via on-line supervised learning," in *Proc. of the European Conference on Computer Vision (ECCV) - Part III*, Marseille, France, 2008, pp. 642–655.

[32] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proc. of the Eleventh Conference on Computational Learning Theory (COLT)*, Madison, WI, 1998, pp. 92–100.

[33] Q. Yu, T. B. Dinh, and G. Medioni, "Online Tracking and Reacquisition Using Co-trained Generative and Discriminative Trackers," in *Proc. of the Tenth European Conference on Computer Vision (ECCV) - Part II*, Marseille, France, 2008, pp. 678–691.

[34] M. Yang, L. Fengjun, X. Wei, and G. Yihong, "Detection driven adaptive multi-cue integration for multiple human tracking," in *Proc. of the International Conference on Computer Vision (ICCV)*, Kyoto, Japan, 2009, pp. 1554–1561.

[35] B. Han and L. Davis, "On-Line Density-Based Appearance Modeling for Object Tracking," in *Proc. of the International Conference on Computer Vision (ICCV)*, Beijing, China, 2005, pp. 1492–1499.

[36] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, "PROST: Parallel robust online simple tracking," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, 2010.

[37] W. T. Freeman and E. H. Adelson, "The Design and Use of Steerable Filters," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 10, pp. 891–906, 1991.

[38] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 871–987, July 2002.

[39] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, San Diego, CA, 2005, pp. 886–893.

[40] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 259–289, May 2008.

[41] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *Proc. of the International Conference on Computer Vision (ICCV)*, Kyoto, Japan, 2009, pp. 1515–1522.

[42] A. Bordes, L. Bottou, P. Gallinari, and J. Weston, "Solving multiclass support vector machines with LaRank," in *Proc. of the International Conference on Machine Learning (ICML)*, Corvallis, OR, 2007.

[43] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, pp. 137–154, May 2004.

[44] N. C. Oza, "Online ensemble learning," Ph.D. dissertation, The University of California, Berkeley, CA, USA, Sept. 2001.

[45] B. Stenger, T. Woodley, and R. Cipolla, "Learning to track with multiple observers," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, FL, 2009, pp. 2647–2654.

[46] A. Nghiem, F. Bremond, M. Thonnat, and V. Valentin, "Etiseo, performance evaluation for video surveillance systems," in *Proc. of the IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, London, UK, Sept. 2007, pp. 476 –481.

[47] T. D'Orazio, M. Leo, N. Mosca, P. Spagnolo, and P. L. Mazzeo, "A semi-automatic system for ground truth generation of soccer video sequences," in *Proc. of IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Genova, Italy, 2009, pp. 559–564.

[48] J. SanMiguel, A. Cavallaro, and J. Martinez, "Adaptive online performance evaluation of video trackers," *IEEE Trans. on Image Processing*, vol. 21, no. 5, pp. 2812 –2823, may 2012.

Samuele Salti received the M.Sc. degree in computer science engineering and the Ph.D. degree in computer science engineering from the University of Bologna, Bologna, Italy, in 2007 and 2011, respectively.

He has been a Post-Doctoral Researcher with the Department of Electronics, Computer Science, and Systems, Computer Vision Laboratory, University of Bologna, since 2011. In 2007, he visited the Heinrich-Hertz-Institute, Berlin, Germany, focusing on humancomputer interaction. In 2010, he visited the Multimedia and Vision Research Group, Queen Mary, University of London, London, U.K., to work on adaptive appearance models for video tracking. His current research interests include adaptive video tracking, 3-D shape matching, Bayesian filtering, and object recognition. He has co-authored 14 publications in international conferences and journals.

Dr. Salti was a recipient of the Best Paper Award Runner-Up at 3DIMPVT, the International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission in 2011. He serves as a reviewer for IEEE TRANSACTIONS ON SIGNAL PROCESSING, IEEE TRANSACTIONS ON IMAGE PROCESSING and a number of international conferences. He is a member of the IEEE and the GIRPR.

Andrea Cavallaro received the Laurea degree (summa cum laude) from the University of Trieste, Italy, in 1996, and the Ph.D. degree from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 2002, both in electrical engineering.

He served as a Research Consultant with the Image Processing Laboratory, University of Trieste, in 1996 and 1998, focusing on compression algorithms for very low bitrate video coding. From 1998 to 2003, he was a Research Assistant with the Signal Processing Laboratory, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. Since 2003, he has been with Queen Mary University of London, London, U.K., where he is a Professor of multimedia signal processing. He has authored more than 130 papers and published two books, *Multi-Camera Networks* (Elsevier, 2009) and *Video Tracking* (Wiley, 2011).

Prof. Cavallaro was the recipient of a Research Fellowship from British Telecommunications (BT) in 2004 and 2005, three Student Paper Awards from IEEE International Conference on Acoustic, Speech, and Signal Processing in 2005, 2007, and 2009, and the Best Paper Award from the IEEE Advanced Video and Signal-based Surveillance (AVSS) in 2009. He is an elected member of the IEEE Image, Video, and Multidimensional Signal Processing Technical Committee; he served as Technical Chair for IEEE AVSS 2011, the Workshop on Image Analysis for Multimedia Interactive Services in 2010, and the European Signal Processing Conference in 2008, and as General Chair for IEEE/ACM International Conference on Distributed Smart Cameras in 2009, the British Machine Vision Conference in 2009, and the IEEE AVSS in 2007. He is an Area Editor for the *IEEE Signal Processing Magazine* and an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING and the IEEE TRANSACTIONS ON SIGNAL PROCESSING.

Luigi Di Stefano received the degree in electronic engineering from the University of Bologna, Bologna, Italy, in 1989 and the PhD degree in electronic engineering and computer science from the Department of Electronics, Computer Science and Systems (DEIS), University of Bologna, in 1994.

He spent six months at Trinity College Dublin, Dublin, Ireland in 1995, as a Post-Doctoral Fellow. He is currently an Associate Professor with DEIS. His current research interests include computer vision, image processing, and computer architecture. He is the author of more than 120 papers and five patents.

Prof. Di Stefano is a member of the IEEE Computer Society and the IAPR-IC. He is a member of the Scientific Advisory Board of Datalogic Group.